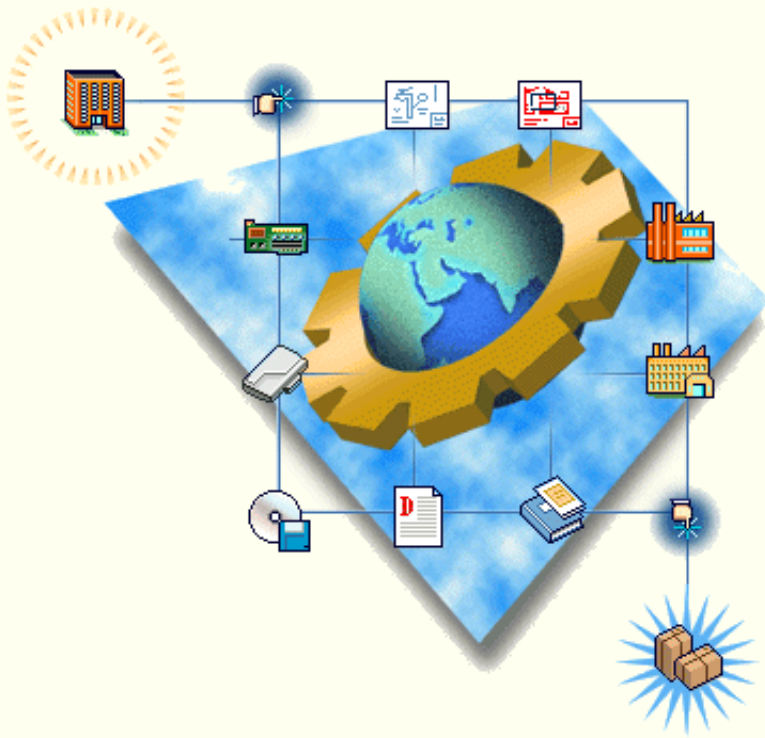


Application Exchange Framework™

Version 10.5



Administrator Guide



Copyright and Trademark Information

©Copyright 2003–2004 by MatrixOne Inc.

All rights reserved.

PROPRIETARY RIGHTS NOTICE: This documentation is proprietary property of MatrixOne Inc. In accordance with the terms and conditions of the Software License Agreement between the Customer and MatrixOne, the Customer is allowed to print as many copies as necessary of documentation copyrighted by MatrixOne relating to the Matrix software being used. This documentation shall be treated as confidential information and should be used only by employees or contractors with the Customer in accordance with the Agreement.

MatrixOne® is a registered trademark of MatrixOne, Inc.

AEF, Application Exchange Framework, MatrixOne Document Central, MatrixOne Engineering Central, MatrixOne Program Central, MatrixOne Sourcing Central, MatrixOne Supplier Central, MatrixOne Team Central, IconMail, ImageIcon, Primary Browser, Star Browser, and State Browser are trademarks of MatrixOne Inc. Oracle® is a registered trademark of Oracle Corporation, Redwood City, California. Orbix® is a registered trademark of IONA Technologies PLC, Dublin, Ireland. CORBA®, Object Request Broker™, ORB™, and IIOP™ are trademarks or registered trademarks of Object Management Group, Inc. in the U.S. and in other countries. All other product names and services identified throughout this book are recognized as trademarks, registered trademarks, or service marks of their respective companies.

This product includes software developed by the Apache Software Foundation. (<http://www.apache.org/>)

This product includes software developed by the OpenLDAP Project for use in the openLDAP Toolkit. (<http://www.openldap.org/>)

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (<http://www.openssl.org/>)

This product includes cryptographic software written by Eric Young. (eay@cryptsoft.com)

This product includes GifEncoder and ImageEncoder software written by Jef Poskanzer (jef@acme.com). Copyright 1996,1998. All rights reserved.

MatrixOne Inc.

210 Littleton Road

Westford, MA 01886, USA

Telephone: 978-589 4000

Fax: 978-589-5700

Email: info@matrixone.com

Web Address: <http://www.matrixone.com>

DM-MX-17-10-5

Table of Contents

	Copyright and Trademark Information	ii
Chapter 1.	Introduction.....	11
	Purpose, Scope, and Intended Audience	11
	Conventions Used in this Guide.....	12
	Requirements and Assumptions	13
	Overview	14
	Applications.....	14
	Framework Components	14
	Application Components	16
	Use of General Matrix Client Applications	17
	Related Documentation	18
	Framework Documentation	18
	MatrixOne Application Documentation.....	18
	Related Documentation Not Installed with the Framework or Applications.....	19
	How to Use this Guide	20
	Using the Online Help	21
	Administrative Object Names.....	24
Chapter 2.	Data Models	25
	Administrative Objects Used in Each Application	25
	How to Read Data Models	27
	Data Models for Applications	29
	MatrixOne Document Central	29
	MatrixOne Engineering Central.....	30
	MatrixOne Product Central	34
	MatrixOne Program Central	38
	MatrixOne Sourcing Central.....	41
	MatrixOne Specification Central	46
	MatrixOne Supplier Central.....	47
	MatrixOne Team Central	50
	Role Type Inheritance	52
	Common Component Data Models.....	53
	Data Common to Many Applications.....	56
Chapter 3.	Schema Definitions	61
	Renamed Administrative Objects	61
	Deprecated Schema	64
	Administrative Objects Used in Expression (Filter) Accesses	66
	Associations.....	72
	Attributes.....	73
	Business Objects	120
	Commands	121
	Formats.....	124

	Groups	125
	Menus	128
	Persons	130
	Programs.....	132
	Relationships	134
	Roles.....	166
	Stores.....	172
	Types.....	173
	Vaults	204
	Wizards	205
Chapter 4.	Policies	207
	Policy Access Tables	207
Chapter 5.	Setting Up and Configuring	371
	Setting Up	371
	Defining Your Company (The Host Company)	371
	Adding Employees for the Host Company.....	373
	Setting Up Groups.....	375
	Setting Up Vaults.....	376
	Overview of Configuring.....	379
	Things to Know Before Making Changes	379
	How the System Identifies Administrative Objects: Symbolic Name Properties ..	379
	Properties of Administrative Objects	381
	Viewing Administrative Properties.....	382
	How the System Identifies State Names.....	384
	Resolving Schema Similarities Between the Framework and Custom Schema...	385
	Reserved Prefixes, Names, and Characters	386
	Configuring the Schema	388
	Changing an Administrative Object Name	388
	Merging Schema	389
	Registering Your Own Administrative Objects.....	392
	Configuring State Names	396
	Configuring Range Values for Attributes	403
	Changing Onscreen Text.....	406
	Configuring the Content of System Notifications and Defining Company-Specific	
	Messages	407
	Configuring Automatic Business Object Naming	409
	Data Model for the Automatic Naming Process	409
	Object Generator Program	411
	How the Automatic Naming Process Works.....	412
	Configuring Automatic Naming.....	413
	Calling the Object Generator from JSP or Tcl.....	415
	Configuring Automatic Business Rules (Triggers).....	417
	How Rules are Automated	417
	Adding a Trigger for a Trigger Event.....	422
	Modifying Inputs and Execution Sequence for Trigger Programs.....	425
	Activating and Deactivating Triggers	425

Setting the Preferred File Store for a Company	426
Setting Up 2D and 3D Viewers	427
How the System Registers Viewers	427
Registering Viewers	428
Assigning Formats to Viewers.....	430
Specify File Extensions for Viewers	431
Specify Formats for which a Preferred Viewer Must Be Chosen	432
Choosing a Preferred Viewer	433
Modifying Supplied Source Code	434
Customizing User Documentation	435
Overview of User Documentation	435
Two Strategies for Making Changes	435
Changing the Documentation Output.....	436
Changing the Documentation Source	437
Configuring the File Collaboration Server for use with JSP Applications.....	438
Changing Shadow Agent Passwords.....	439
Configuring Approval Verification with External Authentication	440
Chapter 6. Internationalizing the Framework and Applications	443
Internationalizing the Applications	443
Resource Bundle File Naming	443
Introducing New Languages	444
Resource String Naming.....	444
Sample of an Internationalized JSP	445
Localizing the Date and Time	446
Debugging i18n.....	446
Translating Schema Names	447
Adding and Editing Translations for Schema	447
Implementing Schema Translations within a JSP	449
Sample Code for Getting Admin Object Translations.....	452
Installing Internationalized User Documentation	454
Internationalizing System-Generated Notifications.....	455
Specifying Default Languages to Include in System Notifications	455
Defining the Languages Users Can Choose for System Notifications	455
Chapter 7. Login and System Properties.....	459
Configuring the Login Process	459
Login Page.....	459
Requirements for Successful Login	460
Login Properties File Description.....	460
Secure ID and RSA Token Authentication	462
Single Signon.....	462
Configuring System Properties.....	464
Chapter 8. Configuring the User Interface.....	485
Overview of Dynamic User Interface	485
Components of the Dynamic User Interface	485
Naming Conventions for UI Administrative Objects	486
Internationalizing Dynamic UI Components.....	488
Using Macros and Expressions in Dynamic UI Components.....	488

Controlling User Access to User Interface Components.....	492
Access to an Application	492
Access Checks for Individual Components	492
Search Page Access	492
Access Parameters and Settings	493
Sample JPO for Controlling Access	495
Configuring Colors, Font Styles, and Images.....	497
Configuring the Banner (Header)	497
Configuring Styles	497
Configuring the Main Navigator Page	507
Configuring the Default Home Page for the Content Frame	507
Adding a Custom Application	511
Building Menus and the Global Toolbar	513
Types of Menus	513
Building the Menus	515
Configuring the Global Search Menu	520
Settings for Commands in My Desk	525
Building Navigation Trees.....	526
Overview of Navigation Trees.....	526
Two Modes for Tree Display	528
Structure Navigator	529
Building a Navigation Tree.....	530
Tree JavaScript API for Custom JSPs	538
Deprecated Methods	542
Parameters for Tree Menu Objects.....	543
Settings for Tree Menu Objects	545
Parameters for Tree Category Command Objects	547
Settings for Tree Category Command Objects.....	550
URL Parameters Accepted by emxTree.jsp.....	556
Using and Configuring the Default Tree	559
Calling Specific Trees for an Object Type.....	559
Configuring the Icon for a Type's Tree	560
Configuring Dynamic Expand for Tree Categories	561
Guidelines for Writing Structure Tree JPO	563
Building Table Pages.....	566
Overview of Table Pages.....	566
Table Body.....	566
How the Pagination Controls Work.....	567
Table Pages can Be Specific to an Object or Not.....	567
Methods for Filtering a Table	570
Building a Table Page.....	570
Configuring an Editable Table	575
Parameters for Table Objects	578
Settings for Table Objects	580
Refreshing the Table After Adding, Editing, or Deleting Objects	588
Getting the Object or Relationship ID of a Selected Table Item	589
Defining Table Column Data.....	589
Defining Editable Table Columns	597
Table Column Calculations.....	602
Improving Performance of Table Columns	605
Parameters and Settings for Inquiry Objects.....	609

URL Parameters Accepted by emxTable.jsp	611
Guidelines for Writing a JPO for Getting Object List	617
Sample JPO for Getting List of Objects	618
Writing a JPO for Updating Column Values	620
Implementing a Custom Filter	621
Custom Sorting Programs	624
Building Form Pages	627
Two Modes of the Form Page	627
Form Page HTML Components	629
Building a Form Page	629
Parameters for Web Form Objects	632
Settings for Web Form Objects	635
Using Action Menus with a Configurable Form	647
Decisions and Options Available for Form Fields	650
Defining Form Fields	651
Implementing Range Helpers for Choosers or Custom Pages	667
Validating Form Field Data	668
Committing Changes Made on Edit Form	670
JPO Interface for Form Fields	670
Sample JPO for Getting Field Values	674
Building Toolbars	679
Overview of the Configurable Toolbar	679
Drop-Down Menus Can Display to the Right	681
Building a Configurable Toolbar	682
Parameters for Toolbar Menu Objects	686
Settings for Toolbar Menu Objects	687
Parameters for Toolbar Link Command Objects	688
Settings for Toolbar Link Command Objects	689
URL Parameters Accepted by emxPortal.jsp	692
Implementing a Toolbar in a JSP	693
Building PowerView Pages	695
Two Kinds of PowerView Pages	696
Overview of PowerView Pages	698
Building a PowerView Page	699
Parameters and Settings for Portal Objects	707
Parameters and Settings for Channel Objects	707
Parameters for Tab Command Objects	707
Settings for Tab Command Objects	709
URL Parameters Accepted by emxPortal.jsp	709
URL Parameters Accepted by emxTable.jsp	710
Configuring Preference Pages	711
Overview of Preferences	711
Building Preference Pages	712
Configuring General Framework Preferences	717
Using Other Configurable Pages	727
Configurable History Page and History Commands	727
Configurable Type Chooser	732
Lifecycle Page	738
Standard Commands and Pages	740
Logout Command	740
Change Password Command	741

Page History Page and Command	742
Implementing Context-Sensitive Help	745
JSP Programming	747
Use Built-In Parameters to Make Pages Reusable	747
Standard Error Handling	747
Accessing the Applications Externally	750
Specifying the Content of the Navigator Page	750
Specifying the Location of the Navigator Page	752

Appendix A: Triggers and Programs 757

Utility Trigger Programs	757
Previous Revision Promotion	758
Required Connection Check	758
Required File Check	760
Check Relative State	761
Valid Revision Sequence	763
Relative Float Action	763
Set Originator Attribute	764
Automated Processes	765
Automation for All Types	765
Automation for Routes and Inbox Tasks	766
Automation for Organizations	769
Automation for Packages	770
Automation for Persons	770
Automation for Quotations	771
Automation for RFQs	771
Coding Examples	773
Using Push and Pop to a Shadow Agent	773
Setting a Target Page	774
Checking that the User is Logged In	774
Getting the User Context	775
Looking Up an Object by Symbolic Name	775
Opening an Object and Getting Information	775
Reading Attributes	775
Checking a User's Role	776
Displaying MQL Notices	776
Getting Admin Object Names from Symbolic Names	776
Getting Symbolic Names	777
Migrating to Version 10 Tree API for Custom Pages	779
Issue #1: Deleting Node & Reloading Tree Page via Custom JavaScript	779
Issue #2: Renaming Tree Nodes via Custom JavaScript	780
Preventing Similar Issues in the Future	781

Appendix B: Dynamic UI Parameters and Settings 783

Introduction	783
Administrative Object Parameters	784
Settings	791
URL Parameters	818
Parameters Automatically Passed to URLs	834

Appendix C: Conversion Routines	837
Introduction	837
Automatic Data Conversions.....	838
AEF 10.0.1.0 Installation	838
AEF 10.0.0.0 Installation	838
AEF 9.5.2.1 Installation	843
AEF 9.5.1.3 Installation	844
AEF 9.5.1.2 Installation	844
AEF 9.5.1.1 Installation	844
AEF 9.5.1.0 Installation	844
AEF 9.5.0.0 Installation	845
Manual Data Conversions.....	849
AEF 10.5 Installation	849
AEF 10.0.0.0 Installation	850
Index	855

Introduction

Purpose, Scope, and Intended Audience

This *Application Exchange Framework Guide* is a reference tool for Business Administrators who are responsible for configuring and maintaining MatrixOne™ applications or who are creating their own applications using the Application Exchange Framework™. Business Administrators who are building their own schema might also find the guide useful for getting examples and ideas.

Conventions Used in this Guide

- “Application” used in place of “suite”. In early versions of the MatrixOne Application, the “Centrals” (Supplier Central, Engineering Central, etc.) were called suites and most consisted of several distinct applications. For the most part, these distinct applications have merged together and each “Central” is now one unit. We now refer to the “Centrals” as applications and generally don’t refer to them as suites. This guide follows this naming convention. However, the term “suite” is still used for some property and parameter names and when it is, it refers to an entire “Central”.

Requirements and Assumptions

For information about system requirements for the Application Exchange Framework, refer to the readme document that accompanies the MatrixOne Application you are installing.

Before you begin administrative work with the Application Exchange Framework, you need to understand:

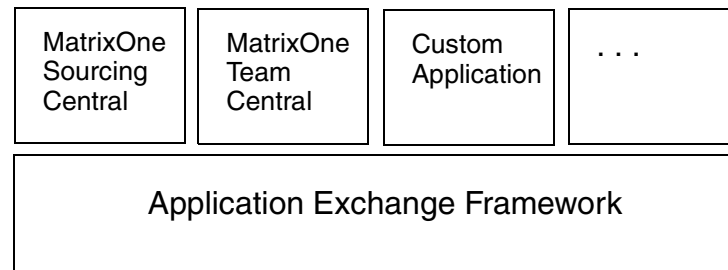
- The concepts, terminology, and processes related to Matrix Business Modeler. To help gain this understanding, see *Matrix Basics* and the *Matrix Business Modeler Guide*.
- The concepts, terminology, processes, and procedures for the MatrixOne Application you will be configuring and maintaining. Refer to the online guide that accompanies the application. The default location for these guides is `MATRIXHOME/framework/Doc` and `APP_SERVER_DIR/ematrix/doc`, where the `APP_SERVER_DIR` is either the document root for non-J2EE implementations or the staging directory under the EJB/RMI directory for J2EE installations.

Always refer to the current readme for any changes since the publication of the online guides.

Overview

The Application Exchange Framework contains schema for all the MatrixOne applications. As the name suggests, the Application Exchange Framework is the framework or foundation for the MatrixOne applications. It also installs the programs and JavaServer Pages needed to construct the user interface shared by all the applications. The framework must be installed before you can install any MatrixOne application (except for Info Central). The framework can also be used as the basis for creating your own applications.

MatrixOne applications



Applications

The framework currently supports the following MatrixOne applications.

- MatrixOne Document Central
- MatrixOne Engineering Central
- MatrixOne Info Central
- MatrixOne Product Central
- MatrixOne Program Central
- MatrixOne Sourcing Central (formerly Supplier Central – Sourcing)
- MatrixOne Specification Central
- MatrixOne Supplier Central (formerly Supplier Central – Quality)
- MatrixOne Team Central

Info Central is unique in that it does not require the framework although it is compatible with the framework.

Framework Components

The Application Exchange Framework contains the items listed in the following table.

Application Exchange Framework Components	
Item	For information, refer to:
associations	Associations in Chapter 3
attributes	Attributes in Chapter 3

Application Exchange Framework Components	
Item	For information, refer to:
business objects	The framework includes only a couple of business objects: – A Company business object called Company Name. See Setting Up in Chapter 5. – A Person business object named Test Everything that corresponds to the administrative object Test Everything. This person is assigned to all roles and is connected to the company business object called “Company Name” (the host company) with the Employee and Company Representative relationships. This person object is stored in the eService Production vault. – eService Trigger Program Parameters objects, which define parameters for triggers. The framework installs objects for triggers needed by several applications. Application install application-specific trigger information. See Configuring Automatic Business Rules (Triggers) in Chapter 5.
commands	Commands in Chapter 3
formats	Formats in Chapter 3
groups	Groups in Chapter 3
menus	Menus in Chapter 3
persons	Persons in Chapter 3
policies	Policies in Chapter 4
programs, including Java Program Objects (but not those that are specific to an application)	Programs in Chapter 3
relationships	To see the relationships used in each MatrixOne Application, see Data Models for Applications in Chapter 2. For a description of all relationships in the framework, see Relationships in Chapter 3.
roles	Roles in Chapter 3
stores	Stores in Chapter 3
types	To see the types used in each MatrixOne Application, see Data Models for Applications in Chapter 2. For a description of all types in the framework, see Types in Chapter 3.
vaults	Vaults in Chapter 3
wizards (for administrative uses only, not for business processes)	The framework contains several administrative wizards that allow administrators to change administrative and state properties. See Wizards in Chapter 3.

Application Components

Each MatrixOne application contains the items listed in this table.

Application Components	
Item	For information, refer to:
Web pages used by the application's users	The user guide that accompanies the application.
programs specific to the application	For general information on how to configure programs and for descriptions of utility trigger programs, see Chapter 5, <i>Setting Up and Configuring</i> . For descriptions of trigger programs specific to the application and to see where triggers are used in the application, see the Administrator Guide that accompanies the application. The Administrator Guides are in PDF format and are located in ematrix/doc/pdf.
other administrative objects specific to the application, such as formats	The Administrator's Guide for the application.
business objects that accomplish system-related tasks, such as objects for automatically-naming objects and for executing trigger programs	For general information on how the objects function and how to configure them, see Chapter 5, <i>Setting Up and Configuring</i> . For a list of the objects included in the application, see the Administrator Guide that accompanies the application.

Use of General Matrix Client Applications

Some of the instructions in this and other administrative guides require the use of a general Matrix client navigator, such as the:

- desktop version of Matrix Navigator (also known as the thick client)
- Web version of Matrix Navigator (also known as the thin client, PowerWeb, eMatrixApplet, and the Web Navigator)
- CWI (Collaborative Web Interface)
- Info Central

It's important to restrict the use of these general navigator applications to only a few specially-trained business administrators and to only the purposes described in the AEF Guide and applications' Administrator Guides. The MatrixOne applications run JavaBean code that requires data to have specific characteristics and conditions. For example, objects may have to have certain relationships defined, have specific values entered for attributes, be in specific lifecycle states, or be in particular vaults. When a person works within the MatrixOne Application user interface, these data conditions are met. But the general Matrix navigators are not necessarily aware of these conditions and therefore a person working within the general navigators can easily compromise data integrity.

There are situations in which the general navigators must or can be used, such as when:

- MatrixOne Application features require data that cannot be created within the MatrixOne Application user interface.

For example, some user profile information and template information must be created in a general navigator.

- Automated business rules and processes need to be configured, such as triggers and autonamers.
- Data needs to be investigated for troubleshooting, testing, or data conversion.

The general navigators should only be used in these situations, using the instructions provided in MatrixOne's documentation, and only by specially-trained business administrators. Standard users of the MatrixOne applications should never be allowed to work with their data in a general navigator and external customers should never be given access to a general navigator. Also, using custom ADK applications or any programming interface that does not go through the applications bean layer has the potential to cause undesirable results within the MatrixOne application data.

Related Documentation

This section lists the documentation available for the framework and MatrixOne applications. Also see [Installing Internationalized User Documentation](#) in Chapter 6.

- [Framework Documentation](#)
- [MatrixOne Application Documentation](#)
- [Related Documentation Not Installed with the Framework or Applications](#)

Framework Documentation

The framework installs with this documentation:

- *Application Exchange Framework Guide* (this guide)
This guide is available in html and pdf format. It is for people in the host company who need to configure and customize the MatrixOne applications and who will use the framework to build their own applications. It describes the schema that underlies the applications and how to configure it. The PDF version is located in MATRIXHOME/framework/Doc/pdf/Framework.pdf. To open the help system, open MATRIXHOME/framework/Doc/framework/launchhelp.htm.
- MatrixOne Application Help
This help system describes how to use features installed with the Application Exchange Framework, such as history pages and pages accessed from the global toolbar. It also explains how to navigate through the user interface, such as how to use table pages and the Context Navigator. Users can access this help system by clicking AEF Help at the top of any application help page.
- JavaDoc for the framework
For descriptions of methods in framework packages and classed, see MATRIXHOME/framework/Doc/javadoc/framework/html/index-all.html or STAGING/ematrix/doc/javadoc.

MatrixOne Application Documentation

The applications install with this documentation:

- User Guide and online help for each application
Users access online help for an application by clicking the Help (?) tool in the upper right corner of every page. The user guide is in pdf format and requires Acrobat Reader to view. To access the pdf version of an application's help system, click **View PDF Version** at the top of any help page.
These guides are for the people who will log in and use any part of the application, including Administrators who use the profile management portions of an application to manage person and company profiles.
- Common Components Administrator's Guide
This guide, CommonComponentsAdmin.pdf and located in ematrix/doc/pdf, describes configuration options for features installed with common components, such as checkin/checkout, routes, and the default vault preference. These configuration options include triggers, configurable properties, and URL parameters.
- Matrix Administrator's Guide for each application

Each application has a separate guide for host company administrators who work with Matrix applications. These are the same people who will use the Application Exchange Framework and the Matrix core applications (MQL, Business Modeler, Matrix Navigator) to configure an application. The Administrator's Guide for each application contains information that is unique for the application and therefore not appropriate for the *Application Exchange Framework Guide*. The Administrator's Guide comes in PDF format.

Related Documentation Not Installed with the Framework or Applications

A readme file is available for each version of the framework and each version of the MatrixOne applications. The readme files contain information about required software, new features, known issues, and closed issues. The readme files are in HTML format.

For instructions on installing the framework and applications, see the *Matrix PLM Platform Installation Guide* for the version of Matrix that is required for your application/framework.

How to Use this Guide

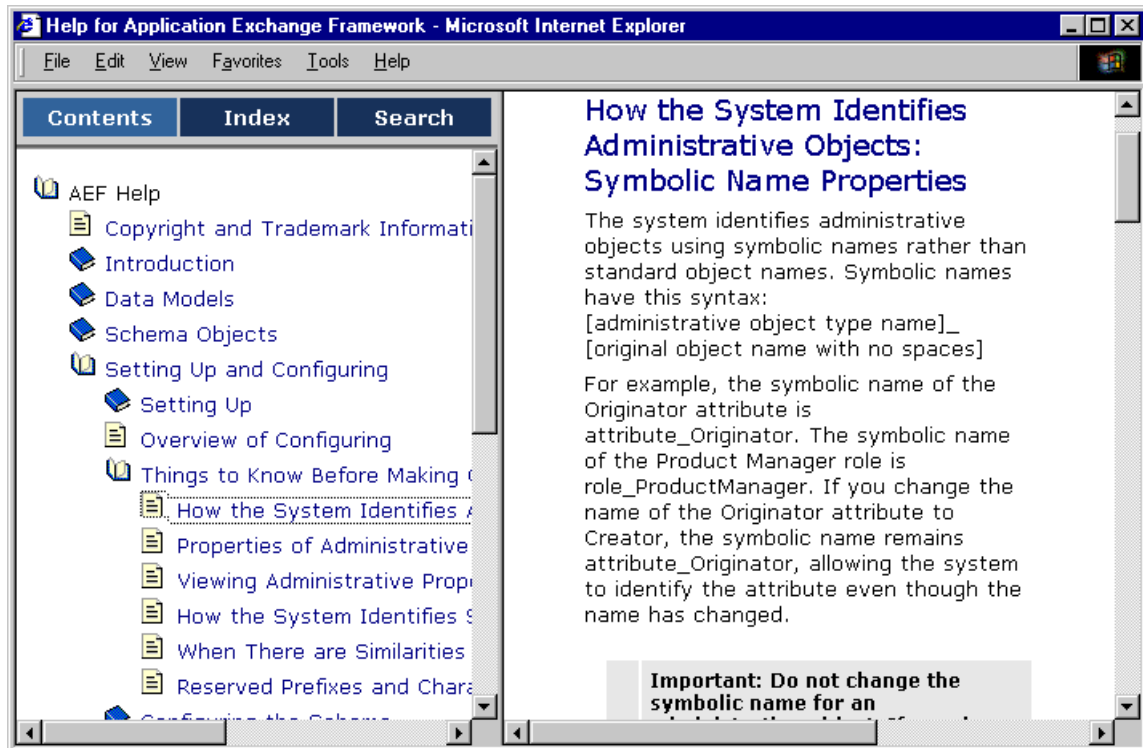
This *Application Exchange Framework Guide* is a reference tool for working with and configuring the supplied schema. Here are some suggestions for using this guide:

- [Chapter 2, *Data Models*](#) contains graphics that show the object types in each application and the relationships between the objects. The data models show the cardinality for the *from* and *to* end of each relationship. They also show type inheritance (which types are derived from other types).
- [Chapter 3, *Schema Definitions*](#) describes every administrative object that is installed with the Application Exchange Framework, except policies which are covered in the next chapter. The objects are listed alphabetically by administrative type. For example, all the attributes in the framework are listed alphabetically, regardless of which type, relationship, or application uses the attribute. The descriptions for roles and groups does include the applications that use each role/group. There is also a list of deprecated schema, schema used in policy expression accesses, renamed schema, and business objects installed with the framework.
- [Chapter 4, *Policies*](#) contains tables that list user accesses for every state in every policy, including filter expressions, whether objects are revisionable and versionable in each state, and other important policy data.
- [Chapter 5, *Setting Up and Configuring*](#) presents the information you'll need to configure the schema and applications for your exchange, including how to:
 - set up your company and its employees
 - change administrative object names and policy states
 - register your own administrative objects
 - control the names of objects that are created automatically by programs and wizards in the applications
 - change range values for attributes
 - add triggers to automate processes and configure existing triggers
 - register 2D/3D viewers
 - customize the documentation
 - configure company file stores
 - configure onscreen text and system notifications
- [Chapter 6, *Internationalizing the Framework and Applications*](#) describes how the schema and applications are internationalized and how to customize the translations.
- [Chapter 7, *Login and System Properties*](#) describes the login pages and lists login and system properties.
- [Chapter 8, *Configuring the User Interface*](#) contains information about how to configure user interface components and add new components, including how to:
 - build new applications and elements in an application using dynamic user interface components; this chapter describes the process for creating menus, tables, forms, and links
 - user standard and configurable pages included with the framework
 - configure links from external applications and emails to the MatrixOne Application

- [Appendix A: Triggers and Programs](#) describes the utility trigger programs included with the framework. These are trigger programs used by most of the MatrixOne applications. The descriptions include how to configure the programs. The appendix also contains information on how to code some common functions.
- [Appendix B: Dynamic UI Parameters and Settings](#) has alphabetical lists of all the parameters and settings used for the dynamic UI components.
- [Appendix C: Conversion Routines](#) describes automatic conversion routines run when a new version is installed, and manual conversion routines that may be necessary under certain circumstances.

Using the Online Help

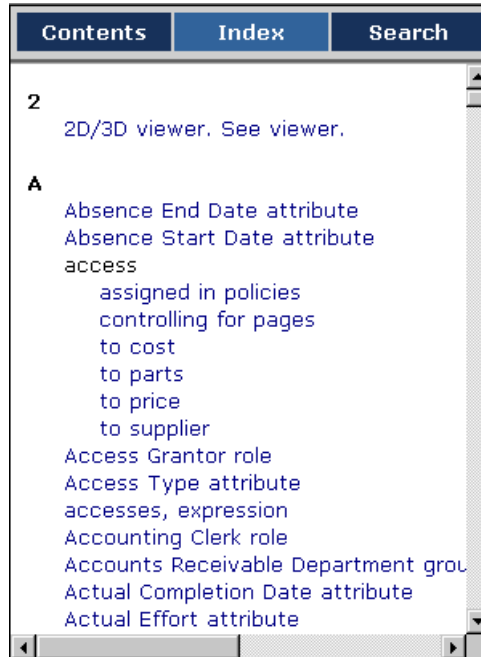
The HTML version of this guide has two frames. The left frame contains tools to help you find the topic you are looking for. The right frame contains the content.



There are several ways to look for the information you need:

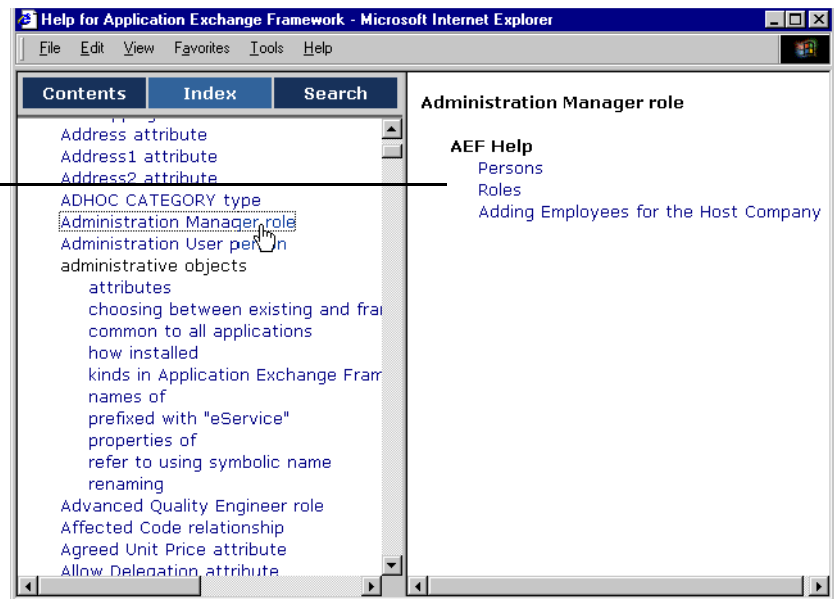
- **Choose a topic from the Table of Contents.**
 - a) Click the **Contents** box on the left side of the help window.
This frame shows the table of contents for the *Application Exchange Framework* guide.
 - b) To expand the contents, click a closed book.
 - c) To contract the contents, click an open book.
 - d) To go to a topic listed in the Contents frame, click the topic title.
- **Choose a keyword from the index list.**
 - a) Click the **Index** box.

- b)** Scroll to the keyword you are looking for, then click the keyword.



- c)** If more than one section of the guide discusses the keyword, the help system will present links for both.

The keyword "Administration Manager role" is discussed in three topics. Click on the blue links to see the topics.



- **Search for a word or phrase.**

- a)** Click the **Search** box.
- b)** Type the word or phrase that you want to search for. The search is not case sensitive.
The help system lists the topics that contain the word.
- c)** To find the word within a topic, click the topic name in the left frame.

The help system found the phrase “duns number” in two topics within the guide.



The screenshot shows a help system window with a search bar and results. The search bar contains the text "duns number" and a "Search" button. Below the search bar, the results are displayed under the heading "AEF Help". The results list two topics: "Attributes" and "Types", both in blue text. A line points from the text on the left to the search results.

Contents	Index	Search
duns number Search		
AEF Help		
Attributes		
Types		

Administrative Object Names

When you look at the administrative objects installed with the Application Exchange Framework, you may find that some object names are prefixed with “eService” and the version number of the installation. The installation program adds this prefix to prevent name collisions between objects in the framework and objects in the existing database. For example, if you install the version 8.0.0.0 framework onto a database that contains a Part type, the installation program renames the Part type within the framework to “eService8000~Part”. Other administrative objects within the framework that refer to the type—such as attributes for the type, relationships that connect the type, and policies that govern the type—refer to the framework Part type, now named “eService8000~Part.” The installation makes no change to the existing Part type.

To see a list of the name collisions the system found during installation, open the file called “installFrameworkx-x-x-x.log”, where x-x-x-x is the software version number. This file is in MATRIXHOME\framework\VERSION. The file also lists all the administrative objects installed or modified during installation. For instructions on configuring objects in the framework, using existing objects instead of the objects installed with the framework, and for information about the installation process, see [Chapter 5, Setting Up and Configuring](#).

Data Models

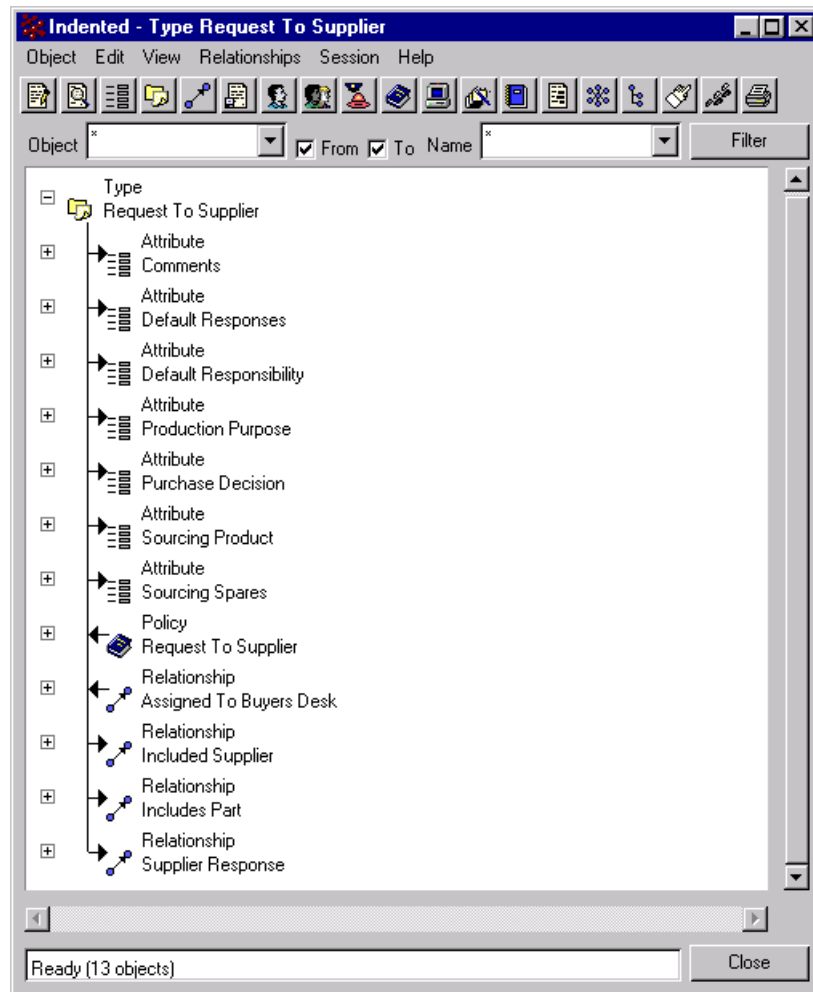
Administrative Objects Used in Each Application

The Application Exchange Framework contains hundreds of administrative objects, some of which are used by many MatrixOne applications and some of which are used by just one application. Use this section as a guide for finding the objects that belong to a particular application.

Info Central has no data model because it is independent of any particular kind of data.

These sections show data models for each application. A data model shows the most basic information needed to understand an application's schema: the business object types and the relationships that connect the types. Once you know the types and relationships, there are several ways to find the other administrative objects used in the application:

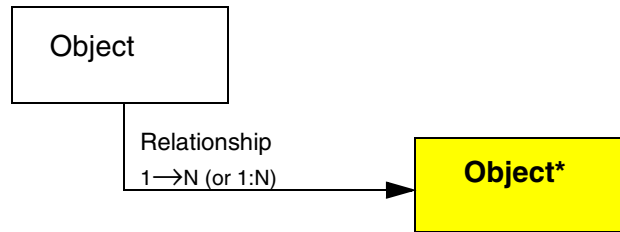
- In Matrix Business Modeler, expand on a type or relationship using the Star or Indented browsers. These browsers show you all related administrative objects. For example, you can see all the attributes, policies, methods, parent types, and relationships for an object type by selecting the type and choosing Indented or Star from the Relationships menu. The graphic below shows an expansion on the Request To Supplier type using the Indented browser. For instructions on using browser, see the Working with Relationship Browsers chapter of the *Business Modeler Guide*.



- Use the tables in [Chapter 3, Schema Definitions](#) to find related administrative objects. For example, the Type table lists the attributes assigned for a type. The Policy table lists the types each policy can govern and the policy states.
- In Matrix Business Modeler, double-click on an administrative object to see all the information about it. For example, double-clicking on a type lets you see the attributes assigned. Double-clicking on a relationship lets you see the types allowed on each end of the relationship.

How to Read Data Models

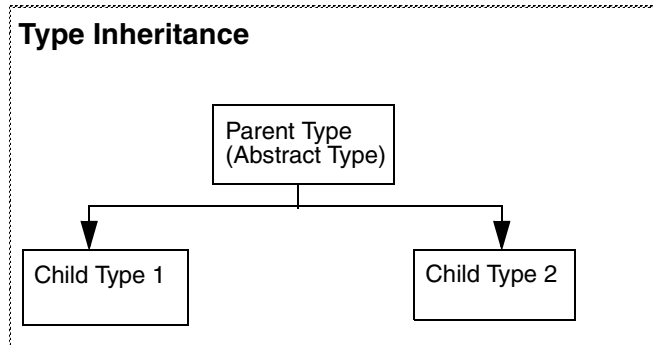
This graphic shows a sample data model.



Here's how to read the data model:

- Each rectangular box (or circle for some models) represents a type of business object. The above data model contains two object types.
- Object types and relationships that are “owned” by another MatrixOne Application are indicated with one or more asterisks. To say that an application owns an administrative object means it is the application that is primarily responsible for creating the object, defining the behavior, and using lifecycle processes associated with that object. The install scripts for the owner application are responsible for installing the object. Other applications often need to create objects that are not owned by them, but they usually use the objects within a limited scope. Ownership shows how applications are related and the types they have in common.
- Each large arrow represents a relationship between object types. The text next to the arrow shows the relationship name.
- The text under each relationship name represents the cardinality defined for the From and To sides of the relationship. “1” means the cardinality is One, and “N” means the cardinality is Many. For example, if the cardinality is 1→N or 1:N, then the cardinality on the From side is One and the cardinality on the To side is Many. N→N and N:N means the cardinality is Many for both sides.
- Objects and relationships highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

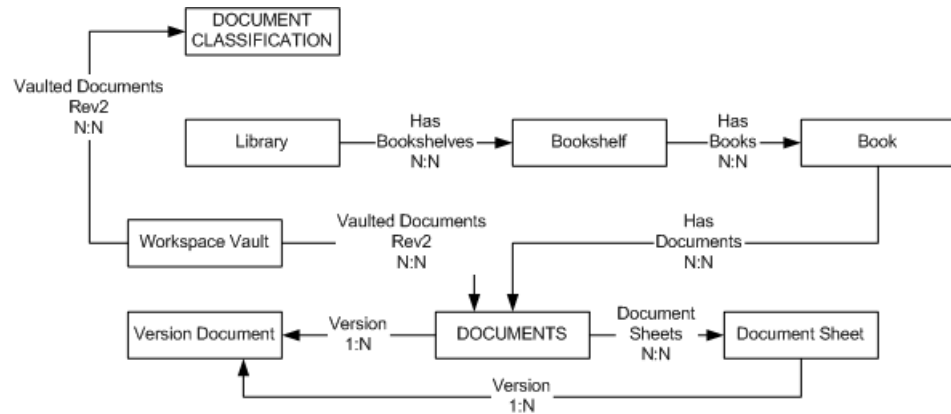
Data models also show any types that have child types. A child type inherits from its parent all attributes, methods, triggers, governing policies, and allowed types for relationships. Parent types are often abstract types but they don't have to be. For example, if a type is a parent of two types, the data model would include the following diagram:



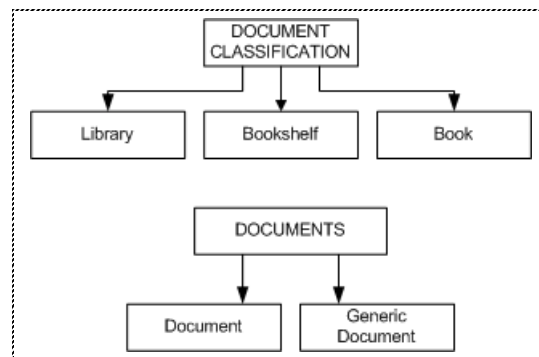
For descriptions of the object types and information about the attributes for the types and relationships shown in the data models, refer to [Chapter 3, *Schema Definitions*](#).

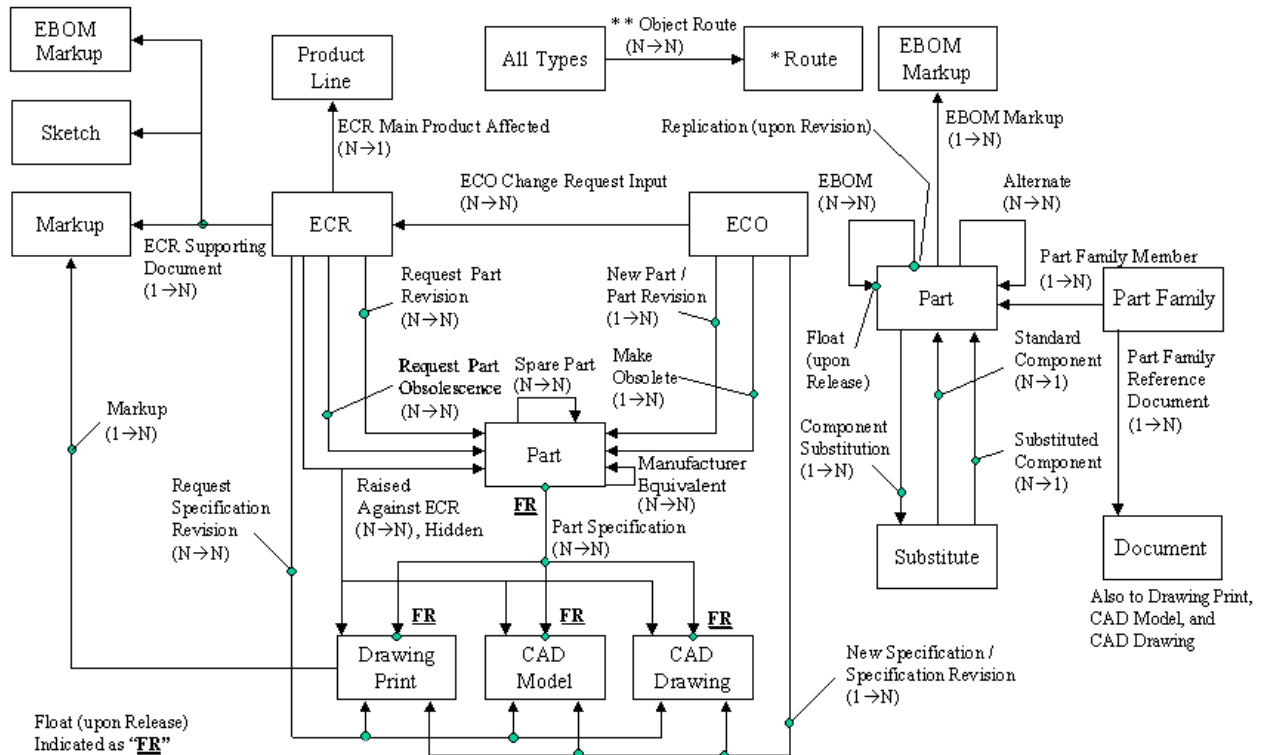
Data Models for Applications

MatrixOne Document Central



Document and Document Classification Type Inheritance

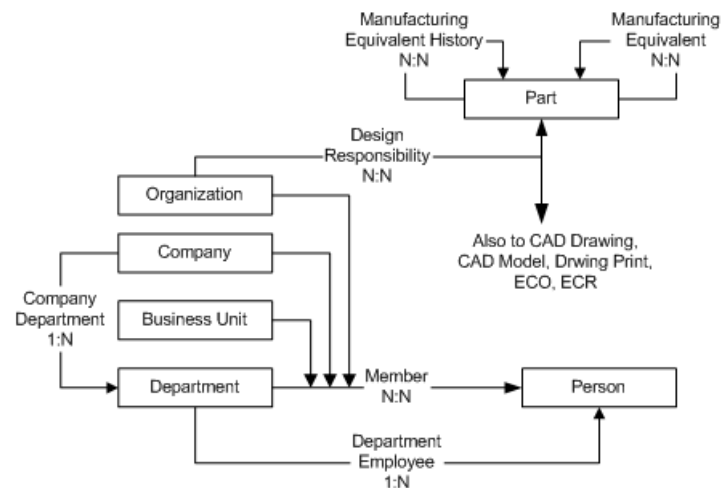




* Items are owned by another application.

**For the complete data model for common routes, see [Common Routes](#).

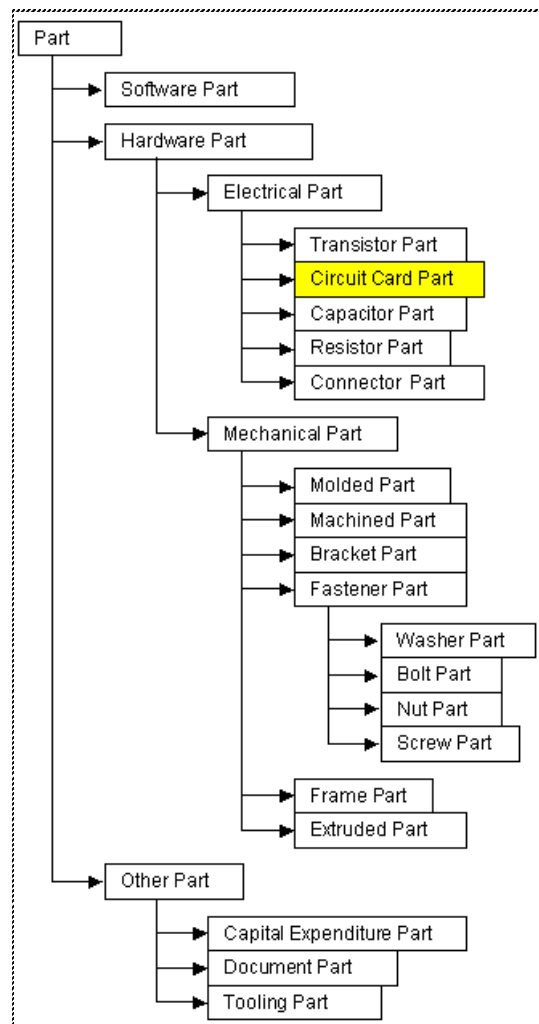
The below drawing shows types and relationships added for Engineering Central 10.0.0.



The type inheritance chart shown below shows the child types of the Part type that are included in the framework. This is only an example of how the part structure can be defined. You'll need to create a structure appropriate for your system.

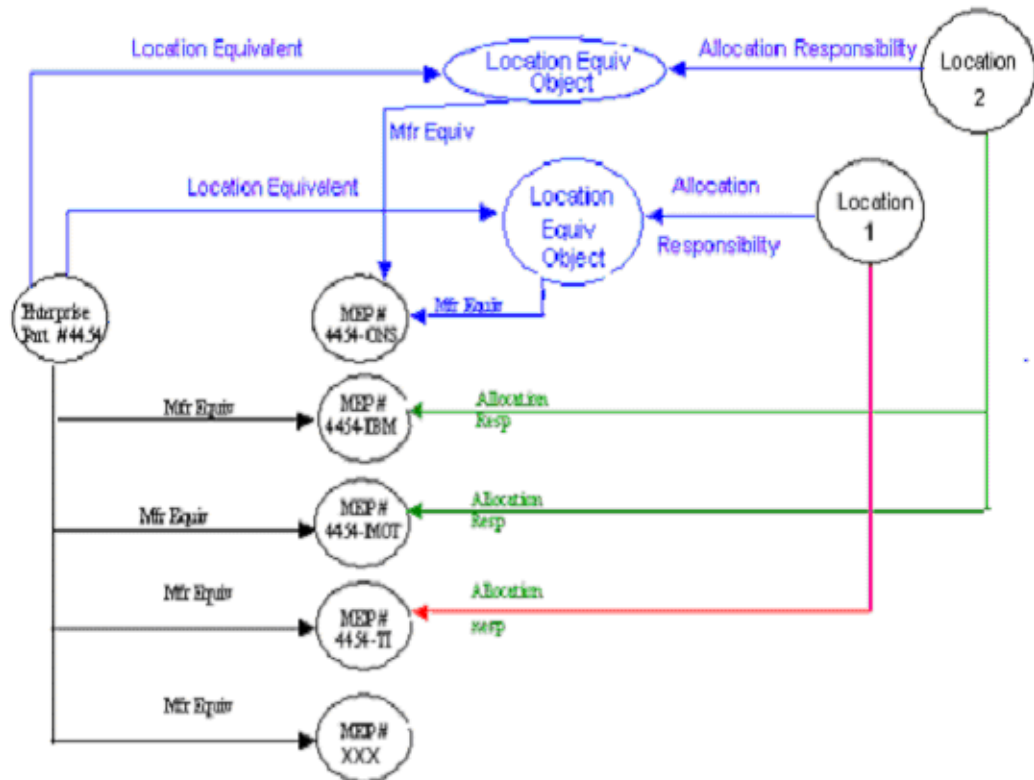
Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

Part Type Inheritance

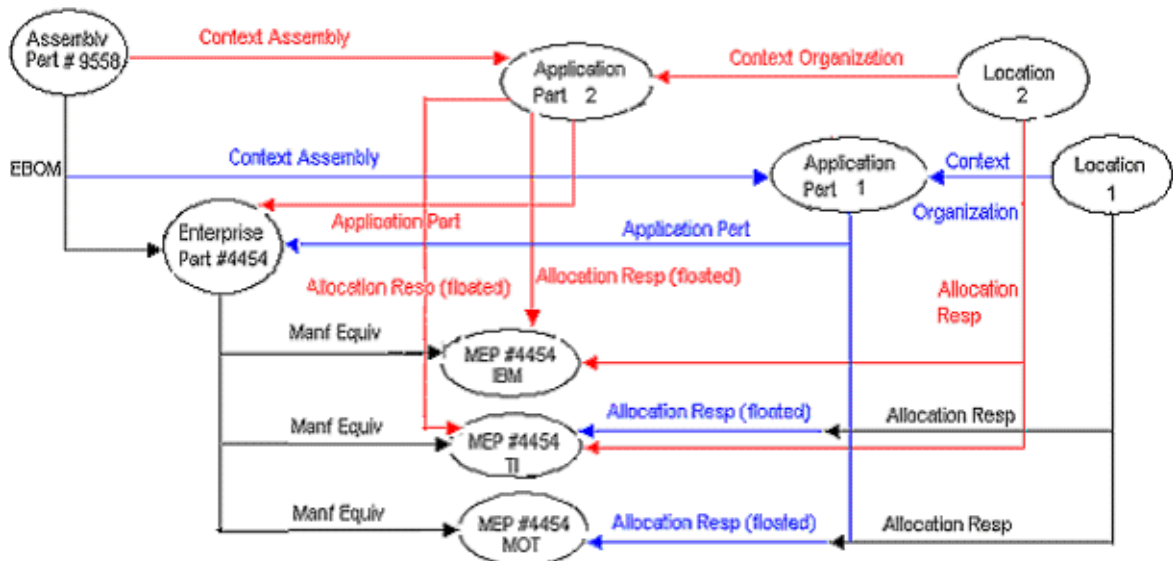


The below drawing shows types and relationships added for Engineering Central 10.5.

Location Context MEP schema



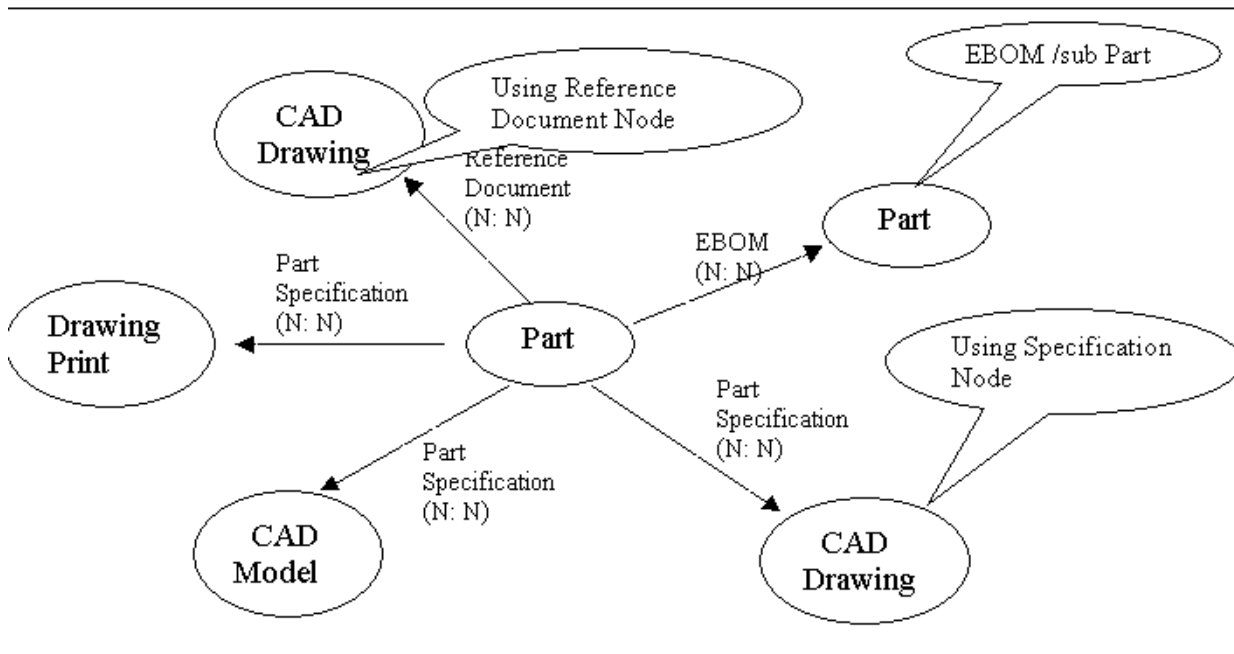
Application Context Schema



- When the application part is created, any existing allocation responsibility relationships that exist from the location to the enterprise part MEPs are copied to the application part.

- In the example above, default allocation responsibility relationships will automatically be created from the application part to MEP #4454-MOT and 4454-TI.
- The intermediate application part has the following relationships:
 - The Application Part object is connected to the assembly (in whose context it was created) with the Context Assembly relationship. This relationship is FROM Assembly TO Application Part with 1:N cardinality.
 - The Application Part object is connected to the Enterprise part it is representing with the Application Part relationship. This relationship is FROM Application Part TO Enterprise Part. The Application Part represents this enterprise part in this assembly context with 1:1 cardinality.
 - The Application Part object is connected to the Location with the Organization Context relationship. This relationship is FROM Location TO Application Part with 1:N cardinality.
 - The Application Part object is connected to enterprise part MEPs with the Allocation Responsibility relationship. The cardinality is restricted to 1:1 with code.
- This application context provides a filter at the assembly level to display only MEPs that have been qualified for use for a specific assembly and location. The application context provides granular control of MEP usage to the assembly and location level.
- When the Application Part reaches the Complete state, any MEP connect or disconnect actions to the Application Part will cause it to revise. The context assembly, Application Part and Context organization relationships will float to the new Application Part revision. New MEP connections will point to the new Application Part revision, existing MEP connections will float to the new Application Part revision. Disconnected MEPs will point to the previous Application Part revision.

Part Document Summary and Associated Objects



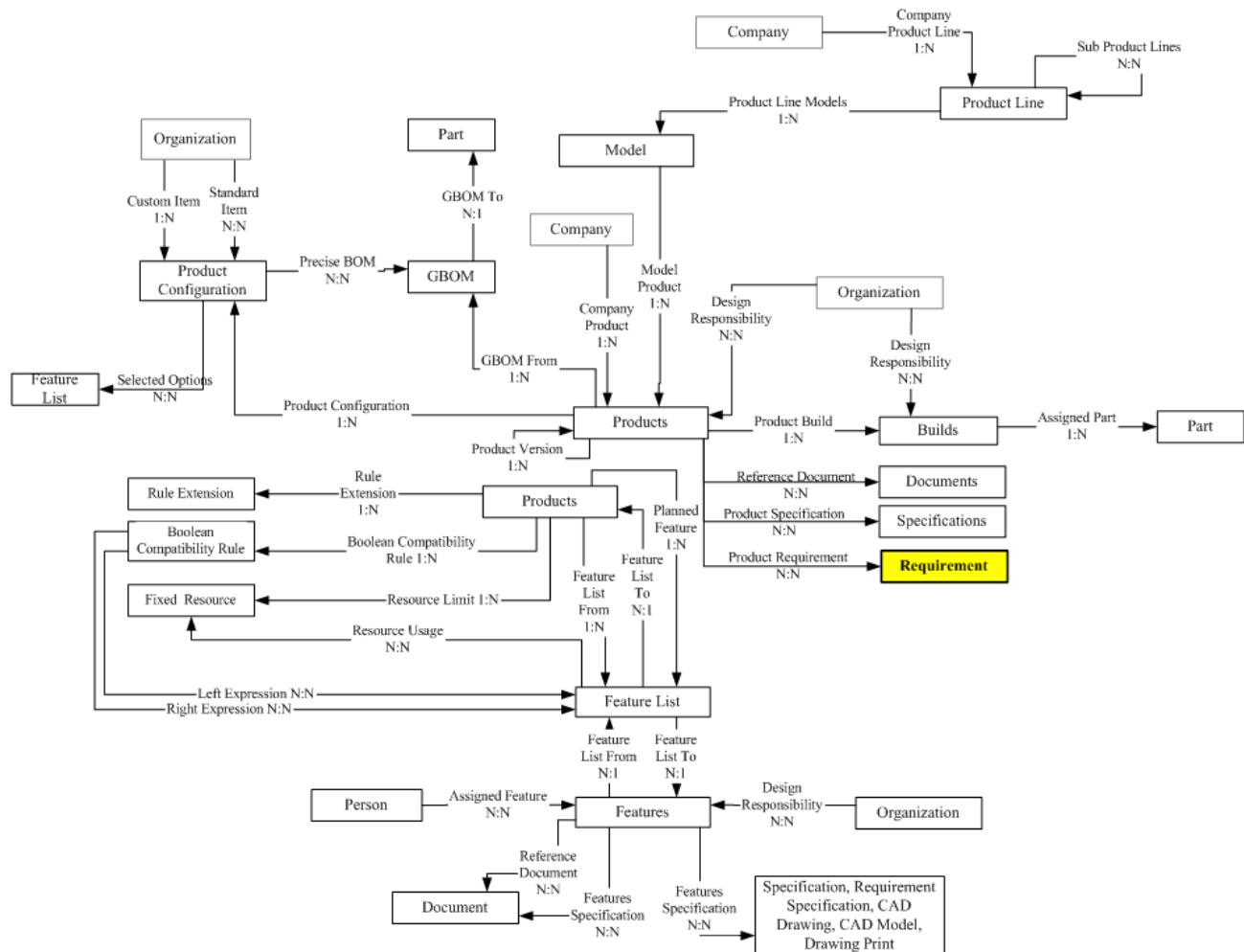
MatrixOne Product Central

The data models for Product Central are divided into these feature categories:

- *Products, Product Line, Models, Features, Product Configuration*
- *Requirement Management and Program Management*
- *Incidents*
- *Images*
- *Builds*

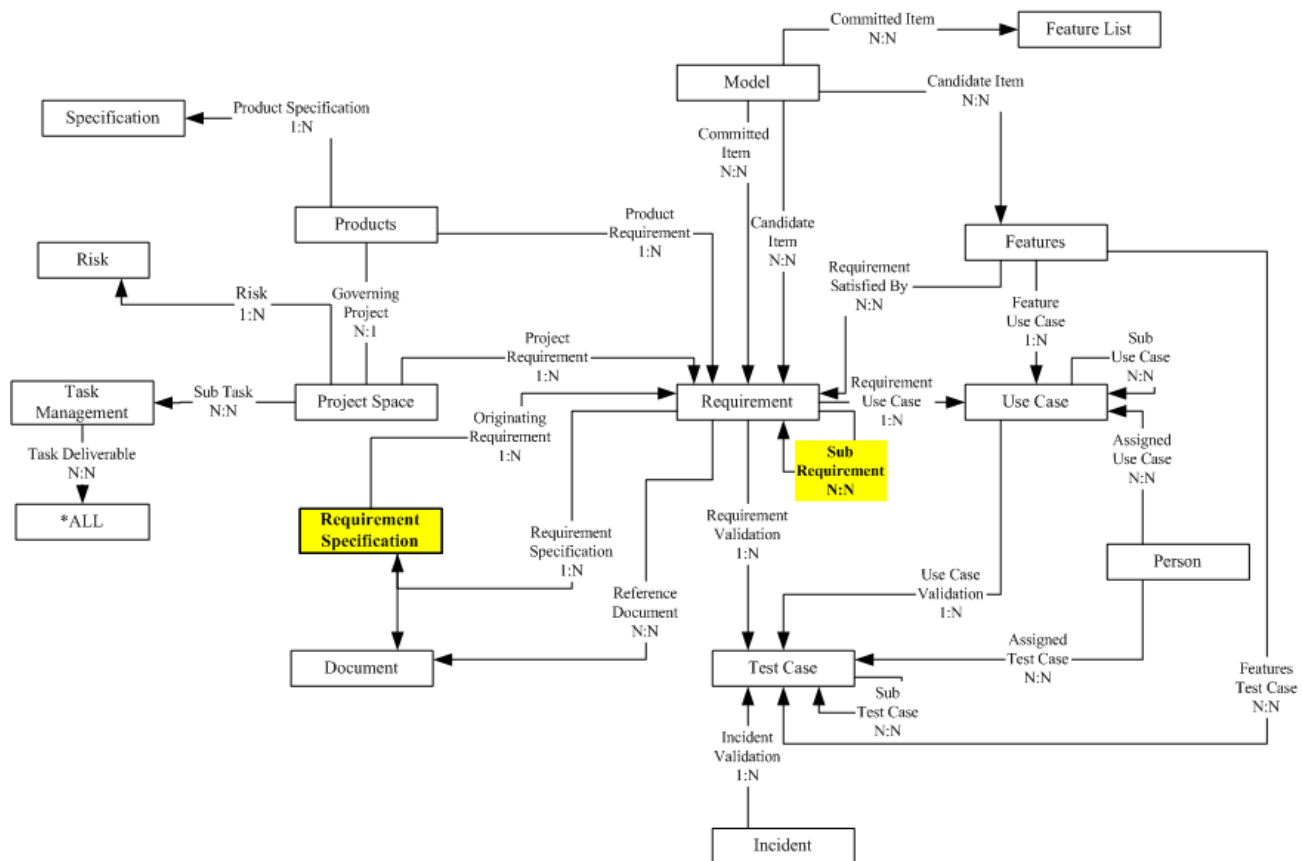
Also see *Common Component Data Models*.

Products, Product Line, Models, Features, Product Configuration



Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see *Renamed Administrative Objects* in Chapter 3.

Requirement Management and Program Management

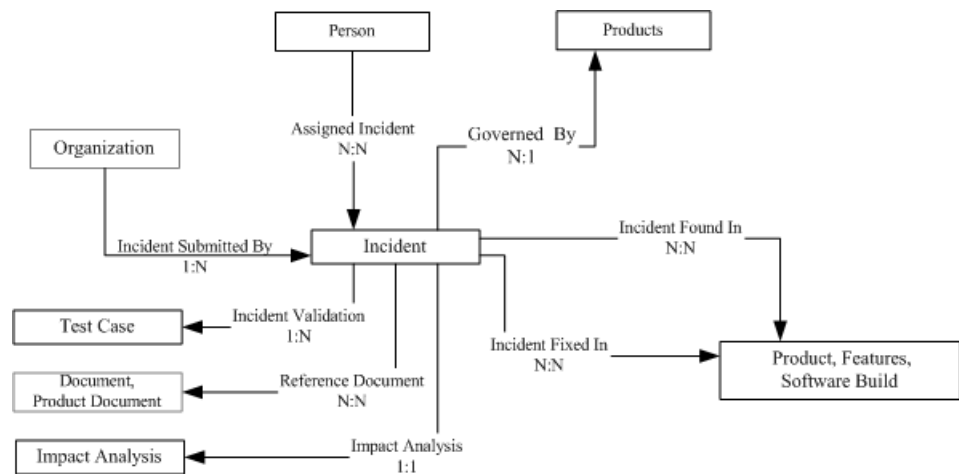


*All == Features, Products, Builds, Requirement, Use Case, Test Case, etc.

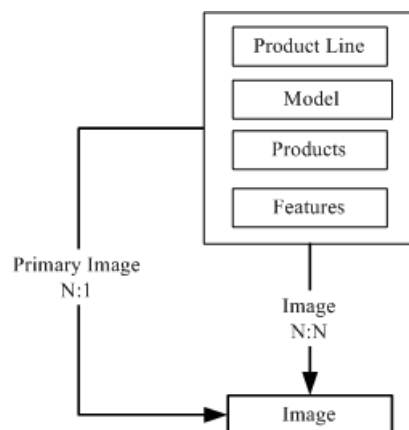
All the types managed in Product Central will be added to the TO side of relationship Task Deliverables.

Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

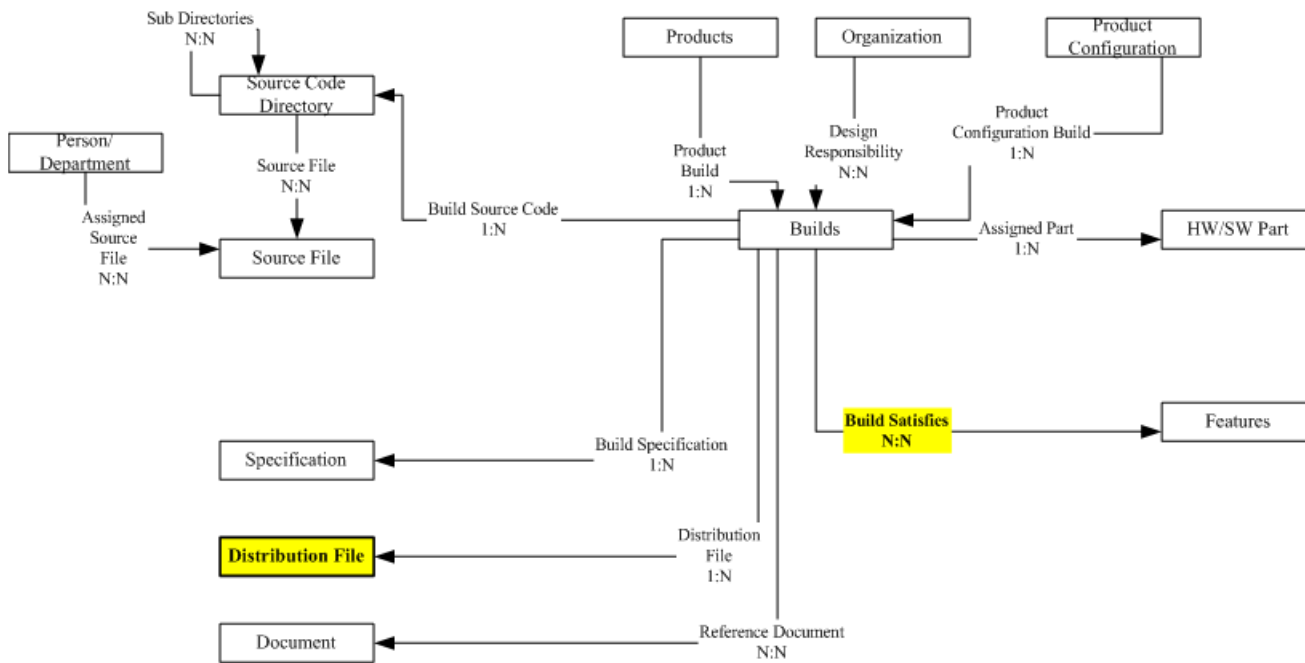
Incidents



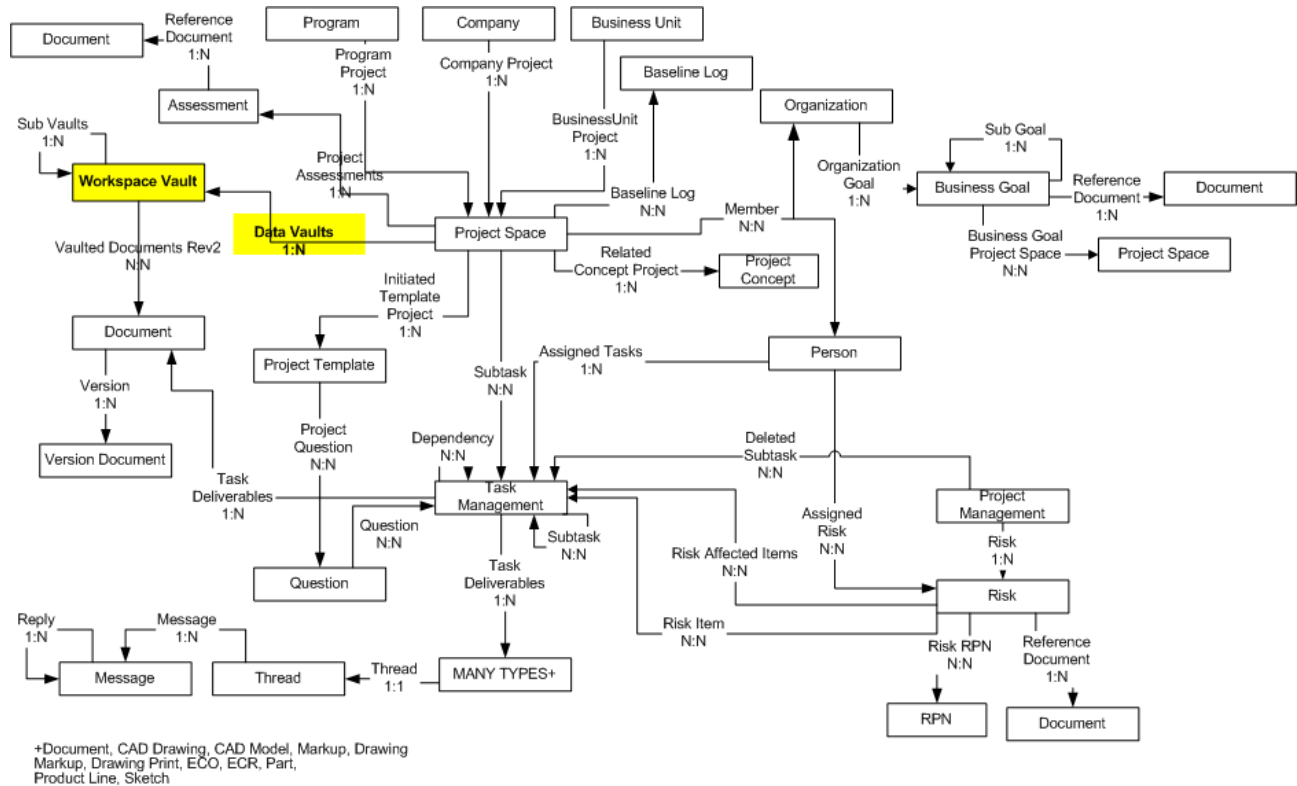
Images



Buils

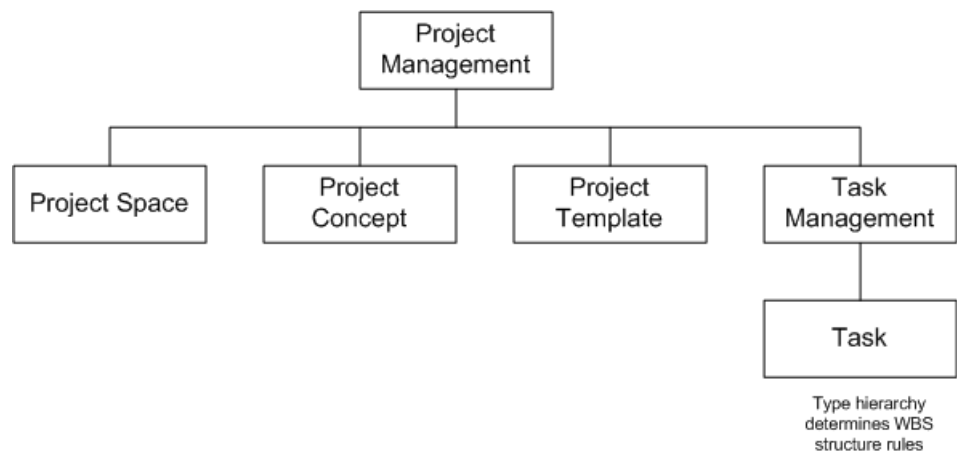


Program Central Main Data Model

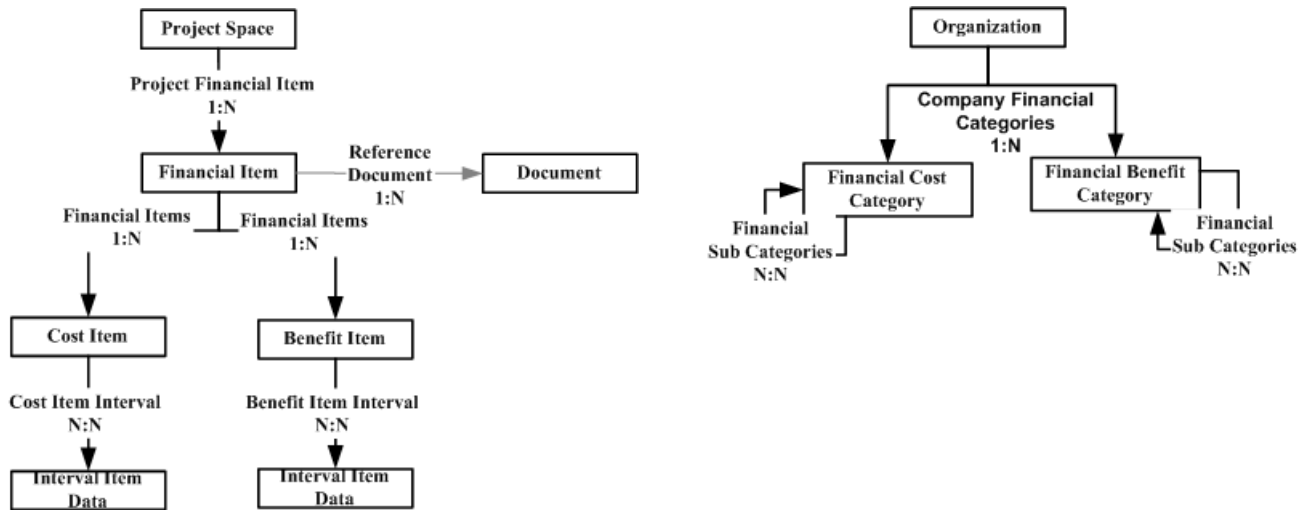


Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

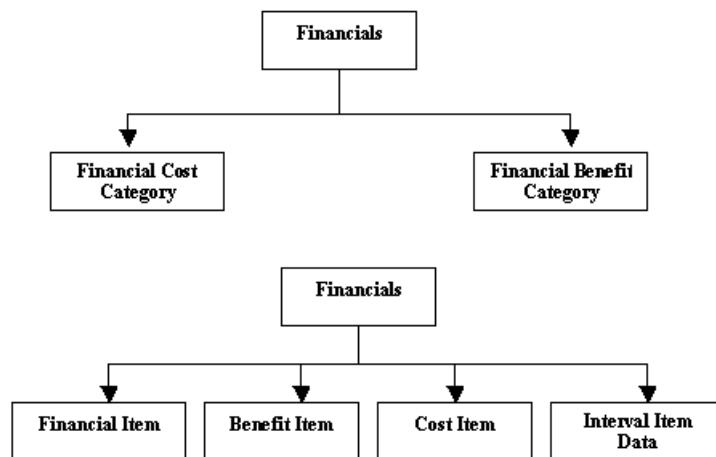
This graphic shows the type hierarchy for Project Management types.



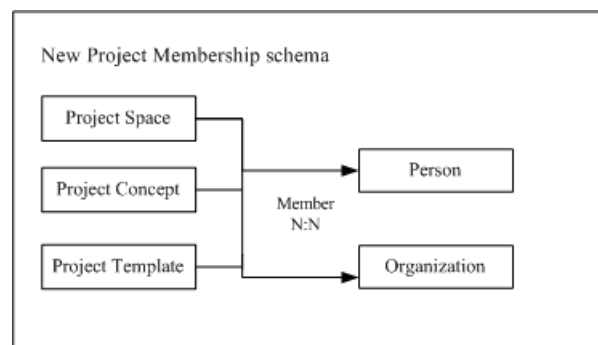
Program Central Data Model Continued: Financial Items and Categories



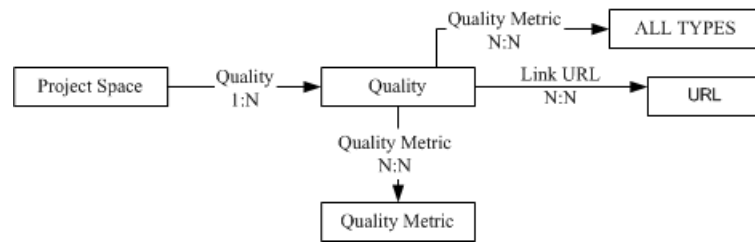
This is the type hierarchy for financial types.



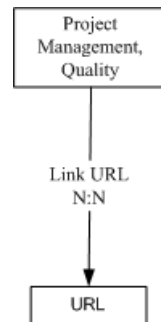
Program Central Data Model Continued: Project Membership



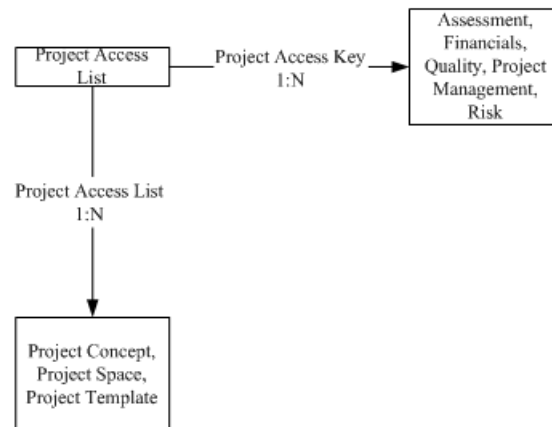
Program Central Data Model Continued: Quality



Program Central Data Model Continued: URL



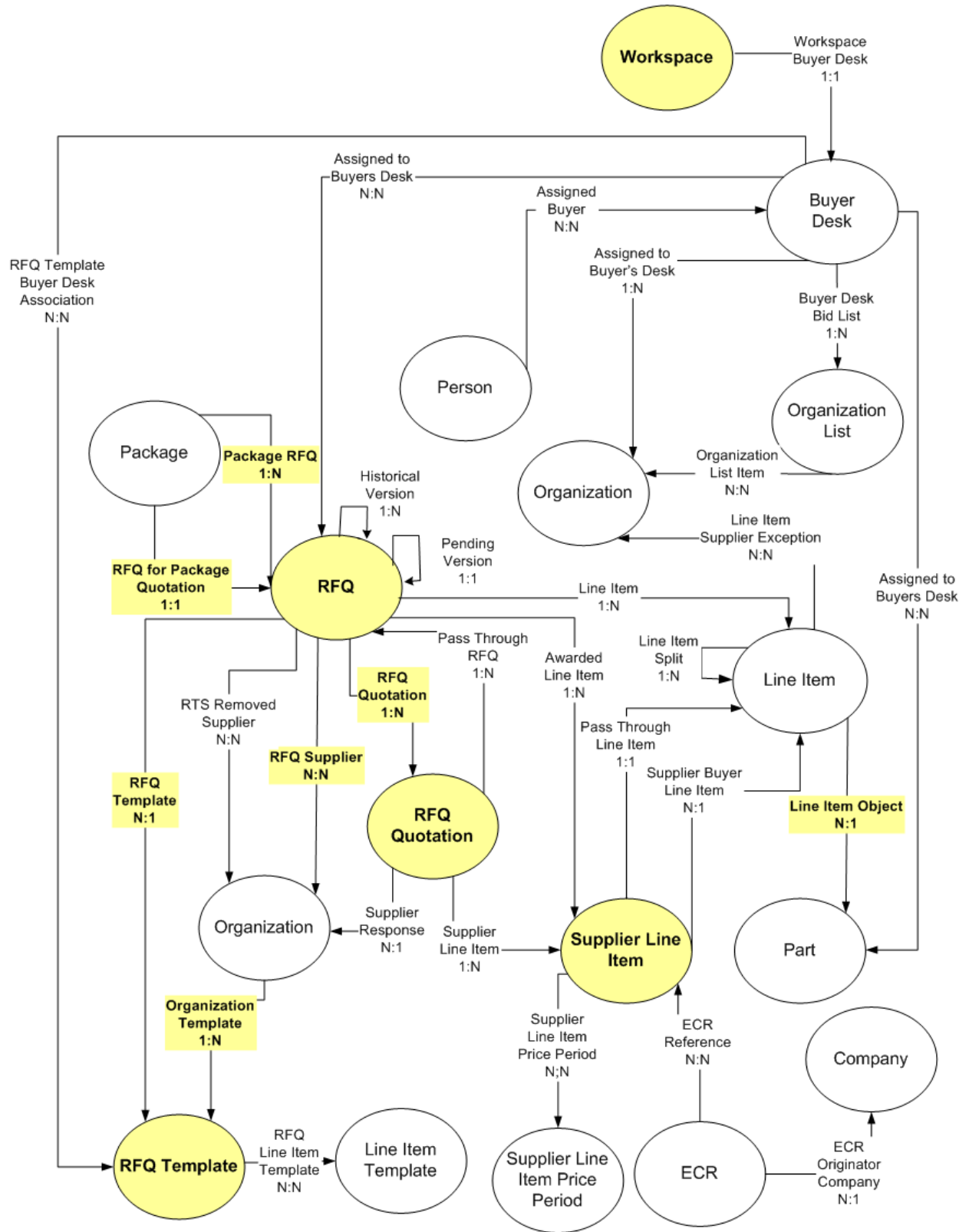
Program Central Data Model Continued: Project Access List



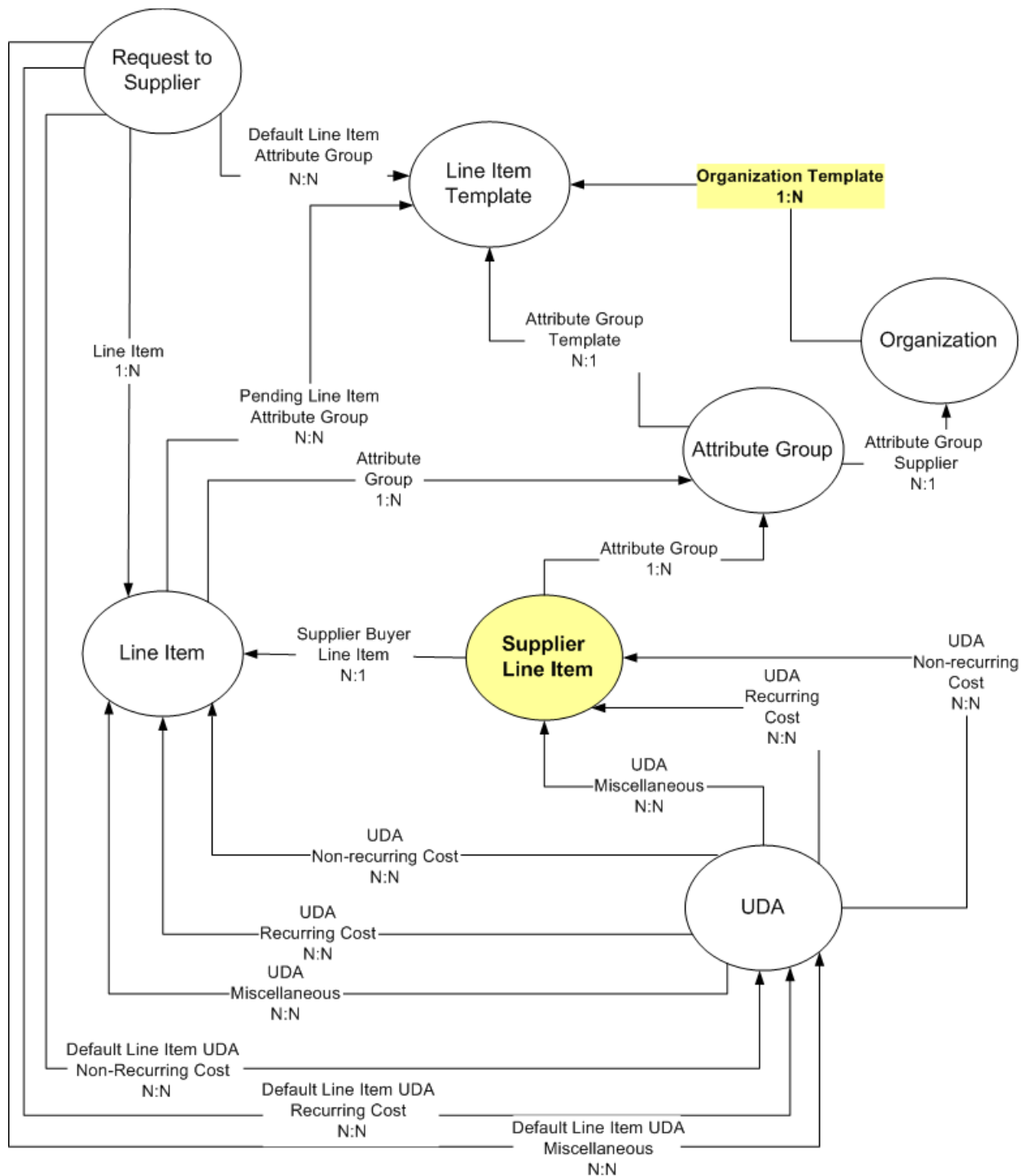
Below are the data models for the Sourcing Central application (formerly called Supplier Central – Sourcing). The data model for the profile management features are the same as the one for the MatrixOne Supplier Central application, [Supplier Central Data Model Continued: Profile Management](#).

Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

Sourcing Central Main Data Model: RFQ, Quotations, and Line Items

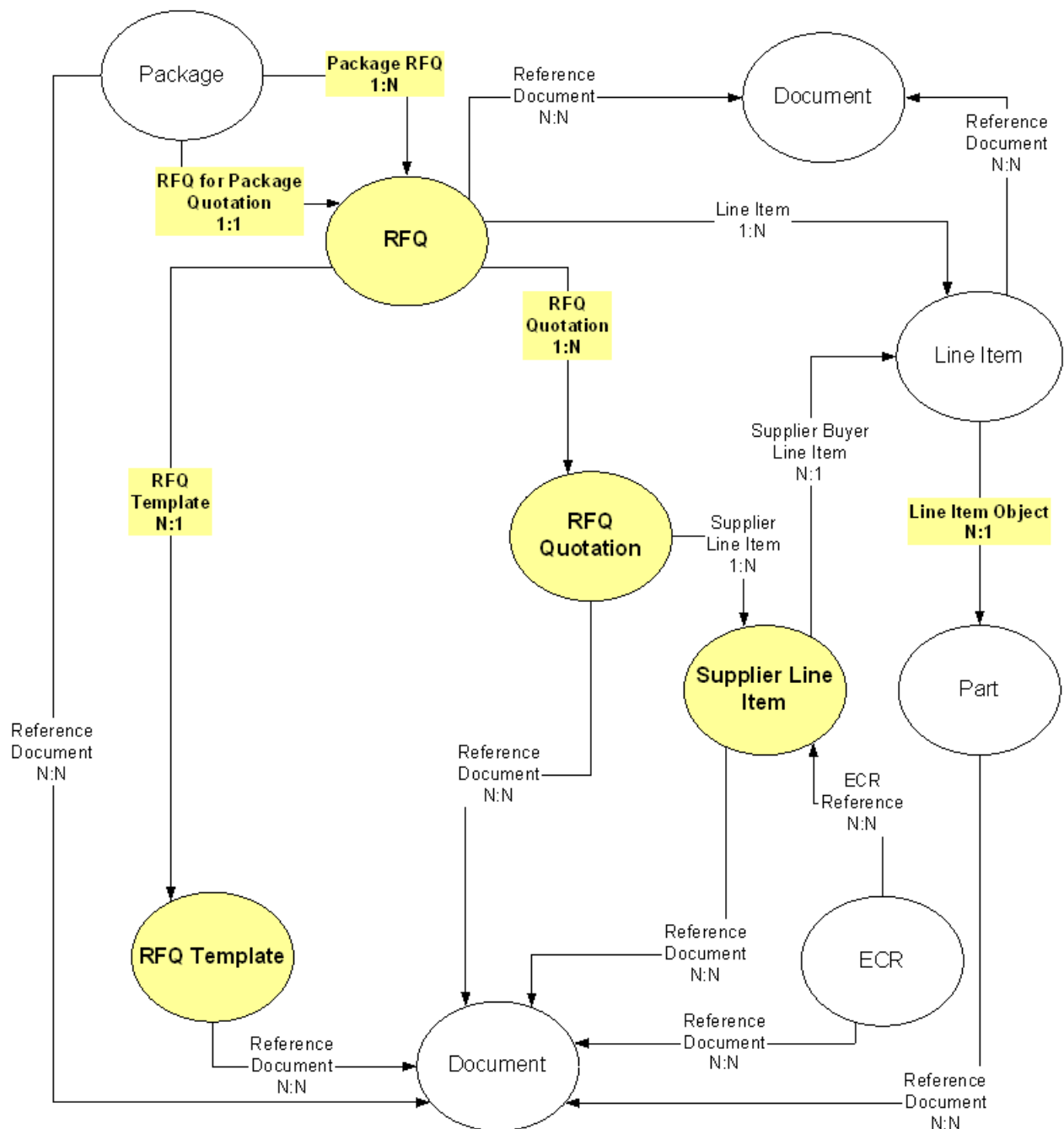


Sourcing Central Data Model Continued: Line Item Extensions



Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

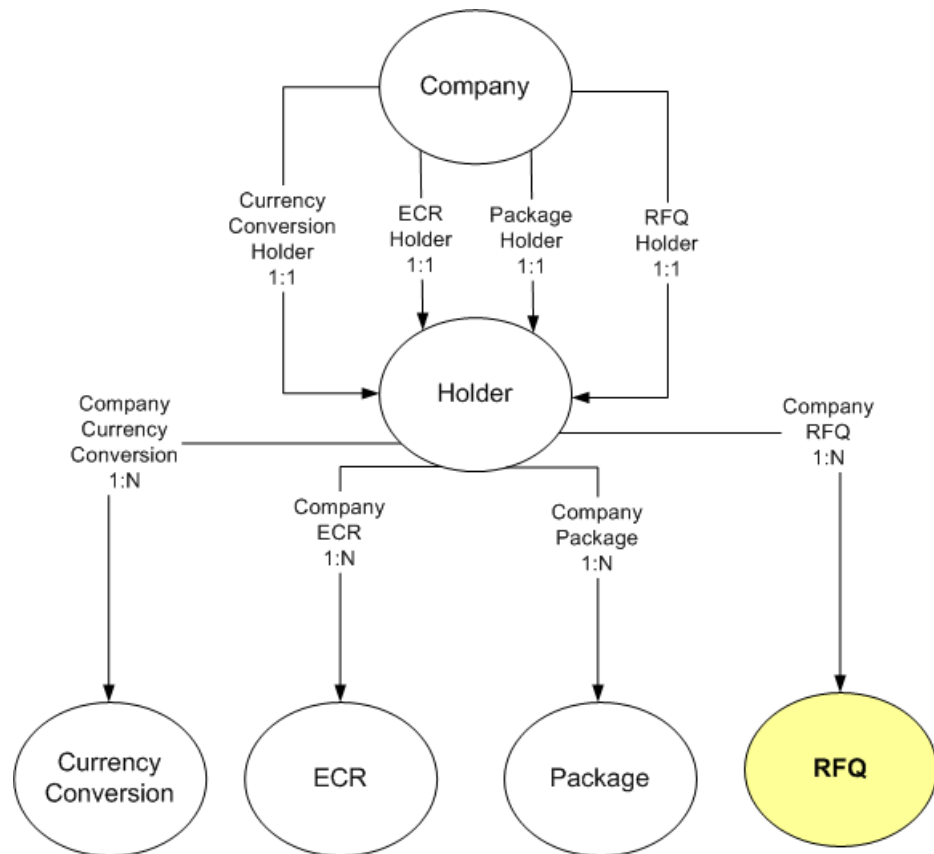
Sourcing Central Data Model Continued: Attached Documents



Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see *Renamed Administrative Objects* in Chapter 3.

Sourcing Central Data Model Continued: Searching with Vaulting Strategy

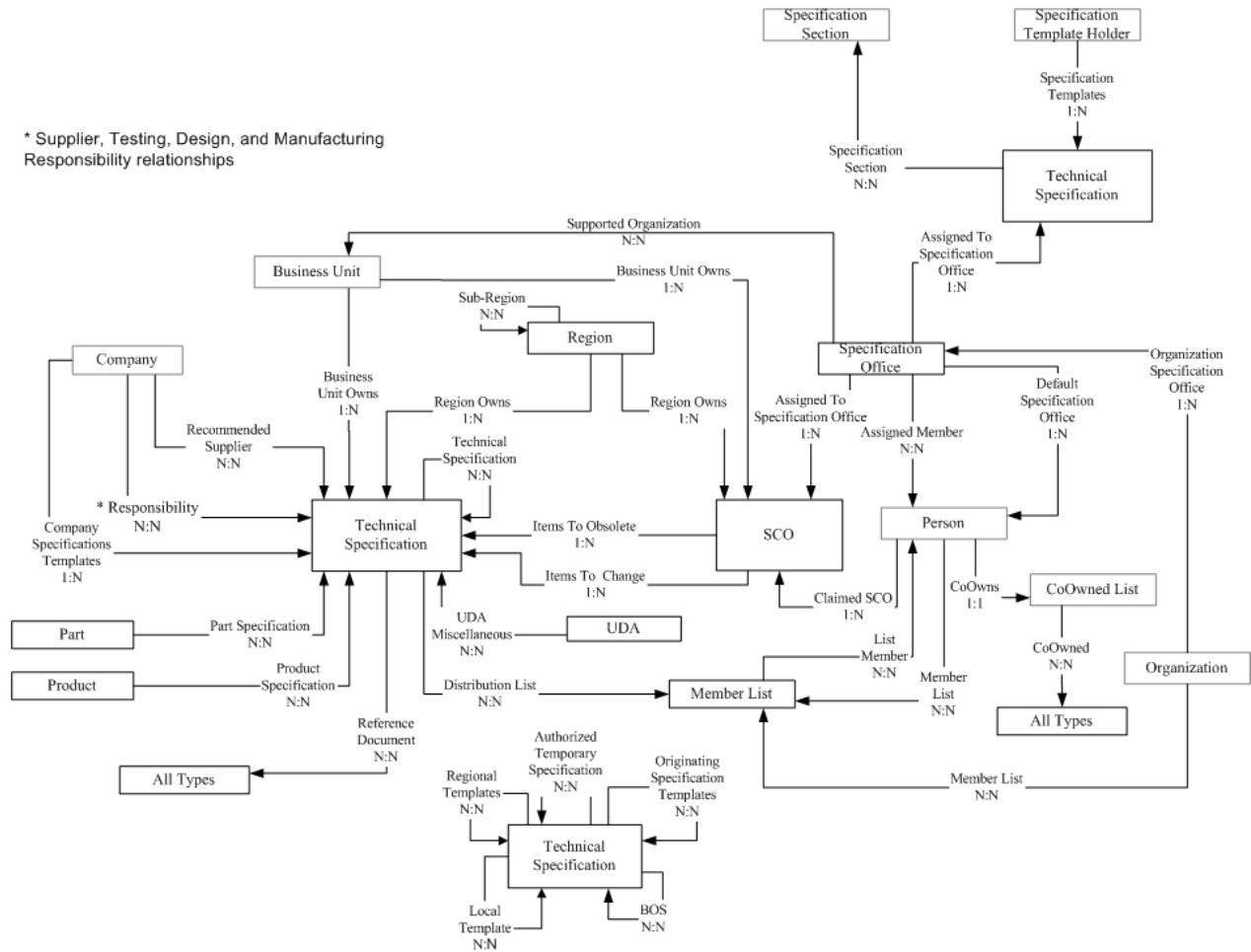
This data model shows schema added to make object searches more efficient and accurate now that a company can use multiple vaults and multiple companies can use a single vault.



Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

This is the primary data model for Specification Central. Also see [Common Component Data Models](#).

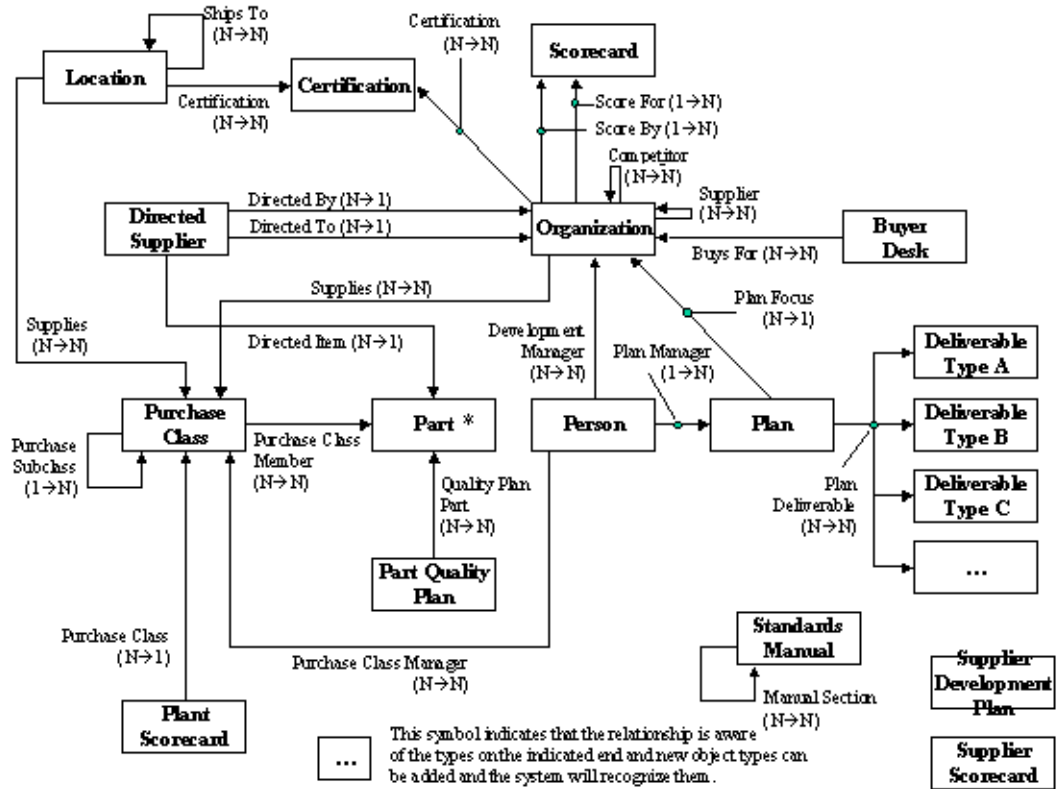
* Supplier, Testing, Design, and Manufacturing Responsibility relationships



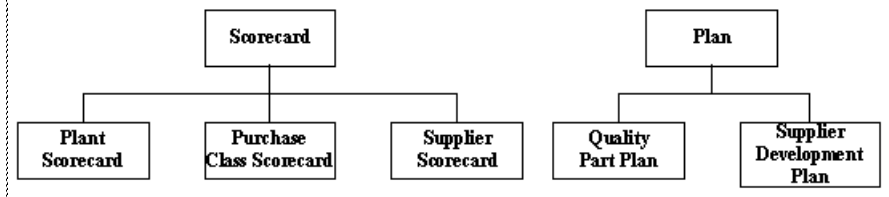
Below are the data models for the MatrixOne Supplier Central application.

Supplier Central Main Data Model: Quality Plans and Directed Suppliers

This data model is used in the quality plans and directed supplier features in MatrixOne Supplier Central (formerly called Supplier Central - Quality).



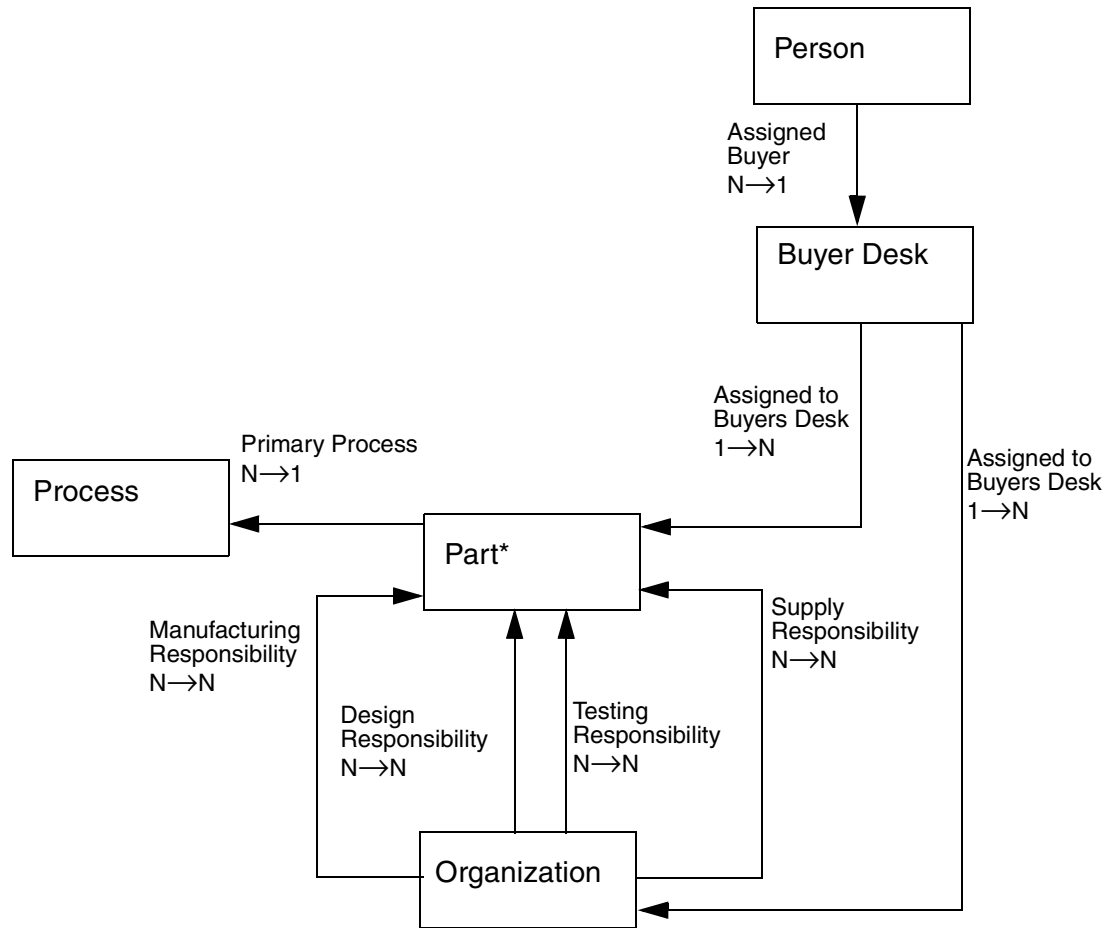
Type Inheritance



* Part objects are owned by Engineering Central.

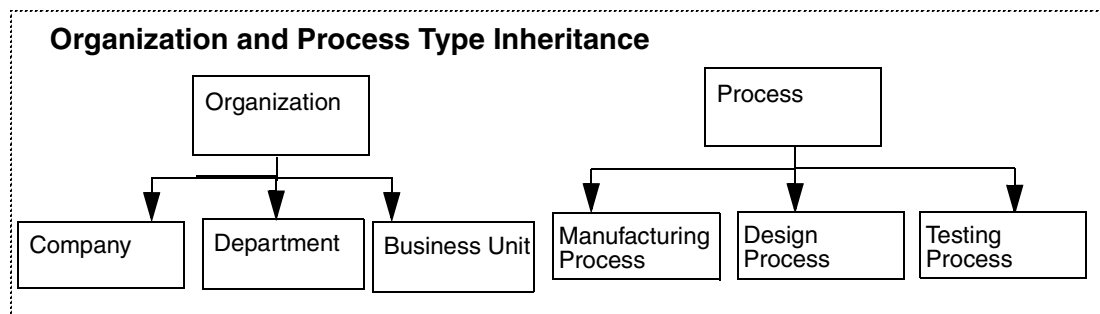
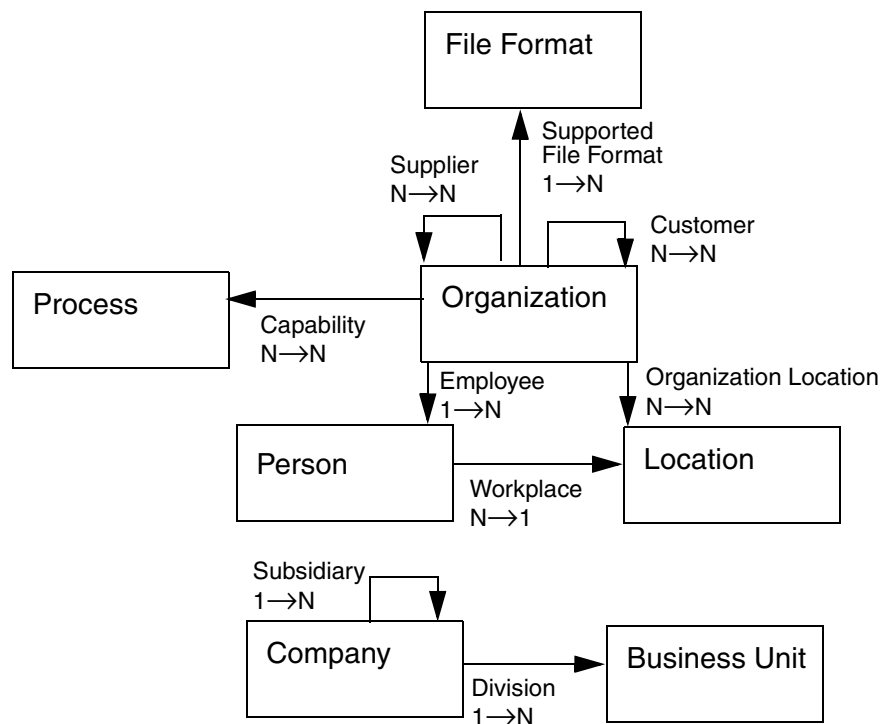
Supplier Central Data Model Continued: Buyer Desk and Access Management

The MatrixOne Supplier Central application no longer includes the bidding and manufacturing assessment functions, although it does include Buyer Desk features. All bidding functions are now in MatrixOne Sourcing Central.

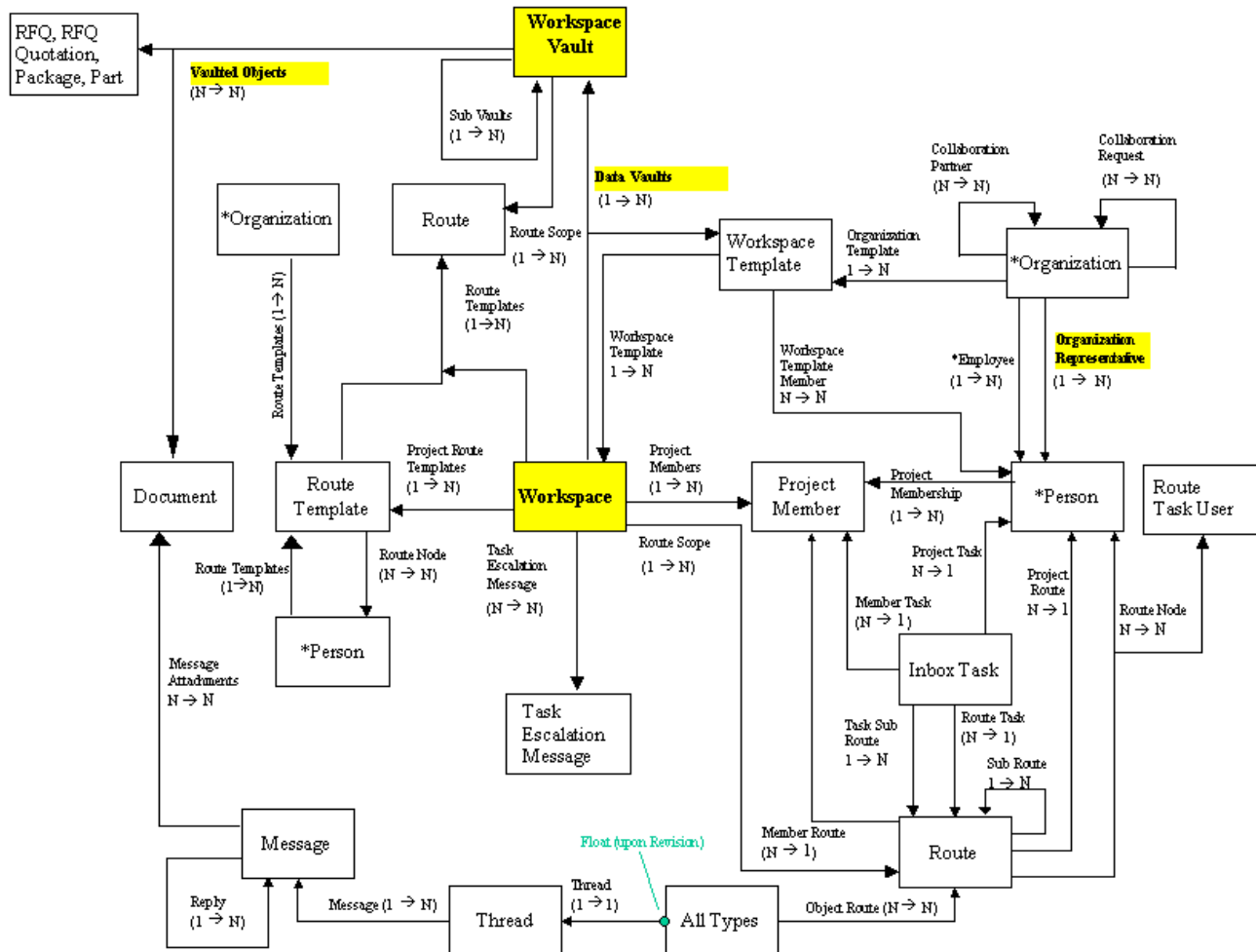


* Part objects are owned by Engineering Central.

Supplier Central Data Model Continued: Profile Management



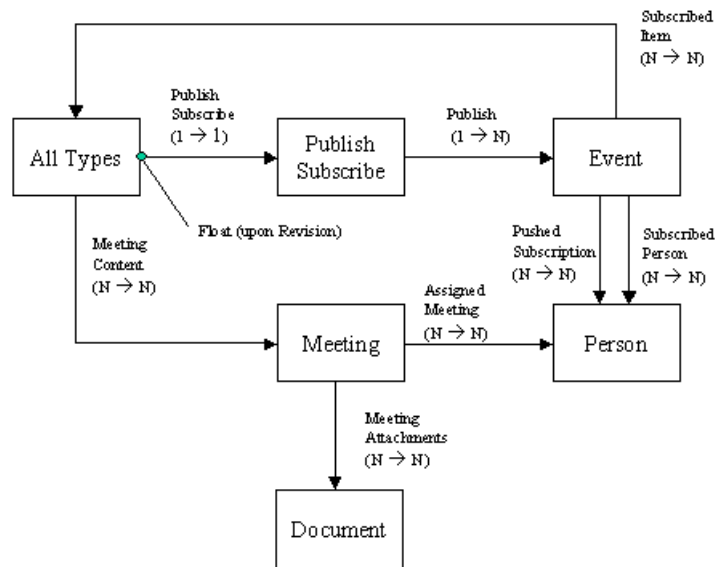
Team Central Main Data Model



* These objects are owned by another application.

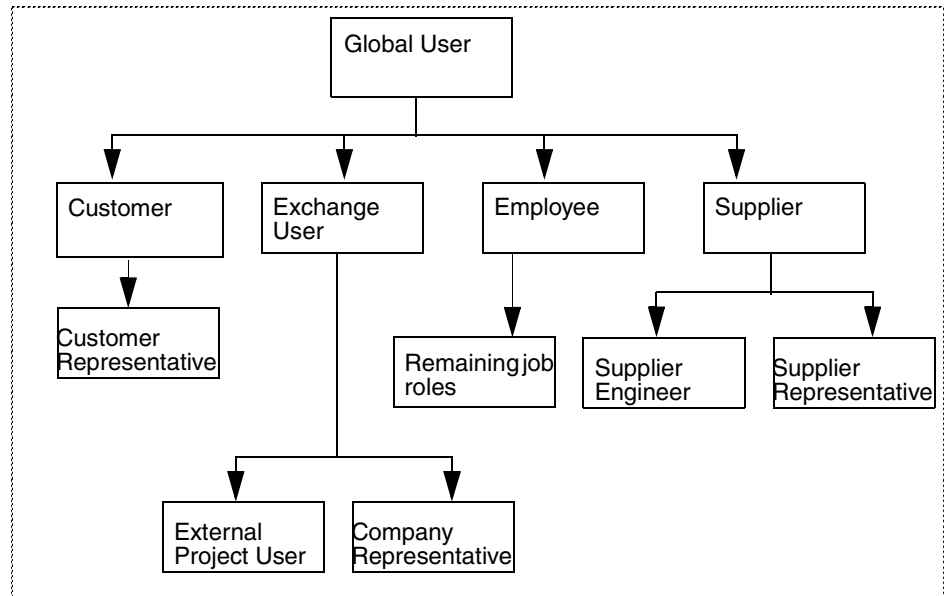
Objects highlighted in yellow and bold text have been renamed but their symbolic names continue to reflect the original name. For a list of the symbolic names, see [Renamed Administrative Objects](#) in Chapter 3.

Team Central Data Model Continued: Publish/Subscribe and Meetings



Role Type Inheritance

This chart illustrates the type inheritance for roles. The standard job-related roles that are not children of Customer or Supplier are children of Employee. To see all child roles of Employee, in Business Modeler, expand the Employee role in the indented browser. The roles not included in this type hierarchy are system administrative roles such as System Conversion Manager, System Transition Manager, and Access Grantor.



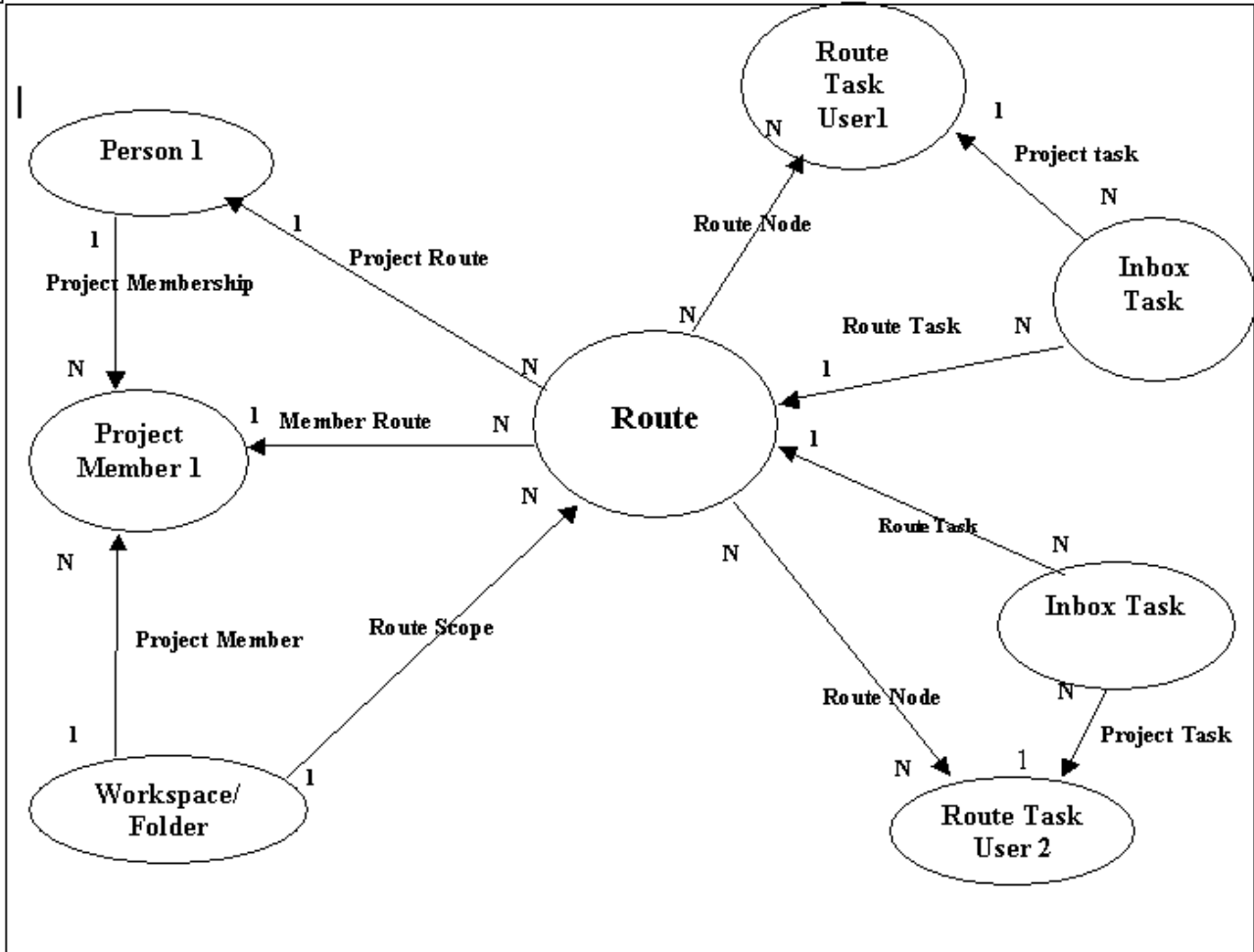
For a description of roles, see [Roles](#) in Chapter 3.

**Common
Component Data
Models**

This section presents data models for features included in Common Components, which is a set of features used by many MatrixOne applications, such as routes and profile management.

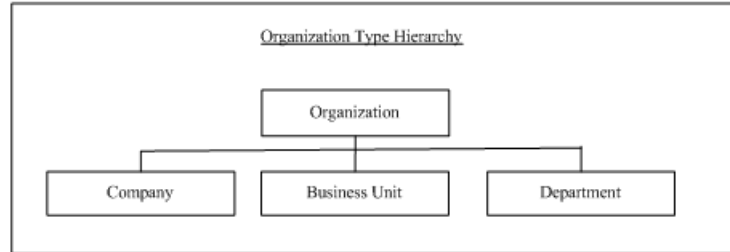
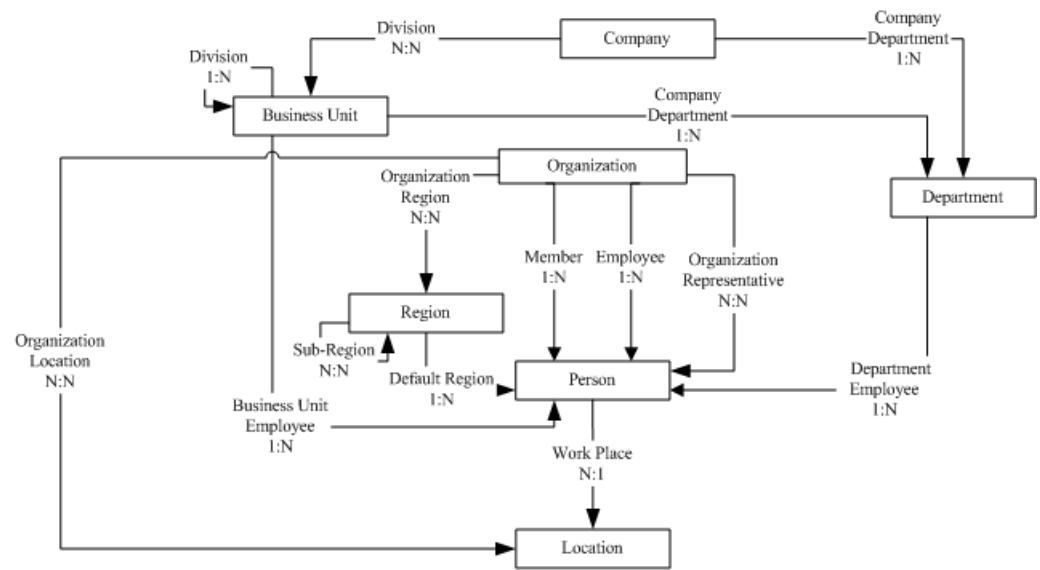
Common Routes

This data model reflects the common component routes.



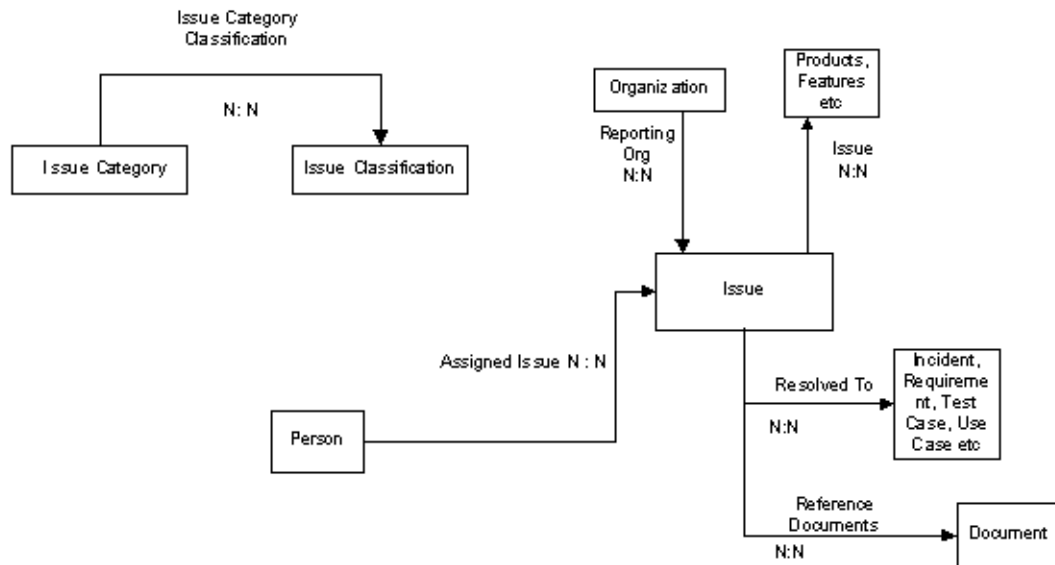
Common Profile Management

This data model reflects the common component profile management.



Common Issue Management

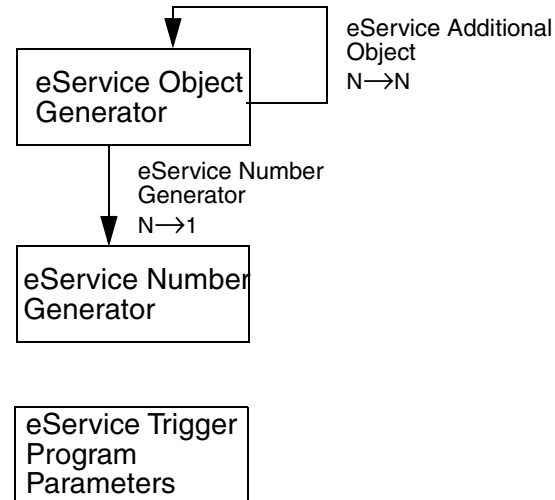
This data model reflects the common component issue management.



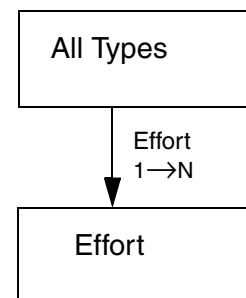
This section shows the object types and relationships that are used by many MatrixOne applications.

Administrative Business Objects

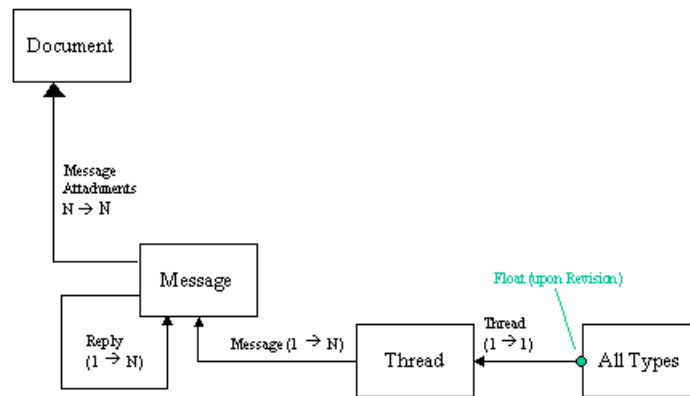
The types and relationships whose names are prefixed with “eService” accomplish system-related tasks and are not related to a business-process.



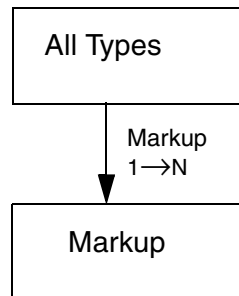
Efforts



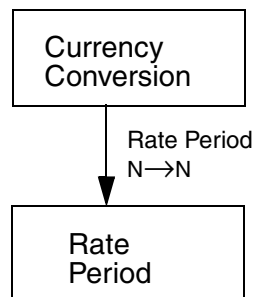
Messages



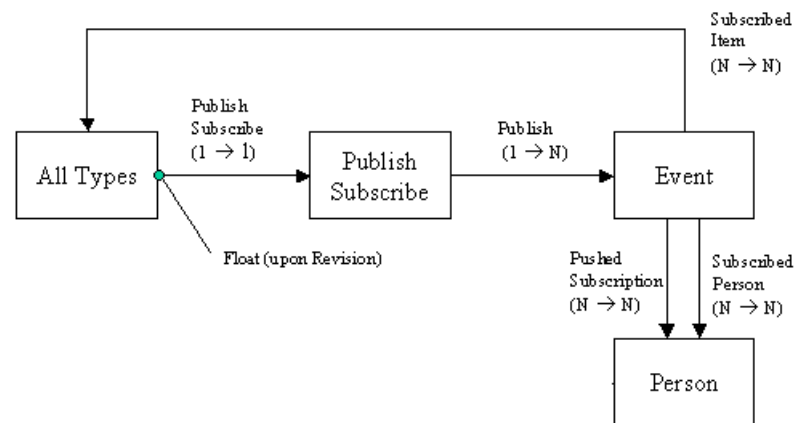
2D/3D Viewers



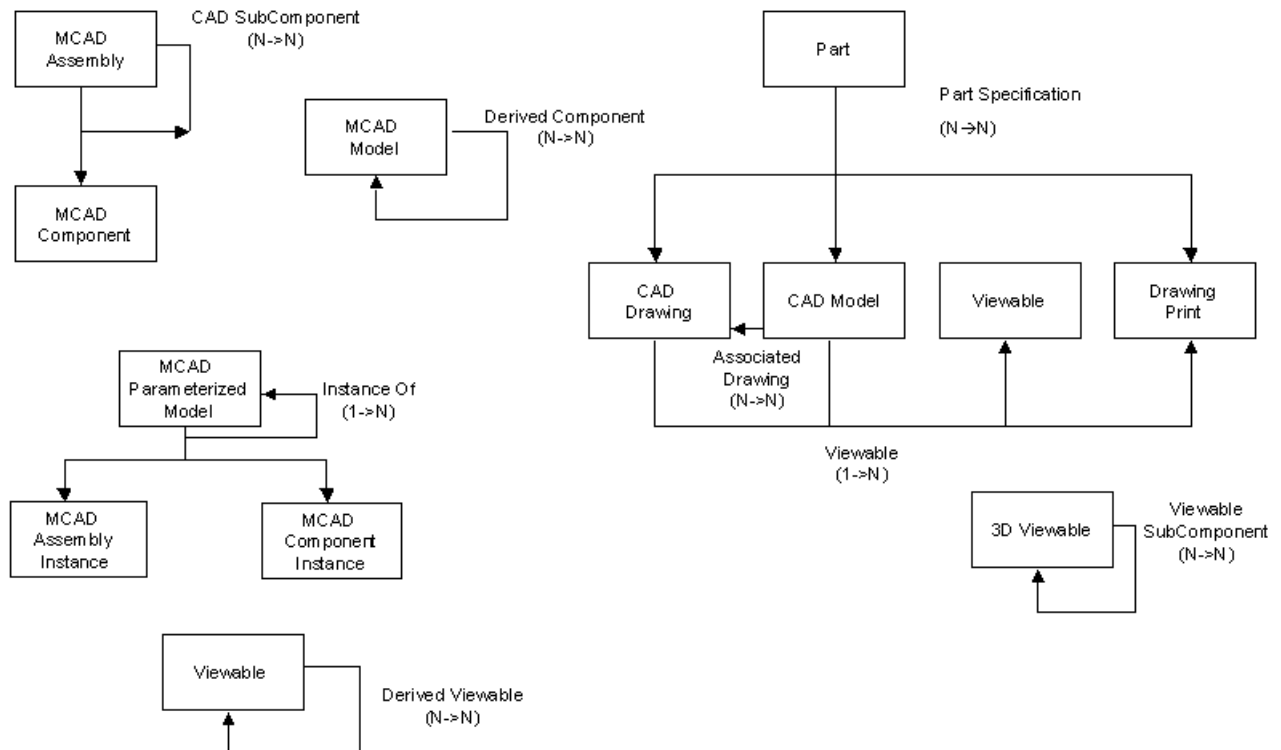
Currency Conversion



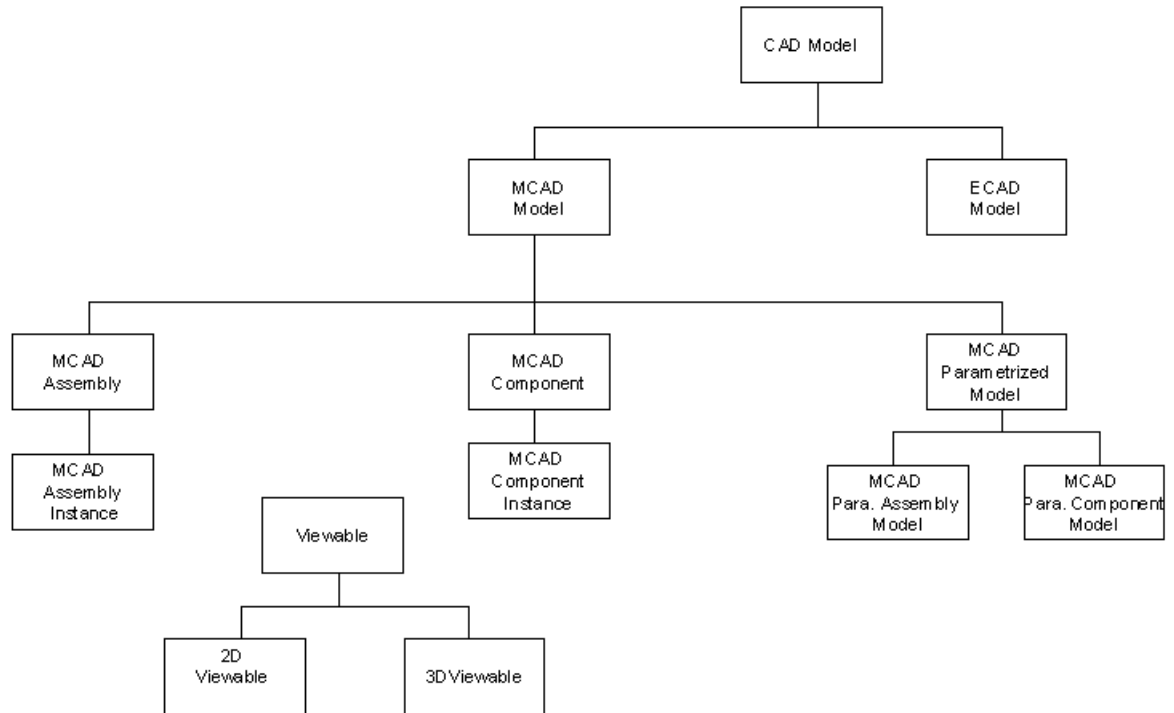
Publish Subscribe



MCAD Integration



This is the type inheritance chart for MCAD integration types.



Schema Definitions

Renamed Administrative Objects

This section lists schema objects that have been renamed. When an object is renamed, its symbolic name does not change and will continue to reflect the original name. So when calling these objects in a program, make sure you use the original symbolic name for the object.

Admin Type	Old Name	New Name	Symbolic Name	Renamed in AEF Version
Attribute	Classification	Part Classification	attribute_Classification	9500
	Co-Owner	Co-Owners	attribute_CoOwner	9500

Admin Type	Old Name	New Name	Symbolic Name	Renamed in AEF Version
Policy	Project	Workspace	policy_Project	9500
	Project Vault	Workspace Vault	policy_ProjectVault	9500
	RTS Supplier Part	Supplier Line Item	policy_RTSSupplierPart	9500
	Request To Supplier	RFQ	policy_RequestToSupplier	10.0.0.0
	RTS Cancelled	RFQ Cancelled	policy_RTSCancelled	10.0.0.0
	RTS History Version	RFQ History Version	policy_RTSHistoryVersion	10.0.0.0
	RTS Pending Version	RFQ Pending Version	policy_RTSPendingVersion	10.0.0.0
	RTS Quotation	RFQ Quotation	policy_RTSSquotation	10.0.0.0
	RTS Template	RFQ Template	policy_RTSTemplate	10.0.0.0
	Software Use Case	Use Case	policy_SoftwareUseCase	10.0.1.0
Relationship	Available On	Build Satisfies	relationship_AvailableOn	10.0.1.0
	Company Representative	Organization Representative	relationship_CompanyRepresentative	9510
	Organization RTS Template	Organization Template	relationship_OrganizationRTSTemplate	9500
	Vaulted Documents	Vaulted Objects	relationship_VaultedDocuments	9500
	Package RTS	Package RFQ	relationship_PackageRTS	10.0.0.0
	Project Vaults	Data Vaults Workspace Vaults (9500)	relationship_ProjectVaults	10.0.0.0 9500
	Requirement Breakdown	Sub Requirement	relationship_RequirementBreakdown	10.0.1.0
	RTS for Package Quotation	RFQ for Package Quotation	relationship_RTSSforPackageQuotation	10.0.0.0
	RTS Part	Line Item Object	relationship_RTSPart	9500
	RTS Quotation	RFQ Quotation	relationship_RTSSquotation	10.0.0.0
	RTS Supplier	RFQ Supplier	relationship_RTSSupplier	10.0.0.0
	RTS Template	RFQ Template	relationship_RTSTemplate	10.0.0.0
State (on RFQ policy)	Initial Package Review	Initial Review	state_InitialPackageReview	10.0.0.0
	Package Sent	Sent	state_PackageSent	10.0.0.0
	Final Package Review	Final Review	state_FinalPackageReview	10.0.0.0

Admin Type	Old Name	New Name	Symbolic Name	Renamed in AEF Version
Type	Configurable Feature	Feature	type_ConfigurableFeature	10.0.1.0
	Circuit Card Part	Circuit Card Part	type_CircuitCardPart <i>The symbolic name for this type was also renamed to correct the misspelling. If you refer to this type in any programs, you must change the symbolic name to reflect the corrected, updated name.</i>	9512
	Project	Workspace	type_Project	9500
	Project Vault	Workspace Vault	type_ProjectVault	9500
	Request To Supplier	RFQ	type_RequestToSupplier	10.0.0.0
	RTS Quotation	RFQ Quotation	type_RTSTQuotation	10.0.0.0
	RTS Template	RFQ Template	type_RTSTemplate	10.0.0.0
	RTS Supplier Part	Supplier Line Item	type_RTSSupplierPart	9500
	Software Distribution	Distribution File	type_SoftwareDistribution	10.0.1.0
	Software Requirement Specification	Requirement Specification	type_SoftwareRequirementSpecification	10.0.1.0

Deprecated Schema

This section lists schema objects that are not longer being used by the applications and that will be removed in an upcoming version of the framework. When the framework version that deprecates an administrative object is installed, the installation makes the necessary conversions to the business objects and relationships that instantiate the deprecated schema and then deletes the schema.

Admin Type	Name	Deprecated in AEF Version
Attribute	Precise BOM	10.0.1.0
	ECR Originator Organization	10.0.1.0
Policy	Container	10.0.0.0
	Configurable Product	10.0.1.0
	Controlled Design Release	10.0.0.0
	Controlled Production Release	10.0.0.0
	Personal Folder	10.0.0.0
	Project Folder	9500
	Public Folder	10.0.0.0
	Software User Requirement	10.0.1.0

Admin Type	Name	Deprecated in AEF Version
Relationship	Contains	10.0.0.0
	Folder Document	9500
	Member Folder	9500
	Member Folder Access	9500
	Project Folders	9500
	Affected Code	10.0.1.0
	Applied Against	10.0.1.0
	Bug Impact Analysis	10.0.1.0
	Candidate Incident	10.0.1.0
	Candidate Request	10.0.1.0
	Deliverables	10.0.1.0
	Duplicate	10.0.1.0
	Fixed In	10.0.1.0
	Found In	10.0.1.0
	Incident Document	10.0.1.0
	Incident Owned By	10.0.1.0
	Request Impact Analysis	10.0.1.0
	Software Requirement Specification	10.0.1.0
Type	ADHOC CATEGORY	10.0.0.0
	Personal Folder	10.0.0.0
	Project Folder	9500
	Product	10.0.1.0
	Public Folder	10.0.0.0

Administrative Objects Used in Expression (Filter) Accesses

This table lists the administrative objects in the framework that are referenced within expressions in policy accesses. If these objects are renamed after the installation is complete, you will need to update any expressions that reference the object names to reflect the changed name.

The expressions used in policy access filters do not recognize symbolic names so when an expression references an admin object, the standard name for the object is used and not the symbolic name. One exception is when the expression calls a JPO and an admin object, often a role, is passed into the JPO for input. In this case, the JPO recognizes symbolic names and so they are used. Admin objects used in expressions in this way, as input for a

JPO, are not included in this list. Since admin object symbolic names do not change anyway, the expressions that include symbolic names shouldn't need updating.

Admin Object Type	Name	Policy that references the object in an expression access
Attribute	Access Type	<i>Controlled Design Release Rev2</i> <i>Controlled Production Release Rev2</i> <i>Document</i> <i>Message</i>
	Global Read	<i>Workspace Vaults</i>
	Originator	<i>Issue</i>
	Primary Key	<i>Attribute Group</i> <i>Buyer Desk</i> <i>Document</i> <i>Issue Categorization</i> <i>Line Item</i> <i>Line Item Template</i> <i>Package</i> <i>RFQ</i> <i>RFQ Cancelled</i> <i>RFQ History Version</i> <i>RFQ Pending Version</i> <i>RFQ Quotation</i> <i>RFQ Template</i> <i>Supplier Line Item</i> <i>Supplier Line Item Price Period</i>
	Risk Visibility	<i>Project Risk</i>
	Secondary Keys	<i>Document</i> <i>RFQ</i> <i>RFQ Quotation</i>
	Security Classification	<i>Restricted Legacy Technical Specification</i> <i>Restricted Technical Specification</i>
Person	Workspace Lead Grantor	<i>Route Template</i>
Relationship	Assigned Buyer	<i>Organization List</i>
	Assigned Feature	<i>Product Feature</i>
	Assigned Incident	<i>Issue</i>
	Assigned Issue	<i>Impact</i> <i>Incident</i>

Admin Object Type	Name	Policy that references the object in an expression access
Relationships, cont.	Assigned Member	Admin Specification Section Basic Specification Section Characteristic CoOwned List Legacy Technical Specification Person Restricted Legacy Technical Specification Restricted Technical Specification Route Template SCO Technical Specification Technical Specification Template Template Filter UDA
	Assigned Requirement	Requirement Source File
	Assigned Tasks	Document
	Assigned Test Case	Test Case Use Case
	Assigned To Specification Office	Admin Specification Section Basic Specification Section Characteristic CoOwned List Legacy Technical Specification Person Restricted Legacy Technical Specification Restricted Technical Specification Route Template SCO Technical Specification Technical Specification Template Template Filter UDA
	Business Unit Employee	Restricted Legacy Technical Specification Restricted Technical Specification Source File
	Business Unit Owns	Restricted Legacy Technical Specification Restricted Technical Specification
	Buyer Desk Bid List	Organization List

Admin Object Type	Name	Policy that references the object in an expression access
Relationships, cont.	Company Document	<i>Document</i>
	CoOwned	<i>Admin Specification Section</i> <i>Basic Specification Section</i> <i>CoOwned List</i> <i>Legacy Technical Specification</i> <i>Restricted Legacy Technical Specification</i> <i>Restricted Technical Specification</i> <i>SCO</i> <i>Technical Specification</i> <i>Technical Specification Template</i>
	CoOwns	<i>Admin Specification Section</i> <i>Basic Specification Section</i> <i>CoOwned List</i> <i>Legacy Technical Specification</i> <i>Restricted Legacy Technical Specification</i> <i>Restricted Technical Specification</i> <i>SCO</i> <i>Technical Specification</i> <i>Technical Specification Template</i>
	Company Product	<i>Configurable Product</i>
	Data Vaults (relationship_ProjectVaults)	<i>Workspace Vaults</i>
	Design Responsibility	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Document Holder	<i>Document</i>
	Employee	<i>Configurable Product</i> <i>EC Part</i> <i>Legacy Technical Specification</i> <i>Manufacturer Equivalent</i> <i>Restricted Legacy Technical Specification</i> <i>Restricted Technical Specification</i> <i>Technical Specification</i> <i>Technical Specification Template</i>
	Feature List From	<i>Product Feature</i>
	Feature List To	<i>Product Feature</i>
	Global Read	<i>Workspace Vaults</i>
	Governed By	<i>Impact</i> <i>Incident</i>

Admin Object Type	Name	Policy that references the object in an expression access
Relationships, cont.	Has Documents	<i>Controlled Design Release Rev2</i> <i>Controlled Production Release Rev2</i>
	Impact Analysis	<i>Impact</i>
	Line Item	<i>Line Item</i> <i>Issue</i>
	Line Item Object	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Line Item Split	<i>Line Item</i> <i>Issue Categorization</i>
	Link URL	<i>URL</i>
	Manufacturing Responsibility	<i>Manufacturer Equivalent</i>
	Meeting Attachments	<i>Document</i>
	Message	<i>Message</i>
	Message Attachments	<i>Document</i>
	Project Access Key	<i>Document</i>
	Project Access List	<i>Document</i>
	Recommended Suppliers	<i>Legacy Technical Specification</i> <i>Restricted Legacy Technical Specification</i> <i>Restricted Technical Specification</i> <i>Technical Specification</i> <i>Technical Specification Template</i>
	Reply	<i>Message</i>
	Route Templates	<i>Route Template</i>
	Sub Vaults	<i>Workspace Vaults</i>
	Supplier	<i>Document</i>
	Supplier Buyer Line Item	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Supplier Line Item	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Supplier Response	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Supply Responsibility	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Sub Vaults	<i>Workspace Vaults</i>
	Task Deliverable	<i>Document</i>

Admin Object Type	Name	Policy that references the object in an expression access
Relationships, cont.	Testing Responsibility	<i>EC Part</i> <i>Manufacturer Equivalent</i>
	Thread	<i>Thread</i>
	Vaulted Documents Rev2	<i>Document</i>
	Vaulted Objects	<i>Document</i>
	Version	<i>Version</i> <i>Version Document</i>

Associations

Association	Description
Public Add	Not person User Agent
Public Delete	Not person User Agent
Public Modify	Not person User Agent
Public Read	Not person User Agent

Attributes

A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z

Attribute Name	Description/Possible Values
Abatement Plan	String The steps needed to mitigate a risk.
Absence Delegation	String Name of a person who is delegated an absent person's tasks.
Absence End Date	Date/Time The date/time that a person's absence ends.
Absence Of	String An item that the characteristic is absent of. Attribute for Microbiological Characteristic.
Absence Start Date	Date/Time The date/time that a person's absence starts.
Access Type	String Indicates how access is applied. –Inherited (default)—Access list comes from other objects. –Specific—Changes have been made to the inherited accesses. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Action Taken	String A description of the actions taken to solve the issue, typically entered by the assigned Issue Manager or Assignee.
Actual Benefit	Real The totaled interval benefit for a specified date.
Actual Build Date	Date Stores the actual date the hardware or software was built.
Actual Completion Date	Date/Time
Actual Cost	Real The totaled interval cost for a specified date.
Actual Effort	String The actual effort for each state.
Actual End Date	Date/Time The actual calculated date/time that a risk was completed.
Actual Net Benefit	Real The totaled interval benefit for a specified date.

Attribute Name	Description/Possible Values
Actual Start Date	Date/Time The actual calculated date/time that a risk was started.
Actual Time	Real The actual effort spent by the engineer to fix the bug or feature.
Address	String, Multi-line Address of the originator.
Address1	String Address of the location.
Address2	String Address of the location.
Agreed Unit Price	Real, Default is 0.0. The agreed (after negotiation) price of a unit of a product or material.
Allow Delegation	Boolean Indicates whether a route task's assignee can delegate it to another user. –True –False
Allow Duplicates	Boolean, Default is True Determine whether the vault will allow duplicates (applies to vaults and files).
Allow User Edit	String Stores the symbolic name of the role that has been granted access privileges to edit the business object.
Annual Part Quantity	Integer, Default is 0. Quantity required of each part or line item.
Appearance	String Stores appearance of a specification.
Approval Status	String –Abstain –Approve –Ignore –Reject –Signature Reset –None
Approved Date	Date/Time Stores the approved date of the specification.
Approver	String The name of the users who will approve the business object.

Attribute Name	Description/Possible Values
Approvers Responsibility	String –Engineering –Manufacturing –Marketing –Material Control –Product Management –Quality
Assessment Comments	String The justification for the overall assessment status.
Assessment Status	String The assessment status assigned to the overall project: – --- (Default) – Red – Yellow – Green
Assignee Role	String Allows tasks to be assigned to a role – this is at project definition before a resource is committed.
Assignee Set Due Date	String Indicates whether the task assignee can set the task due date: –Yes –No (Default)
Attribute Filter	String, Hidden Stores the symbolic names of the company mandatory and optional attributes selected from the template for display and the user’s preference for requiring an entry for the attribute. Any attributes not requiring entry should have their symbolic name appended with an question mark. For example, suppose the value for the Attribute Filter contains: AttrSymbolic1 AttrSymbolic2? AttrSymbolic3 Only the attributes corresponding to symbolic names 1 and 3 require entry.
Attribute Group Type	String Stores the symbolic name of an Attribute Group type’s subtype.
Authorized Temporary Specification	String Specifies whether a specification is ATS or Not.
Average	Real Specifies the average.
Award Status	String Indicates if the supplier’s “RFQ Quotation” has been awarded the buyer's contract. Current values are: –None –Awarded –Awarded – Supplier Hidden Note that a string attribute is used even though the current range is “Boolean” in case more award types need to be added in the future.
Base Price	Real Base price for the product.

Attribute Name	Description/Possible Values
Baseline Current End Date	Date/Time, Hidden
Baseline Current Start Date	Date/Time, Hidden
Baseline Initial End Date	Date/Time, Hidden
Baseline Initial Start Date	Date/Time, Hidden
Baseline Text	String Baseline of the software release.
Benefit Interval	String Specifies the interval length: –Monthly (Default) –Quarterly –Weekly
Benefit Interval End Date	Date Stores the interval end date of the benefit spreadsheet.
Benefit Interval Start Date	Date Stores the interval start date of the benefit spreadsheet.
Billing Address	String The organization’s billing address.
Bounding Box	String Stores the diagonal vertices of a bounding box bounding the model. Default is Unassigned.
Branch To	String, Hidden Used for programmatic purposes only. Stores the place to branch to. –None (Default) –Cancel
Build Serial Number	String Assigned Serial Number of the hardware or software build. (This is for information only in the first release of Product Central.)
Business Benefit	String, Multiline The Business benefits of this goal.
Cage Code	Integer, Default is 0. An ID issued by the federal government for a company or business unit.
Capability Status	String The status of a supplier’s capability for a process type (for example, Manufacturing process: Machining).
Capacity	String Stores capacity for a specification.

Attribute Name	Description/Possible Values
CAS Number	String Specifies the CAS Number.
Category of Change	String Category for reporting purposes: –Unassigned (default) –Cost Reduction –Drafting Error –Facilitate Manufacturing –Marketing Requirement –New Production Introduction –Product Improvement –Vendor Requirement
Change Board	String The change board that is responsible for reviewing the change. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
Checkin Reason	String The reason a new version (revision) has been checked in. Each checkin reason is preceded with a formatted string “stamp”, which indicates the date/time of the checkin and the person who performed it. All text after this “stamp” and before the next stamp is a checkin reason. The following is how this string is formatted: <pre><emxEntry><timestamp><PersonObjectName>entered text 1 ...<emxEntry><timestamp><PersonObjectName>entered text 2 ...</pre>
City	String
Claimed Attribute List	String, hidden Stores the list of claimed attributes.
Closed By	String The name of the user (last, first) who closed the incident (typically Product Manager or QE).
Closed Date	Date The date the Incident was closed. This date can differ from the date the incident was promoted to the issue state.
Collaboration Request Type	String The type of collaboration between two companies. – one-way – two-way (default)
Color	String Stores color for a specification.
Comment	String, Multi-line General comment entered by user.
Comments	String, Multi-line A text field containing comments that are concatenated, dated, and the author identified.

Attribute Name	Description/Possible Values
Commitment Level	String –Proposed –Candidate –Committed
Company Attributes	String Stores a formatted list of the symbolic names of an Attribute Group subtype’s attributes (value of the Attribute Group Type attribute) that are available in a line item template. Each symbolic name value in the list has a character following it to indicate if the attribute is company mandatory (M) or optional (O).
Company Buyer Attributes	String Stores a formatted list of the symbolic names of the “Request To Supplier” type's attributes that are included in the RFQ template. Each symbolic name value in the list has a character following it to indicate if the attribute is company mandatory (“M”) or optional (“O”).
Company Supplier Attributes	String Stores a formatted list of the symbolic names of the “RFQ Quotation” type's attributes that are included in the RFQ template. Each symbolic name value in the list has a character following it to indicate if the attribute is company mandatory (“M”) or optional (“O”).
Comparison Operator	String Determines how left and right expression are compared: –Incompatible (default) –Co-Dependent –Requires
Component Location	String
Conference Call Access Code	String Conference call access code, if the meeting has a con-call support.
Conference Call Number	String Conference call number, for meetings that have con-call support.
Constraint Date	String Identifies the date requirement of a project or task.
Constraints	String, multiline Known limitations.
Container Cost	Real, Default is 0.0. The one-time cost of reusable containers to package and deliver parts for production.
Contextual Error	String Used in Boolean Compatibility Rule to store the Contextual Error.
Co-Owners	String Stores a comma-delimited list of person object names who are assigned as co-owners for an object. <i>This attribute was originally named Co-Owner and therefore its symbolic name is attribute_CoOwner.</i>

Attribute Name	Description/Possible Values
Cost Assessment	String, Multi-line An assessment of the relative cost to make a part.
Cost Interval	String The length of the cost interval. Range values are: – Monthly (Default) – Quarterly – Weekly
Cost Interval End Date	Date The end date for the cost spreadsheet
Cost Interval Start Date	Date The start date for the cost spreadsheet.
Count	Integer The number of messages for a file. Default is 0.
Country	String
Create Folder	String Controls whether a project member can create a folder. –Yes –No (default)
Create Route	String Controls whether a workspace member can create a route. –Yes –No (default)
Create Vault	String Controls whether a project member can create a category and add it to a project. –Yes –No (default)
Created On	Date The date the object was created.
Currency	String Stores standard currency names. Customer can add ranges as necessary. There is no business logic expecting specific currency abbreviations. –Dollar –Won –Yen –Yuan Renminbi –Francs –Deutsche Marks –Pound –Unassigned (default)
Current Route Node	String

Attribute Name	Description/Possible Values
Current Version	Integer The current version number of the product version. Default is 0.
Date Deleted	Time/Date The date the subtask was deleted.
Date Offset From	String Determines if the offset is taken from the task create date or the route start date: –Route Start Date –Task Create Date
Date Requirement	String Identifies constraints of the date on a project or task.
Day Delta	Integer The number of days to add to a timestamp. Value can be positive or negative.
Default Policy	String The default policy to use for an object type.
Default Responses	String The list of responses used for RFQ creation or editing.
Default Responsibility	String The values of responsibility used for RFQ creation or editing.
Default Selection	String Holds the default selection for a particular option: –Yes –No (default)
Defect Definition	String, multiline A description of the defects.
Defects Per Million Opportunities	Real
Defects Per Unit	Integer
Density	String Stores density for a specification.
Dependency Type	String Identifies the type of dependency between two tasks. –FS, finish to start, normal duration start calculation, default –SF, start to finish, must start on start date –FF, finish to finish, must finish on finish date –SS, start to start
Description	String The description of the object.
Design Group Owner	String The business group that owns a document.

Attribute Name	Description/Possible Values
Designated User	String A list of users who are allowed to access the document. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
Diameter	String Stores diameter for a specification.
Difficulty	String Determines the difficulty of a Feature. –High –Medium –Low (default)
Disposition (Field Return)	String Disposition of parts that are returned from the field: –Balance –No Change –Rework –Scrap –Unassigned (default) –Use-up
Disposition (In Field)	String Disposition of parts that are in field: –Balance –No Change –Rework –Scrap –Unassigned (default) –Use-up
Disposition (In Process)	String Disposition of parts that are in process: –Balance –No Change –Rework –Scrap –Unassigned (default) –Use-up
Disposition (In Stock)	String Disposition of parts that are in stock: –Balance –No Change –Rework –Scrap –Unassigned (default) –Use-up

Attribute Name	Description/Possible Values
Disposition (On Order)	String Disposition of parts that are on order: –Balance –No Change –Rework –Scrap –Unassigned (default) –Use-up
Disposition Comments	String, Multi-Line The disposition of an incident.
Division	String Division name for the company or business unit.
Document Classification	String Identifies a classification for how a document is to be used.
Due Date Offset	String Offset, in days, from the route start date or task create date.
DUNS Number	Integer, Default is 0. The ID issued by Dunn & Bradstreet for the company or business unit.
EBOM ID	String
ECR Evaluator	String Indicates which person will evaluate the ECR. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
ECR Originator Organization	String The Primary Key of the organization that originated the ECR for a line item. <i>Deprecated in 10.0.1.0.</i>
Effective Date	Date/time The date/time the Risk Priority Number (RPN) is effective.
Effectivity Date	Date/time The date/time the part revision becomes effective.
Effort Classification	String Stores the Classification of Effort. –Major –Minor (default)
EINECS Number	String Specifies the EINECS Number.
Email Address	String The email address of the person.

Attribute Name	Description/Possible Values
End Effectivity	Date/Time Last date the product can be sold. If Released and Effective Products is chosen for Web Availability, then the current date must be within the product's start and end effectivity date range in order to be presented on the Product Configurator Web pages.
End Effectivity Date	Date/Time, Hidden
End State	String, hidden
Entered Name	String The name entered by the person who creates the business object. This is stored when the actual name for the business object is system generated.
Error Category	String, Multi-line Range: –Data Handling Error –Design Error –Interface Error –Logic Error –Operational Error –Other –Requirements Error –Typing Error
Error Message	String, Multi-Line Message to be displayed.
Error Navigation	String, Multi-line The sequence of steps to reproduce the error.
Error Report	String, Multi-line Validation report.
Escalation Date	Date, Hidden Date of last escalation.
Escalation Recipients	String The users who should receive notification that tasks are late.
Escalation Required	String Enables the owner to identify issues that need upper management support for resolution.
eService Connect Relation	String An attribute for the eService Object Generator. The relationship the Object Generator program should use when connecting the newly-created primary object to the newly-created additional object.
eService Constructor Arguments	String Holds constructor arguments for a JPO.
eService Method Name	String Lets JPO trigger developers define the method to invoke in a JPO. This is important if you use the same JPO with multiple methods in many different triggers. If this value is not set in the eService Trigger Manager Parameters object, then the JPO Trigger Manager uses the main (mxMain) method of the JPO.

Attribute Name	Description/Possible Values
eService Name Prefix	String An attribute for the eService Object Generator. Adds a prefix to the name that is generated automatically for objects. The characters you specify for the prefix are prepended to the unique number that is generated for the name.
eService Name Suffix	String An attribute for the eService Object Generator. Adds a suffix to the name that is generated automatically for objects. The characters you specify for the suffix are appended to the unique number that is generated for the name.
eService Next Number	String An attribute for the eService Number Generator. The number assigned for the name of the next object that is created using an eService Object Generator. Every time a new object is created automatically, the system increments the Next Number value by adding one to the current value.
eService Processing Time Limit	String An attribute for the eService Object Generator. Number of seconds the eService Object Generator has to create a new object. If the time limit is reached and the object is not created, the system presents an error message.
eService Program Argument 1 - 15	String An attribute for the eService Trigger Program Parameters. Contains a value for an argument passed to the trigger program named in the eService Program Name attribute. A Trigger Program Parameter can pass up to 15 arguments.
eService Program Argument Desc 1 - 15	String An attribute for the eService Trigger Program Parameters. Describes what each argument does. This can also include other details like the default value for the argument, etc.
eService Program Name	String An attribute for the eService Trigger Program Parameters. Specifies the name of the program object to be executed (not the symbolic name).
eService Retry Count	String An attribute for the eService Object Generator. Number of times the eService Object Generator object should be rechecked when it is locked.
eService Retry Delay	String An attribute for the eService Object Generator. Time period in milliseconds before the eService Object Generator object should be accessed if it is locked.
eService Safety Policy	String An attribute for the eService Object Generator. Specifies the policy that governs the type of object being created. The object generator program uses this policy only when the value for the ObjectPolicy parameter that is passed to the object generator program is null and more than one policy can govern the object type. The policy name must be specified using the symbolic name. For example, to specify the Part policy, the value for the attribute would be policy_Part.
eService Safety Vault	String An attribute for the eService Object Generator. Under normal circumstances, the object generator creates the business object in the current vault (the vault the user is logged in). But if the current vault is ADMINISTRATION, the object generator creates the object in the vault specified as the eService Safety Vault. Must be specified using the symbolic name for the vault (for example, vault_eServiceProduction).

Attribute Name	Description/Possible Values
eService Sequence Number	String An attribute for the eService Trigger Program Parameters. Specifies the order in which the programs for a trigger event are to be executed by the emxTriggerManager. If two programs have the same sequence number, the sequence they run in will be arbitrary. This attribute must contain a numeric value other than 0.
eService Target States	String An attribute for the eService Trigger Program Parameters. Used by the Trigger Manager when firing promote and demote triggers. Stores a comma separated list of the symbolic names of states. The Trigger Manager fires the trigger program only when promoting or demoting to one of the listed states. If the target state is not in the list, the trigger does not fire. If the attribute is not populated, the Trigger Manager fires the program regardless of the target state.
Estimate Schedule Impact	Integer Number of hours that the schedule could be delayed by resolving this incident.
Estimated Benefit	Real The estimated totaled interval benefit for a specified date.
Estimated Completion Date	Date/Time Estimated completion date for test case.
Estimated Cost	Real, Default is 0.00. Range is ≥ 0.0000 Early estimate of the cost for initial planning purposes.
Estimated Effort	String An estimate of the effort for each state.
Estimated End Date	Date/Time The estimated date/time that a risk must be mitigated.
Estimated Max	Real The estimated maximum effort needed.
Estimated Min	Real The estimated minimum effort needed.
Estimated Net Benefit	Real The totaled interval benefit for a specified date.
Estimated Schedule Impact	Real Estimated schedule impact in days if the incident is taken up.
Estimated Start Date	Date/Time The estimated date/time that a risk will start being mitigated.

Attribute Name	Description/Possible Values
Event Type	String The type of event a person is subscribing to. For projects: – Route Started – Route Completed – Category Created – Category Deleted – Member Added – Member Removed For routes: – File Check In (new file added) – Task Completed For categories and subcategories: – File Added – File Removed For files: – File Checked In (new revision checked in) – File Check Out (lock for edit) For messages: – New Message – New Reply
Expiration Date	Date/Time Specifies the expiration date of a specification.
Expression	String, Multi-line Boolean expression that is evaluated for inclusion and exclusion rules. The expression consists of operands and operators. The operands are feature/option pairs and the supported operators are AND, OR, NOT, (,)
Extend Unit Price Formula	String Indicates whether buyers can add to the unit price definition for RFQs based on the RFQ template: –Yes (Default) –No
Fax Number	String Person's fax number.
Feature Category	String Groups features into general categories. Not currently implemented. –Category 1 –Category 2 –Category 3 –Unassigned (default)
Feature Classification	String Classification under which the given software feature comes: –Internal –Major –Minor

Attribute Name	Description/Possible Values
Feature Selection Type	String How many options within a feature can be selected. –Must Select Only One (default) –Must Select At Least One –May Select Only One –May Select One Or More
File Contents	String –BOM Data (default) –Package Data
File Extension	String The supported file formats in the organization.
File Site	String The URL for a company’s File Collaboration Server.
File Store Symbolic Name	String The symbolic name of the File Store to use.
Finance Comments	String Comments regarding the Finance Assessment status.
Finance Status	String The finance status assigned to the overall project: – --- (Default) – Red – Yellow – Green
Find Number	String Symbol used to identify a part on a drawing.
Find Type	String The type of find for Save Query.
First Name	String First name of the originator.
Folder Access	String Controls the level of access a project member has for a folder. –Add –Add Remove –None –Read (Default) –Read Write –Remove
FTP Directory	String Alternative directory to store related files.
FTP Host	String Alternative computer to store related files.
General Description of Change	String, Multi-line An explanation of the change that is being requested.

Attribute Name	Description/Possible Values
General Requirements	String Stores general requirements of a specification.
Global Attribute List	String, hidden Stores the list of global attributes.
Global Read	String Indicates if the workspace vault (folder) has Global Read access. Global Read access means all users can access the vault, not just those given access through the workspace. – True – False <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Goal	String, multiline A description of the goal or objective.
Governing ECO	String
Hardware Vendor	String List of different company names: –IBM –Microsoft –Oracle
Headquarters Site	String Whether the location is headquarters or not.
Height	String Stores height for a specification.
Hidden Attribute List	String, hidden Stores the list of hidden attributes.
Home Phone Number	String
Host Meetings	String Determines whether this person can host meetings—WebEx integration. –Yes –No (default)
Hours Delta	Integer The number of hours to add to a timestamp. Value can be positive or negative.
Icon Mail	Boolean Identifies whether icon mail should be enabled.
Image Horizontal Size	String Stores the size of the image.

Attribute Name	Description/Possible Values
Image UOM	String The valid units of measure for the image. –Pixel (default) –Inches
Image Vertical Size	String Stores the size of the image.
Impact	String The Impact due to the incident from the customer point of view: –Low –Medium –High
Impact Analysis Effort	Integer Number of hours it has taken to determine the impact.
Implementation Effort	Integer The number of hours that it will take to resolve the incident.
Incident Disposition	String Status of incident: –Design Limitation –Duplicate –Fix –Not Reproducible –Previously Fixed –Rejected
Initial Resource	Real, Default is 0.0. Amount of resource that is started with before any is applied.
Instance Parameters	String, Multi-line Stores variational parameter information that lets a family resolve into an instance.
Interval Date	Date The interval in the cost spreadsheet. Used to determine the format in which to display on the spreadsheet.
Is Template	String Indicates whether it is a template in the context of “Find Like” search.
Is Version	Boolean Identifies if the product is a version. –True –False (default)
Is Version Object	Boolean Hidden, identifies if the document is connected as version. –True –False (default)
Issue Category	String Tells under which category the Issue belongs.

Attribute Name	Description/Possible Values
Issue Classification	String Tells under which classification the Issue belongs.
Issued Date	Date/Time Stores the Issued Date for a specification.
Keywords	String, Multi-line List of keywords related to the business object for faster and efficient search.
Kind of Product	String Stores Kind of Product property for a specification.
Label	String Label number for media object.
Lag Time	Integer The lead or lag time in days between dependant tasks. 0 is default
Language	String Stores the language of the file checked in to the document. –English (default) –Japanese –French –German –Chinese –Korean –Italian
Last Legacy Update	Date/Time Date/time the last update occurred from the legacy system.
Last Login Date	Date The date the person last logged into an application.
Last Name	String Last name of the originator.
Lead Time	String How long it takes for a part to arrive once it is ordered: –1 week –2 weeks –3 weeks –4 weeks –6 weeks –8 weeks –12 weeks –20 weeks –Unassigned (default)
Lead Time Assessment	String, multi-line An assessment of the lead time to make a part.
Lead Time Production	Integer The number of days between receipt of order and delivery of the first production parts. Default is 0.

Attribute Name	Description/Possible Values
Lead Time Prototype	Integer The number of days between receipt-of-order and the delivery of the first prototype part.
Lead Time Tooling	Integer The number of days between receipt-of-order and availability of production tooling.
Ledger Account Number	String The ledger account number (meta-data).
Left Expression	String, Multi-line Left side of compatibility rule, consisting of operands and operators. The operands are feature/option pairs and the supported operators are AND, OR, NOT, (,)
Legacy	String Identifies whether the Technical Specification is legacy.
Legacy System Identifier	String, Hidden Store the identifier that the legacy system uses for imported RFQ data.
Legacy System Source	String, Hidden Stores the name of the legacy system
Length	String Stores length for a specification.
Life Cycle Cost Issues	String, Multi-line Cost issues concerning the lifecycle of the incident.
Line Item Roll-up Calculation Formulas	String, Hidden The line item numeric attribute/expression pairs separated by a ~ for calculation. This is the line item specific roll-up formula for a quotation attribute that is actually used for calculation. It includes selects and symbolic names that are not presented to users.
Line Item Roll-up Formulas	String, Hidden The line item numeric attribute and expression pairs separated by a ~. This is the line item specific roll-up formula for a quotation attribute as displayed to users. The actual calculation used is stored in the corresponding calculation attribute.
Line Item Template Scope	String Defines the usage scope for Attribute Group business objects. –Private – All Line Items –Private – Specific Line Items –Public – All Line Items –Public – Specific Line Items
Link URL	String The URL to an internal or external Website or URL link. Allows the creation of hyperlinked external URLs. The URL may or may not be a Website. It could be an internal URL link to a “document”.
List Price	Real, Default is 0.00. The current price charged to a customer who orders an option.
Local Attribute List	String, hidden Stores the list of local attributes.

Attribute Name	Description/Possible Values
Location Code	String Brief code to identify a location.
Location Preference	String Attribute on Allocation Responsibility relationship, tells whether a location is preferred location for MEP. –Preferred (default) –Not preferred
Location Status	String Attribute on Relationship Allocation responsibility, tells whether a particular location is authorized to inventory MEP for production purposes. –Required (default) –Authorized –Not authorized.
Login Type	String Determines whether the person must enter a secure ID, made up of a pin number plus the number from a key fob, to log into a MatrixOne Application. For more information, see Secure ID and RSA Token Authentication in Chapter 7. –Standard (default) –Secure ID
Lot Effectivity	String
Lower Limit	String Stores lower limit for a specification.
Lower Specification Limit	Real
Manual Title	String Title of Standards Manual/Section/Page.
Manufacturability	String, Multi-line Describes the type of processing to be used.
Manufacturing Site	String Indicates whether the location is a manufacturing site or not.
Marketing Feature	String Determines if the feature is a marketing feature or not. –Yes (default) –No
Marketing Name	String Stores the marketing name of the object.
Marketing Text	String The text entered on Features, Products, Images, Model and Product Lines.
Markup Author	String The name of the last person who modified the markup.
Markup Base Filename	String The base document to which the markup is attached.

Attribute Name	Description/Possible Values
Markup Base Format	The base format of the base document to which the markup is attached.
Markup Connection ID	String Unique ID for the connection.
Markup Function	String Purpose of the markup: –add –modify –delete –replace
Markup Tool	String The name of the tool that is used to create the markup.
Material Category	String –Concrete –Glass –Metal (default) –Plastic –Rubber
Material Limitation	String A limitation of part material(s) for a process capability.
Max Palette Height	String Stores max palette height for a specification.
Maximum Packaging Height	String The maximum packaging height for the object.
Maximum Quantity	Real, Default is 1.0. Maximum quantity that can be ordered of specified option.
Mean	Real
Measure of Success	String
Meeting Attendee	String Indicates whether attendees are added or removed from the meeting: –Add –Remove –Notified
Meeting Duration	Real Default is 0.0. Expected duration for meeting.
Meeting ID	String ID for meeting.
Meeting Key	String A string generated by WebEx that allows a person to join a meeting. The key is passed from our applications to WebEx and is not displayed to the user.

Attribute Name	Description/Possible Values
Meeting Location	String Indicates the physical meeting location, for meetings in which participants meet at a physical location.
Meeting Owner	String – Yes – No (Default)
Meeting Password	String User password used for scheduling and hosting Webex meetings. Not needed to just participate in a meeting.
Meeting Site ID	String WebEx site ID for a company.
Meeting Site Name	String URL assigned to company by WebEx for accessing their meetings.
Meeting Start Date Time	date/time
Meeting Username	String Username used for scheduling and hosting Webex meetings. Not needed to just participate in a meeting.
Metric Integer 01 through 15	Integer 15 generic metric attributes for metrics whose values are integers. Default is 0.
Metric Real 01 through 10	Integer 10 generic metric attributes for metrics whose values are real. Default is 0.
Metric Source	String How the metrics were captured.
Middle Name	String, Multi-line Person's middle name.
Minimum Quantity	Real, Default is 1.0. Minimum quantity that can be ordered of specified option. Note: If minimum and maximum quantity values are both 1, then no edit box will be presented to the Web user for quantity input. Applies only to systems using Product Configurator along with Product Option Manager.
Model Type	String This attribute is used for classifying a CAD Model. The values of this attribute should be ranges that reflect the customer's specific classification scheme. Note that this attribute can be used to reduce the amount of sub typing by storing the type information in this attribute. Default is Unassigned.
Move Files to Version	Boolean Hidden, specifies whether active version of a file will be the master object or in "Active Version" object. –True –False (default)
MQL Script	String An executable MQL string used for defining an item.

Attribute Name	Description/Possible Values
Next Message	Integer An attribute on the Thread object that indicates the sequence order for the next message created in the thread.
No Bid Reason	String A supplier's reason for not bidding on a line item. Default is "(none)".
Notes	String, Multi-line A technical comment about the business object.
Objective	String, Multi-Line The purpose and goals for the build. What is the theme of the build?
Odour	String Stores odor for a specification.
On-line Meeting Instructions	String Instructions for accessing the meeting with the on-line service provider.
On-line Meeting Provider	String Indicates the meeting provider, for example, WebEx.
Operational Definition	String, multiline Description of how the quality metrics will be measured.
Opportunity	String, multiline The benefit of this measured quality.
Order Quantity	Integer The quantity for each item (product configuration) included in a sales order.
Order Recipient	String The person who will receive the items in the sales order.
Ordered Options	String, Multi-line List of all the options that have been ordered for a product configuration and the quantities. The list is structured with each option on a line that specifies the Feature :: Option :: Quantity.
Organization Fax Number	String The main fax number for the organization or the fax number of the primary contact.
Organization ID	String An internal ID, often a number, that the host company gives to a customer or supplier company.
Organization Name	String, Multi-line The name of the organization, company, or business unit—could be the host, supplier, or customer.
Organization Phone Number	String The phone number of the organization.
Origin	String Stores origin for a Specification.

Attribute Name	Description/Possible Values
Origination	String The origin of the incident: –Customer –Distributor –Internal –Partner –Supplier
Originator	String The user who created or revised the business object. This attribute is populated automatically for all types by a trigger program (see Set Originator Attribute in Appendix A).
Other Requirements Affected	String, Multi-line Other requirements that could be affected.
Other Tasks Affected	String, Multi-line Other tasks that could be affected.
Out of Bounds	String, multiline Description of the boundaries or limits.
Packaging Size	String The packaging size for the object.
Packed On Date	Date The date on which the object was packed.
Packed Requirement	String Miscellaneous packing requirement, for example, Fragile.
Pager Number	String Pager number for the person.
Palette Size	String Stores palette size for a specification.
Parallel Node Procession Rule	String Indicates how parallel Route Tasks proceed to the next ordered tasks. There are no defined ranges in case complex conditions with formatted values are necessary (for example, “At least 2”). However, for now only two rule values are stored: “any” and “all”.
Parent Part ID	String The ID for the part that is the parent for the substitute.
Parent Route Node	String

Attribute Name	Description/Possible Values
Part Classification	String Classify the type of part: –Unassigned (default) –Extrusion –Hardware –Machined –Molded –Other –Printed Circuit Board –Sheet Metal –Software –Wiring Harness <i>This attributes was originally named Classification and therefore its symbolic name is attribute_Classification.</i>
Part Family Base Number	String A static number that represents the main part number pattern for the part family, such as 123456.
Part Family Last Used Sequence	Integer, Hidden The last sequence generated for the part family. For example, if the sequence is 3 and the sequence pattern is “001,002,003...”, then three part numbers have been generated for the part family.
Part Family Name Generator On	Boolean Lets the Administrator turn on or off part family naming. Default is True.
Part Family Pattern Separator	String The character separator to use in the part numbers for the part family. The default is -.
Part Family Prefix Pattern	String The prefix for the part family’s name, such as R. A prefix is optional.
Part Family Sequence Pattern	String The format pattern used to generate the part numbers for a Part Family. The format should be set to the number of digits to zero-fill the part number tabulation based on the integer value in the hidden attribute Part Family Last Used Sequence. Default is 0000. The first number generated would be the [prefix]-[base]-0001-[suffix]. If the attribute was set to 000000, the next part generated would be [prefix]-[base]-000001-[suffix].
Part Family Suffix Pattern	String The suffix for the part family’s name. A suffix is optional.
Percent Complete	String Calculated field for project % complete
Percent Concentration	Real Attribute for BOM management in BOM relationship. Specifies the % concentration. Default is 1.0.
Percentage	Real Stores a percentage value. 50% is stored as 50.0 rather than .50.
Period	String Period covered by Scorecard data.

Attribute Name	Description/Possible Values
Period Type	String The amount of time the data pertains to: –Monthly –Quarterly –Yearly
PH	String Stores PH for a specification.
Planned Benefit	Real The totaled interval benefit cost for a specified date.
Planned Build Date	Date Stores the planned date for the build.
Planned Cost	Real The totaled interval cost for a specified date.
Planned Net Benefit	Real The totaled interval benefit cost for a specified date.
Plant ID	String –Central Plant (default) –Plant 1 –Plant 2 –Plant 3
Platform	String The platform the originator was using when the incident was identified: –AIX –Linux –HP –Macintosh –Solaris –Unix –Windows
Postal Code	Real
Precise BOM	String, Multi-line Structured text that lists all of the Parts/ Assemblies directly below the option level that are required for a specific order based on the evaluation of the inclusion rules. <i>Deprecated in version 10.0.1.0.</i>
Preferences	String, Multi-line, Hidden Used to store a user's session preferences, such as the tasks the user has bookmarked for MyTasks.
Primary Contact	String Identifies the person who is the primary contact for a customer organization.

Attribute Name	Description/Possible Values
Primary Key	String, Hidden Unique key for an organization that owns an object. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Prioritization Benefit	Integer Prioritization benefits on a scale of 1 to 9. Range values include every number from 1 to 9.
Priority	String The priority of the change: –Low (default) –Medium –High –Pre-Assigned
Problem Statement	String, multiline A description of the problem.
Process Qualification Status	String Indicates whether the buyer perceives the supplier to have a certain capability to perform a process.
Product	String, Multi-line The software product for which the incident has been reported.
Production End Date	Date/Time The date on which a supplier will end production of a part.
Production Make Buy Code	String Indicate whether the part is make or buy: –Make –Buy –Unassigned (default)
Production Purpose	String, Multi-Line An indication of the phase(s) of production for which a supplier is expected to deliver parts: –All Requirements –Not Applicable (Default) –Pre-Production –Pre-Production and Production –Production –Production and Spares –Prototype –Prototype and Pre-Production –Spares
Production Start Date	Date/Time The date on which a supplier will start production of a part.
Program Rule	String Program name to be executed by rule extension.

Attribute Name	Description/Possible Values
Program Rule Argument	String Arguments to be passed to program rule by rule extension.
Project Access	String –Financial Reviewer –Project Assessor –Project Lead –Project Member (default) –Project Owner
Project Function	String Technology Sales Marketing
Project Member Access	String Not currently used. –Read (Default) –Read Write
Project Name	String Project Nickname (used for search).
Project Role	String Project Members role on the project. – Change Boards – Core Team Member – Design Engineer – ECR Chairman – ECR Coordinator – Employee – Financial Reviewer – Manufacturing Engineer – Organization Manager – Product Obsolescence Manager – Program Manager – Project Assessor – Project Lead – Senior Design Engineer – Supplier Engineer – Supplier Representative – != ! – [null]
Project Security Approach	String The method for controlling the “consumer” access privileges of read, checkout, toconnect, and todisconnect for Team Central-controlled data. Range values currently include: –projectaccess: Access to the project objects is controlled through the existing folder accesses. –globalread: Access to the workspace objects can be given to any other user regardless of the Team Central accesses. Some applications may have their own UI or business rules to restrict access.
Project Task Actual Duration	Real

Attribute Name	Description/Possible Values
Project Task Actual Finish Date	Date
Project Task Actual Start Date	Date
Project Task Duration	Real
Project Task Finish Date	Date
Project Task Resource	String
Project Task Start Date	Date
Project Type	String Identifies the project type. The range values can be configured.
Project Visibility	String Identifies the security level for the project. The range values cannot be configured. – Company – Members
Promised Delivery Date	Date/Time The date the ordered items will be delivered; usually assigned by the Manufacturing Resource Procurement (MRP) system.
Properties Form Name	String Stores the form name for the specification properties page.
Properties Template Form Name	String Stores the form name for the specification Template properties page.
Prototype Capability	String A supplier's ability to make non-production, prototype parts with a process: –Unknown –Yes –No
Public	Boolean Tells whether the issue is private or public. –True –False
Purchase Decision	String The way a group of parts is to be purchased: –Separate Source For Each Part –Single Source For All Parts –Unassigned (default)
Purchase Order	String
Quality Assessment	String, Multi-line An assessment of the relative quality achievable while making a part.
Quality Impact	String, Multi-line Impact on quality.

Attribute Name	Description/Possible Values
Quality Type	String The type of quality being measured: –Discrete –Continuous
Quantity	Real, Default is 1. The quantity for each part included in the Generic BOM and in the Engineering BOM.
Quotation Header Roll-up Calculation Formulas	String, Hidden The quotation numeric attribute/expression pairs separated by a ~ for calculation. This is the Quotation Header formula for a quotation attribute that is actually used for calculation. It includes selects and symbolic names that are not presented to users.
Quotation Header Roll-up Formulas	String, Hidden The quotation numeric attribute/expression pairs separated by a ~ for display to users. This is the Quotation Header formula for a quotation attribute. The actual calculation used is stored in the corresponding calculation attribute.
Quotation Response Review	String Indicates when a Buyer can review the Quotation responses. –closed: Buyers cannot review RFQ Quotation business objects until the Quote Requested By Date is reached. –open: The buyer can review RFQ Quotation business objects as suppliers return them. (default)
Quote Requested by Date	Date/Time The date by which the quotation is requested in a Request for Quote.
Rate	Real Stores a value for making a conversion or calculation.
Raw Material Package	String Stores Raw Material Package for a specification.
Reason For Cancel	String, Multi-line The reason specified by the user for cancelling an ECO.
Reason For Change	String, Multi-line Reason for why this change should be made.
Reason for Closure	String, Multi-line The comments entered by the Product Manager when closing the incident.
Receiving Plant	String Buyers indication of the name of the plant that will receive a split volume of the line item quantity.
Recipient	String The person who should receive a message or some other object. The value may be an actual person object name or some other string that indicates a particular recipient.
Recurrence Interval	Integer, Hidden The number of days to re-alert.
Reference Designator	String Lets the designer specify placement/instance of a component inside an assembly. Default is Unassigned.

Attribute Name	Description/Possible Values
Region	String Geographic region for a location.
Regional Attribute List	String, hidden Stores the list of regional attributes.
Regression	String, Multi-line Yes, if the software was working fine previously but not now. No, otherwise.
Regression Version	String The version of the software in which the feature was working fine.
Relative Cost	Integer, Default is 1. Relative cost on a scale of 1 to 9. Range values include every number from 1 to 9.
Relative Penalty	Integer Relative penalty on a scale of 1 to 9. Range values include every number from 1 to 9.
Relative Risk	Integer Relative risk on a scale of 1 to 9. Range values include every number from 1 to 9.
Release	String Identifier for release.
Release Date	Date/time Release date of content.
Release Distribution Group	String Indicates the user groups to be notified at release. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
Release Type	String A flag used to signal when an object is ready to use the next revision sequence. –Design Release (default) –Production Release
Renamed From	String Used to record the old name of a CAD Model that the current model was renamed from.
Render Error Message	String Stores Rendition Error.

Attribute Name	Description/Possible Values
Render Language	String Stores the language of the rendered PDF. Range values include: -English (default) -German -Italian -French -Japanese -Spanish -Portuguese -Simplified Chinese -Chinese
Render Success Flag	String Stores Rendition status.
Rendered File Name	String Stores the rendered file name in the specification object.
Rendition In Process	Boolean Stores the name of the table for the section display. -True -False (default)
Replacement Scope	String -All Uses -Some Uses -Unassigned (default)
Request Disposition	String Buyer's disposition of the supplier's response. -Accepted -No Response -Rejected -Resubmitted -Unassigned (default)
Request Parameters	String, Multi-line Holds the request parameters for HTTP requests.
Requested Delivery Date	Date/Time The date the customer would like the items in the sales order delivered.
Requested Work	String The list of work in a Request for Quote that needs to be performed. -Design -Match* -Production Manufacturing -Prototype Manufacturing -Testing
Required Availability	String The time frame in which an object must be available for use. The time frame could be a specific date, a month, a quarter, or just ASAP.

Attribute Name	Description/Possible Values
Requirement	String Ad-hoc requirement.
Requirement Category	String Stores a grouping category name for requirements.
Requirement Classification	String Stores a classification of the requirement. –None –Functional (default) –Non-Functional –Constraint
Requirement Importance	String The customer priority for the given user requirement. –Least Important (Default) –Low –Mediocre –High –Very High
Resolution Recommendation	String Recommendation on how to resolve the Issue.
Resolution Statement	String A description of the why the Issue can be closed.
Resolve Effort	Real The actual effort taken to resolve the incident.
Resource	Real (+/-), Default is 1.0. Amount of resource used by a given option on per item basis.
Resource Comments	String Comments regarding the Resource Assessment status.
Resource ID	String Name user assigns to a resource for short, unique identification.
Resource Maximum	Real, Default is 1.0. The maximum amount of a resource that can be used to be a valid order.
Resource Minimum	Real, Default is 1.0. The minimum amount of a resource that can be used to be a valid order.
Resource Operation	String Determines how resource is applied: –Increment (default) –Decrement
Resource Status	String The resource status assigned to the overall project: – --- (Default) – Red – Yellow – Green

Attribute Name	Description/Possible Values
Responsibility	String, Multi-line The values of responsibility used for creating/editing a Quotation/Manufacturing Assessment.
Responsible Design Engineer	String Indicates the lead design engineer who is assigned to the change or product. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
Responsible Manufacturing Engineer	String Indicates the lead manufacturing engineer who is assigned to the change. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
Responsible Product Manager	String Indicates the lead product manager who is assigned to the product. When the framework is installed, this attribute has no range values. So the values should be populated using a popup range helper or by adding a range program that gets a list of values. See Configuring Range Values for Attributes in Chapter 6.
Restart Upon Task Rejection	String Identifies whether route is restarted when a task is rejected. –True –False
Restrict Members	String Identifies whether members can be chosen from another company. –Organization –All (default)
Result	String The final result of a Request for Quote after the request is closed. –Cancelled –Completed –Unassigned
Review Comments Needed	String Determines if the Review Comments have to be entered before promote/demote: –Yes –No (Default)
Review Task	String Determines if the route owner must review the task before it can be completed: –Yes –No (Default)
Reviewers Comments	String, Multi-line A chronologically ordered set of reviewers comments about the change.
Right Expression	String, Multi-line Right side of compatibility rule, consisting of operands and operators. The operands are feature/option pairs and the supported operators are AND, OR, NOT, (,)

Attribute Name	Description/Possible Values
Risk Category	String –Technical –Quality –Resource –Market –Schedule
Risk Comments	String Comments regarding the Risk Assessment status.
Risk Factor	Integer
Risk Impact	Integer 1, 2, 3, 4, 5
Risk Probability	Integer 1, 2, 3, 4, 5
Risk RPN Value	Real number The calculated value between Risk Impact and Risk Probability.
Risk Status	String The risk status assigned to the overall project: – --- (Default) – Red – Yellow – Green
Risk Type	String The type of risk. Range vaults include: –Cost (Default) –Quality –Technical –Resource –Performance –Schedule –Benefits –Manufacturability –Other
Risk Visibility	String The visibility access to the Risk object type. Range values are: –Public –Restricted (Default) <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Route Access	String Controls the access level a project member has for a route. –None –Read (Default) –Read Write

Attribute Name	Description/Possible Values
Route Action	String –Approve –Comment (default) –Information Only –Investigate –Notify Only
Route Base Policy	String The policy that governs the object to be routed.
Route Base Purpose	String Stores the purpose of a List. –Approval –Review –Standard (default)
Route Base State	String Specifies the state (from the Route Base Policy) the routed object must be in when the route is initiated. The object cannot be promoted from the specified state until the route is complete. If no particular state is required, use Ad Hoc. Default is Ad Hoc.
Route Branch	String
Route Completion Action	String –Notify Connected Owner (default) –Promote Connected Object
Route Instructions	String, Multiline Information the route creator enters for each person in the route regarding what the person needs to do to complete the task.
Route Node ID	String This is an attribute on the Inbox Task object. It is the ID for the Route Node relationship that connects the route and the person to which the task is assigned.
Route Sequence	String A number that represents the order in which the folder is routed to each person. This attribute is on the Route Node relationship, which connects a route (or route template) to a person in the route. If the route sequence is 1, the folder is routed to this person first. If the route sequence is 2, the folder is routed to the person after the first person completes the assigned task, and so on. If the route sequences are the same, the folder is routed to these people simultaneously.
Route Status	String –Finished –Not Started –Started –Stopped
Route Task User	String Symbolic names for role and group assignment. This attribute supports Unassigned Tasks. Both Route and Route Templates will support Unassigned Tasks and Assignment of Roles/Groups to tasks.

Attribute Name	Description/Possible Values
Route Task User Company	String Symbolic names of the company or business units for role and group assignment. This is necessary for exchange environments where only the users assigned to the specified company will be assigned to the route task.
RTS ID	String Contains temporary Request to Supplier (RTS) ID values.
Rule Type	String Defines whether true expression includes or excludes the item: –Inclusion (default) –Exclusion
RuleExtension ID	String ID assigned to the rule extension. The rule extension ID is unique for every rule extension and is the only way by which a rule extension is identified.
Safety Description	String Stores Safety Description for a specification.
Safety Symbol	String Stores Safety Symbol for a specification.
Safety Warning	String Stores Safety Warning for a specification.
Schedule Comments	String Comments regarding the Schedule Assessment status.
Schedule Duration Unit	String Project Duration in weeks or days Weeks Days (default)
Schedule Status	String The schedule status assigned to the overall project: – --- (Default) – Red – Yellow – Green
Scheduled Completion Date	Date/Time The date the person should complete the assigned task.
Scope	String The areas that are covered by the project or the range of actions. Whether the Route Template should be available to only the creator or to all route members for the company. –Enterprise –User
Search Type	String The type of the searched object(s).

Attribute Name	Description/Possible Values
Secondary Keys	String, Hidden Unique key for an organization that uses an object. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Secondary Vaults	String List of additional vaults available for a company.
Section Access	String Defines access for different sections.
Section Type	String Stores the type of display used for the spec view section.
Security Classification	String Stores the specification security classification. A space is the default range value. –Highly Restricted –Confidential –(space) (default) <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Send Reminder	Integer, Hidden The number of days from due date to send a reminder. May be negative.
Sequence Order	Integer, Default is 0. The order in which features and options are presented to customer on Web.
Serial Effectivity	String
Service Make Buy Code	String Indicate whether the part is make or buy: –Make –Buy –Unassigned (default)
Severity	String Severity of the incident: –Low –Medium –High
Shape Limitation	String A limitation of part shape for a process capability.
Share Types	String Stores the different types that can be shared with a collaboration partner.
Sheet Number	String The sheet number of the sheet attached to the business object.

Attribute Name	Description/Possible Values
Shelf Life	String Stores Shelf Life for a specification.
Ship To	String, Multi-line The address the customer wants the items in the sales order to be delivered to.
Shipping Address	String Shipping address for an organization.
Show Sub Components	String, Multi-line A navigation switch allowing access from a Part object to the Assembly relationship: –None (default) –All Levels
Show Target Cost	String, Multi-line A view switch that controls access to the Target Cost attribute for a part: –Yes (default) –No
Show This Supplier Price to Other Suppliers	String, Multi-line A switch to control visibility of a supplier agreed price to other supplier: –Yes –No (default)
Show This Supplier to Other Suppliers	String, Multi-line A switch to control visibility of a supplier to other suppliers: –Yes –No (default)
Sigma	Real
Size Limitation	String A limitation of part dimension for a process capability.
Software Vendor	String List of different platform names: –IBM –Microsoft –Oracle
Source Qualification Status	String The status of the qualification of a supplier's ability to manufacture a part: –Planning (default) –Qualifying –Qualified –Re-qualifying
Source Selection Status	String, Multi-line The selection status of suppliers responsible for a part: –Candidate –Selected

Attribute Name	Description/Possible Values
Sourcing Product	String The way a part is to be sourced for product requirements: –Make –Not Applicable (default) –Purchase Assembly Only –Purchase Complete –Purchase In Assembly –Purchase Incomplete
Sourcing Spares	String The way a part is to be sourced for spare requirements: –Make –Not Applicable (default) –Purchase Assembly Only –Purchase Complete –Purchase In Assembly –Purchase Incomplete
Spare Part	String –Yes –No (default)
Spatial Location	String Stores the geometric information (as a 4x4 matrix) needed to position the component within an assembly. Default is Unassigned.
Specific Description of Change	String, Multi-line Detail about changes that apply to the connected part only.
Specification Category	String Stores the Specification Category. –Standard (default) –Experimental
Specification View Form Name	String, hidden Stores the form name for the spec view.
Sponsoring Customer	String Stores the sponsoring customer for the requirement.
Standard Deviation	Real
Standard Lead Time Production	Integer The standard time (days) between receipt-of-order and delivery of the first production parts.
Standard Lead Time Prototype	Integer The standard time (days) between receipt-of-order and delivery of the first prototype parts.
Standard Lead Time Tooling	Integer The standard time (days) between receipt-of-order and availability of production tooling.
Start Effectivity	Date/Time First date product is effective. If Released and Effective Products is chosen for Web Availability, then the current date must be within the start and end effectivity date for the product in order to be presented on the Product Configurator Web pages.

Attribute Name	Description/Possible Values
Start Effectivity Date	Date/Time, Hidden
Start State	String, Hidden Start state for project.
State Comments	String, Multi-line Comments on promotion or demotion of a business object.
State/Region	String The state or region for the person or organization.
Status	String Status of an object.
Steps to Reproduce	String Describes how to reproduce issue.
Storage Conditions	String Stores Storage Conditions for a specification.
Sub Component Level	String Indicates the level of the sub component.
Sub Route Visibility	String Indicates whether the sub route is visible to people viewing the parent route. –Yes –No (default)
Subject	String Subject of the issue, message, etc.
Submitted Date	Date/Time The date on which a Request for Quote is submitted.
Subscription Level	String A person's subscription level, which determines whether the person can access some features.
Supplier Classification	String
Supplier Response	String The supplier's interest in bidding on part: –None –Submit –Unassigned (default)
Suspend Versioning	Boolean Specifies if version of files is allowed in the master object. Used to prevent versioning when the object connected to Document is in a specific state. –True –False (default) For example: Document ABC 0 is connected to a RFQ R-001 0. If RFQ is in the Response Complete state, Suspend Versioning is required to prevent any changes to connected Document objects. In this case, when the RFQ is promoted to Response Complete state, all the connected Documents will be modified (Suspend Versioning attribute will be set to True).

Attribute Name	Description/Possible Values
Symbol	String Safety or company-specific symbol name.
Synopsis	String Short description of business object.
Table Name	String Used to store the name of the table for the section display.
Target Cost	Real Target cost for a part.
Task Actual Duration	Real The actual effort.
Task Actual Finish Date	Date/Time
Task Actual Start Date	Date/Time
Task Comments Needed	String Determines if the Task Comments have to be entered before promoting the task. –Yes (default) –No
Task Edit Setting	String Defines the rules for edit of a task if the route is created from a template: –None (Default) –Maintain Exact Task List –Extend Task List –Modify Task List –Modify/Delete Task List
Task Estimated Duration	Real The estimated duration in days. Default is 0.0.
Task Estimated Duration Maximum	Real The estimated maximum effort.
Task Estimated Duration Minimum	Real The estimated minimum effort.
Task Estimated Finish Date	Date/Time
Task Estimated Start Date	Date/Time
Task Requirement	String Mandatory Optional (Default)

Attribute Name	Description/Possible Values
Task Transfer	Boolean Allows flexibility on how to copy tasks when defining project templates. A question can be related to multiple tasks. Questions can be formulated to generate a TRUE or FALSE answer (yes/no respectfully). If the question generates a Yes answer then all the tasks related to the question that have a “true” value will be copied to the new project. If the answer is “no” then all the tasks relationships with a “false” value, if any, will be copied to the new project. All mandatory tasks will be copied regardless of the questions. Any summary task that is copied will automatically carry it’s parent task (a task cannot be copied without it’s parent)
Task WBS	String Default is 1.
Technical Feature	String Determines if the feature is a technical feature or not. –Yes –No (default)
Template Level	String Stores the level of the template. –Global –Regional –Local (default)
Template Task	String Indicates that the task is created from a template: –Yes –No (Default)
Time of Occurrence	String The time the incident occurred: –Design phase –Implementation phase –Maintenance phase –Requirement phase –Testing phase
Title	String The title for a project, task, meeting, or document.
Tolerance	String Stores Tolerance for a specification.
Tolerance Lower	Real Lower Tolerance for the Characteristic noted.
Tolerance Upper	Real Upper Tolerance for the Characteristic noted.
Tooling Assessment	String, Multi-line An assessment of the tooling requirement to make a part.
Tooling Cost	Real, Default is 0.0. The one-time cost of tooling to manufacture a part.

Attribute Name	Description/Possible Values
Total Budget Cost	Real The totaled budget for the project (roll-up value).
Total Cost	Real The sum of all sub-total costs for a sales order. The sub-total cost is the unit price of the configuration multiplied by the quantity ordered.
Total Count	Real Total Count for the Characteristic noted.
UDA Mandatory	String Indicates whether entry for the User-Defined Attribute is mandatory: –Yes (Default) –No
UDA Name	String The name of the UDA as entered by the user who added it.
Unit of Measure	String The inventory's unit of measure: –each (default) –feet –gallon –inch –pound
Unit of Measure	String Stores Unit of Measure for a specification.
Unit Price	Real, Default is 0.0. The price for an entire product configuration (also called ordered item) at the time the item was ordered. The unit price is calculated by adding the base price for the configurable product plus the unit price for each ordered option. OR Unit cost of the supplying part.
Unit Price Calculation Formula	String, Hidden The expression, in string format, used to calculate the Unit Price of a line item using attribute group attributes and UDAs. The calculation formula is the actual formula used to calculate the unit price and includes selects and symbolic names that are not presented to users.
Unit Price Formula	String The unit price formula for an RFQ or RFQ template, as displayed to users. The actual calculation used is stored in the corresponding calculation attribute.
Units	Real Units related to a Characteristic.
Upper Limit	String Stores Upper Limit for a specification.
Upper Specification Limit	Real

Attribute Name	Description/Possible Values
Usage	String Why the part is included in the Bill of Material: –Standard (default) –Reference –As Required –Per Sales Order
Use Case Estimated Duration	String Stores the number of days it takes to complete a use case.
User Requirement Classification	String Classification under which the given user requirement comes.
User Requirement Importance	String Importance of a requirement based on customer's perspective: –Least Important (default) –Low –Mediocre –High –Very High
User Symbolic Name	String Symbolic name of user, role, group, or association.
Validation Comments	String, Multi-Line Validation comments entered by the user who executed the test.
Validation Effort	Integer Number of hours it will take to validate the incident.
Validation Status	String Indicates the status of the validation program: –Not Validated (default) –Validation Failed –Validation Passed
Value	String Stores value for a specification.
Value (Real)	Real Stores a generic real value.
Value (String)	String Stores a generic string value.
Value Type	Type of data values contained in Scorecard. –Actual (default) –Budget –Estimate –Forecast –Target

Attribute Name	Description/Possible Values
Vault Access	String Controls the access a project member has for a category. –None –Read (Default) –Read Write –Add –Remove –Add Remove
Version	String, Multi-line The version of the software release for which the incident has been reported.
Version Date	String Identifies the date of checkin for documents.
Viewable Base Filename	String This attribute records the filename of the base file.
Viewable Base Format	String This attribute records the symbolic name of the format of the business object that holds the base file.
Viewable State	String This attribute records the status of a viewable with regards to the model it is derived from. –Empty –Synchronized –Unsynchronized
Viewing Tool	String This attribute records the name of a viewing tool. Default is Unassigned.
Wall Thickness	String Stores Wall Thickness for a specification.
Warning Description	String Description used as a warning on packaging or marketing material.
Web Availability	String Specifies the criteria for when customers can create configurations for a product using the Product Configurator Web pages. Ranges for this attribute are: Released Products(default) – customer can configure any product that is in the Release state Released and Effective Products – customer can configure only products that are in Release state AND are currently effective (have a start effectivity date equal to or greater than the current date and an end effectivity date less than the current date).
Web Site	String, Multi-line The URL of the organization’s Web site.
Weight	Real
Weight Limitation	String A limitation of part weight for a process capability.
Where Used Component Reference	String Holds a list of the Type, Name, Revisions of the components from which the “where used” is executed.

Attribute Name	Description/Possible Values
Width	String Stores width for a specification.
Work Around	String, Multiline The workaround for a bug or other incident.
Work Phone Number	String Person's phone number at work.

Business Objects

This section describes the business objects installed with the framework.

Business Object	Description
AEFPowerView	Portal object installed by the AEF, used for Desktop PowerView page. Applications can add their own channels to AEFPowerView.
AEFPowerViewChannel	Channel object that contains the first channel of the PowerView. All other applications can add tabs to this channel.
Company Name Company object	This is a placeholder for your company. You should replace “Company Name” with the name of your company. For more details about setting up your company, see Setting Up in Chapter 6.
eService Number Generator	See eService Object Generator below.
eService Object Generator	This type of object and the eService Number Generator are responsible for automatically naming objects. The only autonamer objects installed with the framework are the ones for common objects (such as Publish Subscribe and Event objects). The autonamer objects for objects specific to an application are installed with the application.
eService Trigger Program Parameters	There are several business objects of this type that run trigger programs to: –populate the Originator attribute whenever any type is created or revised –perform actions when a task is completed (populate the completion date, notify route owners when the last task is completed) –performs actions when a route is started (create a task for route members with order numbers of 1 and notify them) These objects are stored in the eService Administration vault. For information about triggers, see Configuring Automatic Business Rules (Triggers) in Chapter 6.
Sample data	If sample data was installed along with the framework, then the framework includes many test persons. There are three test persons to represent each role in the framework. Each person is added as an person administrative object and as a business object of the type Person. The sample Person business objects are all in the Active state. Each person is assigned to the role indicated in the person’s name. The naming convention for the persons is “Test ROLE_NAME”, “Test1 ROLE_NAME”, and “Test2 ROLE_NAME”. For example, the test persons added for the Product Manager role are named Test ProductManager, Test1 ProductManager, and Test2 ProductManager. All sample business objects are stored in the eService Sample vault.
Test Everything Person object	This person corresponds to the person administrative object Test Everything. This person is assigned to all roles and should be used for testing. The business object is connected to Company Name with the Employee and Company Representative relationships. This person is stored in the eService Production vault.

Commands

This table lists the command objects installed with the framework. Some of these commands represent common tools used by all MatrixOne applications, such as the logout tool.

Other commands are pre-configured to be either a tree category or menu link, as listed in the Configured As column. Commands configured to be tree categories have settings that display the href URL in the content frame, which is the main frame on the right side of the MatrixOne application window, and have an image file assigned to them for the category icon. Commands configured to be toolbar items have settings that display the href URL in a popup window. For information about how to configure commands, see [Building Menus and the Global Toolbar](#), [Building Navigation Trees](#), and [Building Action Bars](#) in Chapter 9.

Command	Description	Configured As:
AEFAddToCollection	Add To Collection link for adding selected objects to a collection.	Menu link
AEFAppDownloadPackage	Download Package link for downloading .zip or .gz file containing files attached to objects.	Menu link
AEFAppMenuToolbar	Tool that lets users go to Preferences.	Menu link
AEFAttachmentsTreeCategory	Attachments category for IconMail.	Tree category
AEFChangePasswordToolbar	Tool that lets users change their own password.	Menu link
AEFCollectionDelete	Remove Selected link for Collections table page.	Menu link
AEFCollectionItems	Collections category for the Category tree.	Tree category
AEFCollectionItemsDelete	Remove Selected link for removing items in a collection.	Menu link
AEFCollections	Command installed by the AEF that will be the first command of the first channel (AEFPowerViewChannel) of the PowerView page.	Tab for channel menu
AEFCollectionAddExisting	Command installed by the AEF to add items to a collection.	Menu link
AEFCollectionsCreateNew	Command installed by the AEF to create a new collection.	Menu link
AEFCollectionsEditDetails	Command installed by the AEF to edit details of an existing collection	Menu link
AEFConversionPreference	Command that lets users choose their preferred currency. See Configuring Preference Pages in Chapter 9.	Preference page link
AEFExportTablePreference	Command that lets users choose their preferred export file format. See Configuring Preference Pages in Chapter 9.	Preference page link
AEFGeneral Search	Tool that opens the About page, which lists software versions.	Menu link
AEFHelpAboutToolbar	Tool that opens the About page, which lists software versions.	Menu link

Command	Description	Configured As:
AEFHistory	Command that calls a common history page, emxHistory.jsp, and that passes a parameter to the page so the page shows only events for the current revision of the object. This command is designed to be used as a tree category. For information on the common history page, see Configurable History Page and History Commands in Chapter 9.	Tree category
AEFHistoryAllRevisions	Command that calls a common history page, emxHistory.jsp, and that passes a parameter to the page so the page shows events for all revisions of the object. This command is designed to be used to display the History page in a popup window and should be used as a toolbar item. For information on the common history page, see Configurable History Page and History Commands in Chapter 9.	Menu link
AEFHistoryAllVersions	Command that calls a common history page, emxHistory.jsp, and that passes a parameter to the page so the page shows events for all revisions of the object but where “revisions” is referred to as “versions”. For example, Team Central uses the term “version” for document revisions. This command is designed to be used to display the History page in a popup window and should be used as a toolbar item. For information on the common history page, see Configurable History Page and History Commands in Chapter 9.	Menu link
AEFHomePagePreference	Command that lets users choose the initial page they see when they log in and the page that is shown when they click the Home icon in the global toolbar. See Configuring Preference Pages in Chapter 9.	Preference page link
AEFHomeToolbar	Tool that takes users to their preferred Home page.	Global Toolbar tool
AEFIconMailToolbar	Tool that opens the IconMail page.	Menu link
AEFLanguagePreference	Command that lets users choose the language they want to receive system notifications in. See Configuring Preference Pages in Chapter 9.	Preference page link
AEFLifecycle	Command that calls the framework Lifecycle page, emxLifecycle.jsp. This is also a menu link on the Locations page.	Tree category, Menu link
AEFLifecycleDemote	The Demote link for the Lifecycle page.	Menu link
AEFLifecyclePromote	The Promote link for the Lifecycle page.	Menu link
AEFLogoutToolbar	Command installed by the AEF that will be the only command actions toolbar (AEFPowerViewActionsToolbar) of the PowerView page.	Toolbar for page
AEFMonitor	A command available only for Administration Managers and assigned to the AEFAdminTools menu, which is a My Desk menu. Displays a pages that shows session-based Matrix Collaboration server information. For a description of the information displayed, see the AEF Help system by clicking the Help button on the Monitor page.	My Desk menu link
AEFMQL	A command available only for Administration Managers and assigned to the AEFAdminTools menu, which is a My Desk menu. Displays a page that lets the administrator run MQL commands.	My Desk menu link

Command	Description	Configured As:
AEFPageHistoryToolbar	Tool that opens the Page History page, which lists the last 50 pages a user has visited within the MatrixOne applications. Users can use Page History to revisit pages, bookmark pages, and copy URLs for pages to include in emails or documents that invite other users to visit a particular page.	Global Toolbar tool
AEFPaginationPreference	Command that lets users choose the default for how pages should be paginated. See Configuring Preference Pages in Chapter 9.	Preference page link
AEFPowerViewHome	Command to access the system-defined PowerView page.	Menu link
AEFPreferenceToolbar	Command that lets users choose general and application-specific preferences. See Configuring Preference Pages in Chapter 9.	Preference page link
AEFReloadCacheToolbar	Command that lets Administration Managers refresh the user interface when configuring dynamic user interface administrative objects. When the Reload Cache link is clicked, the entire cache data is reset and all cached data expires. So, any further request to an admin object from UI cache obtains the data fresh from the database and cached again. For information about automatic cache updates, see the ComponentAge property in Miscellaneous Properties in Chapter 7. The cached data also includes trigger data, such as input arguments, program names, and changes to the state of the eService Trigger Program Parameters objects. When changes occur to this trigger data, the cache must be reloaded to reflect the change.	Menu link
AEFReviseSearch	Tool that redisplay the Search page from the Search Results page.	Toolbar tool
AEFSaveAsSearch	Command that opens the Save As popup from the Search page.	Menu link
AEFSavedSearches	Command that opens the Save Searches popup where users can select previously-saved search items.	Menu link
AEFSaveSearch	Command that opens the Save Search popup where users can specify a name to save search criteria for later use.	Menu link
AEFSeparator	Dividing line between toolbar links and drop-down menu links.	Separator for the configurable toolbar links
AEFSystemData	Command that lets Administration Managers access information about the system configuration.	Menu link
AEFTimeZonePreference	Command that lets users choose the time zone for data/time stamps. See Configuring Preference Pages in Chapter 9.	Preference page link

Formats

Formats are installed as required with the MatrixOne applications. The generic format is the only format installed with the Application Exchange Framework.

Format	Description
generic	This format accepts any document type and launches an application based on the file extension. (Works on Windows only, not Unix.)
HPGL	HPGL markup format.
IGES	IGES markup format.
Image	Image format.
JT	3D JT image files
Markup	This format is for Markup documents, which have mrk extensions.
MSDS	MSDS format.
PDF	PDF format.
TIF	TIF format.
TXT	TXT format.
XML	XML format.

Groups

Groups are generally used to route (notify by IconMail/email and assign ownership) a business object to specific people when the business object is promoted. In contrast, roles are used to control access to business objects (within the policy that governs the business object).

If you chose to install sample data with the framework, you'll also get some sample groups, such as Sample Change Boards and Sample Distribution Groups.

Group	Description
Accounts Receivable Department	Used in the Product Configurator application. People who receive payment for sales orders and then promote sales orders to the Payment Received state.
Build Management	Used in the Software Manager application. People responsible for managing Software Build objects.
Change Boards	Used in Engineering Central. This is the parent group of all the change board groups. Members of the change board are responsible for reviewing ECRs. At least one Change Board must exist — the applications are designed around the assumption that there is generally more than one in a company. Product Lines contain a Change Board attribute, declaring which change board is responsible for it. All Change Boards must be child groups of this group.
Company Name	Used in all applications. You should change the name of this group to the name of your company, making sure it's the exact same name used for the Company business object that represents your company. For more information, see Defining Your Company (The Host Company) in Chapter 6. All employees of your company should be members of this group. This approach is helpful when users outside the company are provided access to the database through the MatrixOne applications or the Matrix Web Navigator. Certain accesses may then be given to employees, but not to others. If you installed sample data with the framework, many sample users are assigned to this group.
Design Engineering	Used in Product Central. Design Engineers are responsible for adding engineering-related rules for a configurable product, including engineering-related compatibility rules, inclusion rules, rule extensions, and resources. Design Engineers also create parts, generate the Generic BOM, and review and test the product structure.
Distribution Groups	Used in Engineering Central. Distribution Group is the parent group of all the Distribution Groups. Members of Distribution Groups are notified upon the release of an ECO and its related parts. At least one Distribution Group must be created for the process to work correctly.

Group	Description
Documentation	Used in Product Central. Persons in charge of documenting the software.
Order Entry	Used in Product Central. People responsible for validating product configurations (when the system does not validate automatically) and generating Precise BOMs for product configurations. Order Entry Clerks also may enter the order into the MRP system and enter the promised delivery date for a sales order, as assigned by the MRP system.
Product Management	Used in Product Central. People responsible for adding features and options to configurable products, adding marketing-related compatibility rules, and reviewing and testing the product structure.
Production Support	Used in the Product Configurator application. People responsible for monitoring sales orders as they move through the manufacturing system.
Project Originators	Used in Project Manager. Parent is Project Users.
Project Users	Used in Project Manager. Parent of Project Originators.
Sample Change Board 1, 2, 3	Used in Engineering Central. Children of Change Boards.
Sample Distribution Group 1, 2, 3, 4	Used in Engineering Central. Children of Distribution Groups.
Shadow Agent	Used by all applications. The Shadow Agent group has ALL permissions. Programs push the context to a member of this group, the User Agent person, when the program requires access not granted to most users. For example, in order to ensure that Configurable Product objects are created using the Create Product wizard, and not the Object/New/Original dialogs, create access is only allowed by the Shadow Agent group.
Shipping Department	Used in the Product Configurator application. People responsible for shipping ordered items to customers and promoting sales orders to the Shipped state.
Software Change Board	Used in Product Central. People who assess and assign everything involved in the development of a software product, including specifications, incidents, features, impact analyses, etc. All Change Boards are part of this group, including the Change Control Board (CCB).
Software Engineering	Used in Product Central. People who write code for the software, perform impact analyses, and write functional specifications.

Group	Description
Software Quality Engineering	Used in Product Central. People who are involved with the entire quality development process of the software.
Software Release Engineering	Used in the Software Manager application. People responsible for software components, software library, and the integrity of sequencing the configuration management environment. These people also create/oversee the development/test of runtime environments and support both the development and QA environment. They manage the release of new versions of software to customers. They build release and packing tools, and develop release policy in conjunction with Quality Assurance, Development and Program Management.

Menus

The framework installs some common menu objects that are used by all MatrixOne applications. Menus are top level holders or collection points for other menus that represent sub-collection points (for example, a submenu) and for command objects that represent individual links in the user interface. Menus represent the My Desk and Actions menus, the global toolbar, navigation trees, the Preferences page, and other top level menu items. For information about how to configure menus, see [Building Menus and the Global Toolbar](#) in Chapter 9.

Menu	Description
Actions	Menu object that represents the Actions menu. Each application installs a menu connected to this menu object to represent the submenu for the application.
AEFAdminTools	Represents the Admin Tools menu on the My Desk menu. Contains links available to Administration Managers.
AEFCollectionsMenu	Represents the navigation tree for collections.
AEFGlobalSearch	Represents the Search menu on the global toolbar.
AEFGlobalToolbar	Represents the global toolbar.
AEFLifecycleMenu	Represents the Actions menu for the common Lifecycle page.
AEFPowerViewToolbar	Represents the toolbar menu that is installed by the AEF. This is used as the toolbar for the PowerView page. Uses the AEFPowerViewActionsToolbar
AEFPowerViewActionsToolbar	Represents the Actions menu that is installed by the AEF. This is used by the AEFPowerViewToolbar for its actions.
AEFSearchActionsToolbar	Represents the Actions menu on the Search page that allows saving searches.
AEFSearchResultToolbar	Represents the toolbar on the Search Results page.
AEFSearchToolbar	Represents the Search toolbar menu that is installed by the AEF. This is used as the toolbar for the Search page. Uses the AEFSearchActionsToolbar
AEFSearchTypes	Represents the search types menu on the Search page.
Default Tree	Menu object that represents the default navigation tree. See Using and Configuring the Default Tree in Chapter 9.
Event Post Processing	This menu object is for a future enhancement and will be documented when the enhancement is included in the framework and Matrix core. Engineering Central is currently using a partial implementation of this future capability, but customers should not use it until it is fully supported.
General	The menu that represents the General tab within the Preferences page. Commands assigned to this menu represent preferences used by most applications, such as the default language for notifications and the home page preference. See Configuring Preference Pages in Chapter 9.

Menu	Description
Icon Mail	Menu object for the IconMail navigation tree; contains the Attachments category.
My Desk	Menu object that represents the My Desk menu. Each application installs a menu connected to this menu object to represent the submenu for the application.
Preferences	Top level menu that defines items to include in the Preferences page, which is accessed from Preferences on the Tools menu. Menus assigned to this menu, such as General, define the tabs within Preferences. Commands assigned to these tab menus define the individual preferences. See Configuring Preference Pages in Chapter 9.
Toolbar	Menu object that represents the global toolbar. All global toolbar command objects installed with the framework are connected. Applications can install additional command objects to add to the global toolbar. For a list of commands installed with the framework, see Commands .
Tree	Menu for IconMail.

Persons

Aside from the default persons, creator and guest, the Application Exchange Framework contains these persons:

Person	Description
Administration User	The Administration User person is a hidden person that has all business administration privileges, as well as full Matrix access. The Administration User is assigned to the Administration Manager role and is designed to be responsible for configuring the system. This person and anyone else belonging to the Administration Manager role can use the Administration wizards in Matrix Navigator. This person requires a password, which is “administrationsecret” upon installation. You should change this password.
Common Access Grantor	Hidden. Assigned to the Access Grantor role. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6.
Corporate	The Corporate person owns all objects that are in the Released, Active, Obsolete, or Inactive state. When objects reach the end of their lifecycle, the system reassigns them to the Corporate user. Having the Corporate person own objects in the Released state allows standard users to perform owner-based queries that return only objects that currently require the users’ attention. The Corporate person is assigned the role of System Conversion Manager, which provides permissions for overriding and promoting objects in order to convert data from legacy systems and load them into the new system. This person is defined without a password. You should create a password for the Corporate person before going live into production.
Project Space Access Grantor	Used within Program Central and assigned to the Access Grantor role. Assigns accesses to project space. Hidden. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6.
Route Delegation Grantor	Used within Team Central and assigned to the Access Grantor role, hidden. Controls temporary granted accesses in order for a user to perform delegated tasks. The delegated grantee gets the accesses to the Routed objects and the route object that the original grantee has. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6.
Test Everything	Test Everything is assigned to all roles, and to the groups Company Name and Shadow Agent. Because the person is assigned to all roles, you may want to set the context using this person when you conduct tests and configure the system. Test Everything has no password.
Test persons	If sample data was installed along with the framework, then the framework includes many test persons. There are three test persons to represent each role in the framework. Each person is assigned to the role indicated in the person’s name. The naming convention for the persons is “Test ROLE_NAME”, “Test1 ROLE_NAME”, and “Test2 ROLE_NAME”. For example, the test persons added for the Product Manager role are named Test ProductManager, Test1 ProductManager, and Test2 ProductManager. For each test person, the sample data includes a business objects of the Person type. See Business Objects. Most sample persons are assigned to the group Company Name and the corresponding business objects are connected to the Company Name business objects, so they represent users from the host company.

Person	Description
Unmanaged Document Grantor	Used within Team Central and Sourcing Central and assigned to the Access Grantor role, hidden. Used to grant access to unmanaged documents in Sourcing Central, which are documents that are not in a Team Central workspace folder. Sourcing Central documents are attached to business objects and users generally have the same access to the attached documents as they do for the business object. This person grants accesses to business object co-owners and to users who can just view the documents. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6.
USERNAME	Every person who needs to log into a MatrixOne application must have a person admin object defined for them. For more information on the requirements for login, see Requirements for Successful Login in Chapter 7. When users choose framework-installed preferences, the system stores the selection using admin properties on the users' person object. For more information, see Configuring General Framework Preferences in Chapter 9.
User Agent	The User Agent person is a hidden person. This person has all business and system administration privileges as well as full Matrix access. The User Agent belongs to the Shadow Agent group. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6. Several programs change the session context to User Agent. These programs “push” the context to the User Agent person when the programs must perform actions that require forbidden privileges. The program then “pops” the context back to that of the initial user. For example, you may not want to give users permission to promote every object in a product structure. Therefore, the program that implements the promotion of a Product object to Released pushes the context to the User Agent so it can promote all objects in the product structure. For a sample of how to invoke push and pop in a JSP, see Using Push and Pop to a Shadow Agent in Appendix A. <i>History entries reflect the User Agent when context is pushed to this person.</i>
Workspace Access Grantor	Used within Team Central and assigned to the Access Grantor role, hidden. Grants accesses to all objects in the data structure in addition to the access that Workspace Member Grantor grants. These accesses are automatically granted down the data structure to at least all underlying folders (Workspace Vault) and content (Documents, Packages, etc.) business objects. The discussions (Message) objects for content also get the accesses from higher-level objects if their owners have not removed any accesses. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6.
Workspace Lead Grantor	Used within Team Central and assigned to the Access Grantor role, hidden. Grants accesses to the Workspace and folder (Workspace Vault) objects only for all Workspace Leads. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Workspace Member Grantor	Used within Team Central and assigned to the Access Grantor role, hidden. Grants read access to the Workspace object only for all Workspace members regardless of specified accesses. For information on the password for this and other shadow agents, see Changing Shadow Agent Passwords in Chapter 6.

Programs

The Application Exchange Framework contains many programs needed to manage the schema. The following table describes the most important and visible programs. The framework also contains some trigger programs used by many MatrixOne applications. For a description of these programs, see [Utility Trigger Programs](#) in Appendix A. Programs specific for an application are installed with the application.

Program	Description
eServiceCheckPageAccess.tcl	One of four programs that run when a person logs into the MatrixOne applications. This program checks to see if any of the roles to which the user is assigned has been assigned access to the JSP page for the selected feature.
eServicecommonWzrdListFromCommand.tcl	This program accepts any MQL command that generates a delimited list and prepares that list to load either a wizard widget or a program attribute.
eServicecommonGetViewers.tcl	Gets a list of viewers that have been registered for a specified format. Can be called from a JSP page to get viewers for a format. For more information on this program and on registering viewers, see Setting Up 2D and 3D Viewers in Chapter 6.
eServicecommonRevisionGenerator.tcl	Automatically generates unique revisions for objects. The revision is a number that is unique across the entire database. No other object in the database will have that revision at the time the program is called. The parameters are: –sObjectType: Type of an object for which to generate revision –sObjectName: Name of the object for which to generate revision –sObjectPolicy: Policy supported by the type specified (Optional, default policy for type is used if not given)
eServiceCreateList.tcl	This program is called from the programs that run when a person logs into the applications (eServicesGetUserApps, GetUserFeatures, GetUserSuites). It helps constructs a list having the suite name, application name, and JSP programs as its elements.
eServiceGetUserApps.tcl	One of four programs that run when a person logs into the applications. This program gets the names of the applications that have been installed for a specified suite.
eServiceGetUserFeatures.tcl	One of four programs that run when a person logs into the applications. This program gets the feature list for a specified application.
eServiceGetUserSuites.tcl	One of four programs that run when a person logs into the applications. This program gets the names of the suites that have been installed.
eServiceListSchemaNames.tcl	Returns all property names and their values as a (pipe)-separated list.
eServiceRegistry	This program object is a placeholder for suite, application, and feature registration. It contains no code.
eServiceSchemaVariableMapping.tcl	This program contains all the procedures required for adding administrative objects to the database, wherein collisions and version mismatches are handled. For more information, see How the System Identifies Administrative Objects: Symbolic Name Properties in Chapter 5 and <i>The Installation Process</i> in the <i>Matrix PLM Platform Installation Guide</i> .

Program	Description
eServiceSystemInformation.tcl	When an application is installed, a property is added to this program. The name of the property is the same as the application name and the value of the property is the version being installed. The code of this program contains a log of all the applications installed in sequence. The log contains three fields: Name of Application, Version, and Date installed. Do not manually edit the log. For more information, see eServiceApplicationSetVersion in Chapter 2 and eServiceVersionCompare in Chapter 2.
emxTriggerManager	Executes multiple trigger programs for a trigger event. Queries for the eService Trigger Program Parameter business objects that represent the trigger programs to be executed. The program then executes the trigger programs in the sequence and with the parameter values specified by the Trigger Program Parameters. For more information, see Configuring Automatic Business Rules (Triggers) in Chapter 6.
MatrixIniDefaults	Sets Matrix environment variables that are needed by the framework and MatrixOne applications, such as: MX_NORMAL_DATETIME_FORMAT=moy/dom/yr4 h12:min:sec mer MX_TERSE_DATETIME_FORMAT=moy/dom/yr4 MX_TRIGGER_RECURSION_DETECTION=SIGNATURE MX_JIT_TRIGGER_MACROS=TRUE MX_NESTED_TRIGGER_MACROS=TRUE <i>Note that if the ematrix.ini or UNIX startup scripts contain different values for settings that are defined in MatrixIniDefaults, Matrix uses the values in ematrix.ini or the startup scripts. Therefore, it's best to remove these settings from the ini/startup scripts to make sure the framework's settings are used.</i>

Relationships

The revision and clone rules for both sides of a relationship are None unless specified otherwise in the Description column.

A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z

Relationship	Description	Attribute
Active Version	Connects from DOCUMENTS to DOCUMENTS with Float on revision at To end.	—
Affected Code	Relationship between a request or a bug and the source file. To side revision rule is Float. <u>Deprecated in version 10.0.1.0.</u>	—
Allocation Responsibility	Relationship created when a MEP is associated to a location. Connects application context MEP and application part. Connects location and location equivalent object (location context MEP).	Location Status Location Preference
Alternate	Relationship between parts.	—
Application Part	Relationship created when application context MEP is created, connects an application part and the enterprise part it is representing.	—
Applied Against	Incident is found in this feature. <u>Deprecated in version 10.0.1.0.</u>	—
Applied Markup	Lets users associate markup objects to a ECO.	—
Assigned Build Part	The part that will be used to track the build of the product.	—
Assigned Buyer	Person assigned to the buyer desk. From side revision rule is Float <u>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</u>	—
Assigned Documents	Identifies documents for a given project or task item.	—

Relationship	Description	Attribute
Assigned Feature	Person assigned to the feature to complete all required tasks. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Assigned Incident	Person assigned to the incident. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Assigned Issue	Relationship between issue and assignee.	—
Assigned Meetings	Identifies the members of a meeting.	Meeting Attendee Meeting Owner
Assigned Member	Identifies the people assigned to a spec office. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Assigned Part	Relationship between part and builds.	—
Assigned Requirement	Person assigned to the requirement. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Assigned Risk	Identifies the assignees for a risk.	—
Assigned Sales Engineer	The Sales Engineer assigned to handle a Request for Quote.	—
Assigned Tasks	Connects a person to a task. Lets the project lead define the project member(s) who can own a specific task.	—
Assigned Test Case	Person assigned to the test case. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—

Relationship	Description	Attribute
Assigned To Buyers Desk	Indicates the assignment of items, such as purchase classes, suppliers, parts, and RTSes, to a buyer desk. To side revision rule is Replicate.	—
Assigned To Specification Office	Identifies the office that issues the technical specs and SCOs. To side revision and clone rules are Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Assigned Use Case	Person assigned to the use case.	—
Associated Drawing	Represents the association between CAD Drawings and the CAD Models from which they are derived and associated. A given CAD Model could have many drawings and a given CAD Drawing could have been derived from multiple CAD Models (example is an Assembly Drawing).	—
Attribute Group	Used to connect a business object that requires extended data to an Attribute Group.	Default Selection
Attribute Group Supplier	Temporary relationship used to indicate that an Attribute Group is intended for a Supplier Line Item that has not been created yet. From side clone and revision rules are Replicate.	—
Attribute Group Template	Indicates the template, if any, used to define the Attribute Group object's behavior. From side revision and clone rules are Replicate.	—
Authorized Temporary Specification	Identifies the authorized temporary spec for a spec.	—
Available On	Connects the Software Feature object to the respective Software Build object.	
Awarded Line Item	Defines the specific line item responses from a supplier quotation that the buyer has recommended for award.	Percentage
Baseline Log	Identifies baseline logs for a project space.	Comments Originator
Benefit Item Interval	Connects benefit items to interval item data.	Planned Benefit Estimated Benefit Actual Benefit Interval Date
Boolean Compatibility Rule	Indicates general Boolean compatibility rule for products.	—
BOS	The Bill of Specification structure attached to a Technical Specification. From side revision rule is Replicate.	Sequence Order

Relationship	Description	Attribute
Bug Impact Analysis	Impact analysis done for a bug. To side revision rule is Float. <i>Deprecated in version 10.0.1.0.</i>	—
Build Satisfied	Relationship that connects the Software Feature object to the respective Software Build object. <i>Deprecated in version 10.5.</i>	—
Build Satisfies	Connects the Feature object to the respective Build object. <i>This relationship was originally named Available On so its symbolic name is relationship_AvailableOn.</i>	—
Build Source Code	The source code that was used to generate the distribution file.	—
Build Specification	Describes the specifications/requirements for the build.	—
Business Goal Project Space	The projects that satisfy a given goal.	—
Business Unit Employee	Identifies an employee of a business unit. To side revision rule is Replicate <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Business Unit Owns	Identifies the business unit that owns technical specs and SCOs. To side revision and clone rules are Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Business Unit Project	A Company must sponsor a project. Optionally and additionally, the project can be sponsored by a Business Unit.	—
Business Unit Role	Assign roles at the business unit level <i>Deprecated in version 10.5.</i>	—

Relationship	Description	Attribute
Buyer Desk Bid List	Associates an Organization List with a Buyer Desk so all Buyer Desk members can use the list to easily retrieve suppliers for an RFQ bid list. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Buys For	Relates Buyer Desk to unit(s) of host organization. From and To side revision rules are Float.	—
CAD SubComponent	This relationship captures the dependency that an assembly model has with its subassemblies and components. Attributes on this relationship are a convenient location to capture instance-specific information of the components and subassemblies in the context of the parent assembly.	Bounding Box Reference Designator Renamed From Spatial Location
Candidate Incident	Bug, Erratum, or Request is resolved in a software release. <i>Deprecated in version 10.0.1.0.</i>	Commitment Level
Candidate Item	Used to manage all the candidate features and requirements of a model.	—
Candidate Request	Indicates which software feature resolves a bug, request, or erratum. <i>Deprecated in version 10.0.1.0.</i>	—
Capability	Indicates the ability of an organization to perform a process. From side revision rule is Replicate. To side revision rule is Float.	Capability Status Process Qualification Status
Certification	Associate an organization or location to a certification. From and To side revision rules are Float.	—
Characteristic	Identifies characteristics for a technical specification.	—
Characteristic Specification	Relationship between Characteristic object and Specification object.	—
Child Markup	Lets users create markups to the BOM before submitting an Engineering Change Request.	Find Number Markup Connection ID Markup Function Quantity Usage
Claimed SCO	Relationship between Specification Office person and SCO, mainly used for the claiming purpose of the SCO.	—
Co-Dependent	Connects two Option Rule objects (every Option Rule object represents an option for a configurable product). Whenever one option is present, the other option is required.	Comment

Relationship	Description	Attribute
Collaboration Partner	Created when one company accepts an invitation from another company to collaborate. Connects the two companies. From and To side revision rules are Float.	Shared Types
Collaboration Request	Created when one company invites another company to collaborate. Connects the two companies. From and To side revision rules are Float.	Collaboration Request Type
Committed Item	Used to manage all the committed features and requirements of a model.	—
Company Currency Conversion	Identifies a company's Currency Conversions. To side revision and clone rules are Replicate.	—
Company Department	Identifies the departments within a company or business unit.	—
Company Documents	Connects Holder object to Document objects. This relationship is created when Document objects are checked in with package summary reports.	—
Company ECR	Identifies a company's ECRs. To side revision and clone rules are Replicate.	—
Company Financial Categories	Identifies Financial Cost Category and Financial Benefit Category objects for an organization.	—
Company Package	Identifies a company's packages. To side revision and clone rules are Replicate.	—
Company Product	Used to manage product by company. To side revision and clone rules are Replicate. <i><u>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</u></i>	—
Company Product Lines	Used to manage product lines by company. (A company's product line.)	—
Company Program	Connects programs with a company. A company can have many programs.	—
Company Project	A company must sponsor a project management object. Optionally and additionally, the project management can be sponsored by a Business Unit.	—
Company Project Templates	Groups project templates by Organization.	—
Company Report	Identifies a company's Download Package report.	—
Company RFQ	Identifies a company's RFQs. To side revision and clone rules are Replicate.	—

Relationship	Description	Attribute
Company Role	Assign roles at the company level. <i>Deprecated in version 10.5.</i>	—
Company Route Templates	Groups route templates by Organization.	—
Company Specifications Templates	Identifies specs to use as templates for a company. To side revision and clone rules are Replicate.	—
Company Teams	A company may owned many teams such as IT, Customer Service, Marketing, Finance. <i>Deprecated in version 10.5.</i>	—
Competitor	Associates two organizations as competitors. From and To side revision rules are Float.	—
Component Substitution	Connects a part to a substitute.	EBOM ID Quantity
Configuration File	Connects the Configuration File object to the respective Distribution File object. From side revision rule is Float.	—
Contains	Connects Adhoc Category to Documents Classification, Documents, or Adhoc Category, indicating that the Adhoc Category contains the attached object. <i>Deprecated in version 10.0.0.0.</i>	—
Context Assembly	Relationship created when an application context MEP is created, connects intermediate application parts to assembly in whose context they are created.	—
Context Organization	Relationship created when an application context MEP is created, from the location or organization to the application parts.	—
CoOwned	Identifies the CoOwned list for any object type. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—

Relationship	Description	Attribute
CoOwns	Identifies members of a CoOwned list. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Cost Item Interval	Connects cost items to interval item data.	Actual Cost Estimated Cost Planned Cost Interval Date
Currency Conversion Holder	Identifies the holder object for a company's currency conversions.	—
Custom Item	Identifies a custom configuration for a Configurable Product. Custom configurations are created by a customer and can be accessed only by users within the customer organization.	—
Customer	Indicates that an organization is a customer for a host organization. From and To side revision rules are Float.	—
Customer Design	Connects customer details about a Request for Quote; connects Request for Quotes to any object that contains details, such as data documents and parts. To side revision rule is Replicate.	—
Data Vaults	Identifies the data storage items for a workspace, workspace template, project space, project template, project concept. To side revision rule is Replicate. <i>Originally named Project Vaults and therefore its symbolic name is relationship_ProjectVaults. Named Workspace Vaults in AEF 9500 and then renamed to Data Vaults.</i> <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Default Line Item Attribute Group	Defines a default attribute group to be used for all existing and new line items in the RFQ. From side revision and clone rules are Replicate.	Attribute Filter
Default Line Item UDA Miscellaneous	Defines default miscellaneous UDAs for all existing and new line items in the RFQ. From side revision and clone rules are Replicate. Allows duplicates.	UDA Mandatory UDA Name

Relationship	Description	Attribute
Default Line Item UDA Non-recurring Cost	Defines default non-recurring cost UDAs for all existing and new line items in the RFQ. From side revision and clone rules are Replicate. Allows duplicates.	UDA Name
Default Line Item UDA Recurring Cost	Defines default recurring cost UDAs for all existing and new line items in the RFQ. From side revision and clone rules are Replicate. Allows duplicates.	UDA Name
Default Region	Identifies a person's default region.	—
Default Specification Office	Identifies a person's default spec office.	—
Deleted Subtask	Connects Project Management objects to Task Management objects.	Comments Date Deleted Sequence Order Task WBS
Deliverables	This software release gets implemented as part of this product release. From and To side revision rules are Float. <i>Deprecated in version 10.0.1.0.</i>	—
Department Employee	Identifies employees of a department. To side revision rule is Replicate.	—
Dependency	Defines task dependencies across a single project.	Lag Time
Derived Component	Relationship between any two MCAD Models, captures general dependencies among MCAD Models.	—
Derived Viewable	Connects Viewable types with Viewable types.	—
Design Responsibility	Indicates the responsibility of an organization, project space, or location for design of a part, CAD Drawing, CAD Model, Drawing Print, ECO, Features, Products, Builds, Requirement, Source Code, Directory, Technical Specification, or ECR. To side revision and clone rules are Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Agreed Unit Price Comments RTS ID Show Sub Components Show Target Cost Source Qualification Status Source Selection Status Sub Component Level
Development Manager	Identifies the person who manages a supplier. From and To side revision rules are Float.	—
Directed By	Connects a customer organization to directed supplier. From and To side revision rules are Float.	—

Relationship	Description	Attribute
Directed Item	Connects a directed supplier to item. From and To side revision rules are Float.	—
Directed To	Connects directed supplier to supplier organization. From and To side revision rules are Float.	—
Distribution Available On	Connects the Software Platform object to the respective Distribution File object. From side revision rule is Float.	—
Distribution File	The distribution file that will be ship with the product or build.	—
Distribution List	Identifies the distribution list for a technical spec.	—
Distribution Media	Connects the Media object to the respective Distribution File object. From side revision rule is Float.	—
Division	Indicates the business units within a company. From and To side revision rules are Float. Allow for Business Units to be N levels.	—
Document Holder	For Summary Reports and Multi-Level BOM reports, connects the Report Holder object to user's Company.	—
Document Sheets	Connects Document object to a Document Sheet indicating the document has sheets. To side clone rule is Replicate.	—
Document Structure	Connects document objects to other document objects to represent assemblies and subassemblies within 3D JT files.	—
Document Version	Relationship from Documents to Version Document <i>Deprecated in version 10.5.</i>	—
Duplicate	Relationship between a request, bug or, erratum to a duplicate. From and To side revision rules are Float. <i>Deprecated in version 10.0.1.0.</i>	—
EBOM	Engineering Bill of Material; indicates the component parts of an assembly part. From side revision and clone rules are Replicate.	Component Location End Effectivity Date Find Number Quantity Percent Concentration Reference Designator Start Effectivity Date Usage

Relationship	Description	Attribute
EBOM History	In an Engineering Bill of Material, indicates the component parts of an assembly part. Allows the system to keep track of old EBOM connections.	Component Location End Effectivity Date Find Number Quantity Percent Concentration Reference Designator Start Effectivity Date Usage
EBOM Markup	Connects a part to an EBOM Markup.	—
ECO Change Request Input	Indicates which ECRs are driving the ECO.	—
ECR Holder	Identifies the holder object for a company's ECRs.	—
ECR Main Product Affected	Connects the product line that is most affected to the ECR affecting it.	—
ECR Originator Company	Used to connect ECR to the creator's company. From side revision and clone rules are Replicate.	
ECR Reference	Identifies a supplier-suggested change to a supplier line item that may eventually be used to start a formal change against an engineering part or document.	—
ECR Supporting Document	Identifies objects that will support the ECR details, such as markups and sketches.	—
Effort	Connects any object type to an Effort object to track and estimate the effort.	—
Employee	List of employees in the organization. From and To side revision rules are Float. <i><u>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</u></i>	—
Employee Representative	Identifies the person who acts as the representative for a company. To side revision rule is Replicate.	—
eService Additional Object	Connects the Object Generator for a primary object to the Object Generator for an additional object.	eService Connect Relation
eService Number Generator	Connects a Object Generator to its Number Generator.	—
Family Member	Identifies the part family a part is a member of. <i><u>Deprecated in version 10.5.</u></i>	Sequence Order
Feature Breakdown	Indicates subfeatures for a software feature. From and To side revision rules are Float.	—

Relationship	Description	Attribute
Feature Design Specification	Design specification of the software feature, which serves as implementation reference for software feature. From and To side revision rules are Float.	—
Feature Functional Specification	Connects a functional specification to the software feature it describes. From and To side revision rules are Float.	—
Feature List	List of features for the attached product.	Sequence Order
Feature List From	Connects features and products. The intermediate object, Feature List, is used to ensure that each product instance has unique rules and feature rules. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Sequence Order
Feature List To	From side clone rule is Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Feature Rule	Connects the Feature Rule object to the corresponding Feature object.	—
Feature Task Breakdown	Connects a feature to its tasks. From and To side revision rule is Float.	—
Feature Test Case	Connects the test case that is used to validate the attached feature.	—
Feature Test Specification	Connects a test specification to its feature. From and To side revision rule is Float.	—
Feature Use Case	Connects a use case to a feature.	—
Features Specification	Describes the feature and its requirements.	—
Financial Items	Identifies cost and benefit items for a financial item.	—
Financial Sub Categories	Identifies subcategories of Financial Cost Category and Financial Benefit Category.	—
Fixed In	Connects the Incident object to the respective Software Build object. From side revision rule is Float. <i>Deprecated in version 10.0.1.0.</i>	—

Relationship	Description	Attribute
Folder Document	Identifies the items in a folder, which can be a document or other routed item, such as an RFQ quotation, RFQ, or package. To side revision rule is Float. <i>Deprecated in version 9.5.0.0.</i>	—
Found In	Incident found in software release or build. <i>Deprecated in version 10.0.1.0.</i>	—
GBOM	Specifies the parts to be included in the Generic Bill of Materials, and whether they are attached to options or parts.	Find Number Quantity
GBOM From	The Generic Bill of Material that points to the intermediate object, which contains part specific rules.	Find Number Quantity
GBOM To	The Generic Bill of Material that points to the appropriate part. From side clone rule is Replicate.	—
Governed By	The product that is affected by the incident. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Governing Project	The project responsible for completing the tasks. From side clone and revision rules are Replicate.	—
Has Books	Connects the books for a bookshelf. To side clone rule is Replicate.	—
Has Bookshelves	Connects the bookshelves for a library. To side clone rule is Replicate.	—
Has Documents	Connects the documents for a book. To side clone rule is Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Historical Version	Identifies the read only baseline versions created for a business object revision.	—
HTTP Request	Connects HTTPRequestHolder to HTTPRequest objects.	—
HTTP Request Holder	Connects person to HTTPRequestHolder object.	—

Relationship	Description	Attribute
Image	Describes the product, product line, model, etc. using marketing visuals. From side revision rule is Replicate.	Sequence Order
Impact Analysis	Impact analysis done for an incident. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Incident Document	Connects an incident to product documentation. <i>Deprecated in version 10.0.1.0.</i>	—
Incident Fixed In	The objects that were fixed in order to close the incident.	—
Incident Found In	The incident's origin.	—
Incident Owned By	Identifies the product that owns an incident. <i>Deprecated in version 10.0.1.0.</i>	—
Incident Submitted By	Identifies the organization that reported an incident.	—
Incident Validation	Test Case for an incident.	—
Includes Supplier Part	Indicates that a supplier and part are included on an RFQ. Connects an RFQ to an RFQ supplier part.	—
Incompatible	Connects two options that are incompatible—if one is included in a configuration the other cannot be included.	Comment
Initiated Template Project	A project space or workspace is derived from this template project.	—
Initiating Route Template	Connects a route to a route template.	—
Instance Of	Represents the instantiation of a specific instance of a parameterized model to the model. Note that an instance of a parameterized model can also be another parameterized model. Typically a parameterized model has multiple instances.	Instance Parameters
Internal Design	Connects employee details about a Request for Quote; connects Request for Quotes to any object that contains details, such as data documents and parts. To side revision rule is Replicate.	—
Issue	Indicates an issue for any object.	—
Issue Category Classification	Connects Issue Classifications to Issue Categories.	Issue Owned By
Items To Change	Connects from SCO to Technical Specification for issuance process of the specifications.	—

Relationship	Description	Attribute
Items To Obsolete	Connects from SCO to Technical Specification to obsolete the specifications.	—
Latest Version	Connects from DOCUMENTS to DOCUMENTS with Float on revision at To end.	—
Left Expression	Indicates the left expression of the compatibility rule for an optional feature. From side clone rule is Replicate.	—
Line Item	Identifies all line items the buyer wants to have suppliers quote on for an RFQ. To side revision rule is Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Annual Part Quantity Required Availability Unit of Measure
Line Item Object	Connects an RFQ supplier part to a part. From side revision and clone rules are Replicate. <i>Original name was RTS Part so the symbolic name is relationship_RTSPart.</i> <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Line Item Split	Identifies a volume split of a main line item by receiving plant. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Annual Part Quantity Receiving Plant Required Availability Unit of Measure
Line Item Supplier Exception	Identifies suppliers that should not bid on a line item. From side revision and clone rules are Replicate.	—
Link URL	Connects Project Management and Quality types to URL types. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
List Member	Identifies the people who belong to a member list.	—

Relationship	Description	Attribute
Local Template	Identifies the local template for a spec. To side revision and clone rules are Replicate.	—
Location Equivalent	Relationship created when location specific MEP is created, from enterprise part to location equivalent object.	—
Make Obsolete	ECO will obsolete the parts connected with this relationship.	Disposition (Field Return) Disposition (In Field) Disposition (In Process) Disposition (In Stock) Disposition (On Order) Specific Description of Change
Manual Section	Identifies a section within a Standards Manual. From and To side revision rules are Float.	Sequence Order
Manufacturer Equivalent	Connects enterprise parts or location equivalent objects with MEPs. From side revision rule is Replicate.	Start Effectivity Date End Effectivity Date
Manufacturer Equivalent History	Identifies equivalent parts. Tracks the history of Manufacturer Equivalent Parts. When an equivalent part is revised and released, the latest released and in-process equivalent relationships float to the new released part, and the latest released equivalent connections are changed to history connections. This is the same logic as is handled with the EBOM History relationship.	Start Effectivity Date End Effectivity Date
Manufacturing Responsibility	Indicates the responsibility of an organization or location for manufacture of a part or technical spec. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Agreed Unit Price Comments Production End Date Production Purpose Production Start Date RTS ID Show Sub Components Show Target Cost Show This Supplier Price to Other Suppliers Show This Supplier to Other Suppliers Source Qualification Status Source Selection Status Sub Component Level
Markup	Connects the markup object to the object that is marked up (for example, the drawing).	Markup Author Markup Base Filename Markup Base Format
MBOM	Creates a Manufacturing Bill of Material by linking parts to part masters. From side revision rule is Replicate.	Component Location Find Number Plant ID Quantity Reference Designator Usage

Relationship	Description	Attribute
Meeting Attachments	Identifies documents for a meeting. To side revision rule is Float. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Meeting Context	Defines the context in which the meeting takes place. For example, the context might be a specific project or software build.	Meeting Owner
Member	Connects project spaces, project concepts, and project templates to persons, organizations, companies, business units, and departments. Lets the project lead define the project members who have read access and can own subtasks.	Project Member Access Project Role
Member Folder	To side revision rule is Replicate. <i>Deprecated in version 9.5.0.0.</i>	—
Member Folder Access	Connects a project member to a folder to allow the member access. To side revision rule is Replicate. <i>Deprecated in version 9.5.0.0.</i>	Folder Access
Member List	Identifies member lists for an organization or person.	—
Member Route	Identifies the workspace member who created the route. To side revision rule is Replicate.	—
Member Task	Identifies the project member who created the route to which the task belongs. To side revision rule is Replicate.	—
Member Vault Access	Assigns a project member or organization access to a vault.	Vault Access
Message	Connects a message to a thread. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Access Type

Relationship	Description	Attribute
Message Attachments	Identifies documents for a discussion message. To side revision rule is Float. <u><i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i></u>	—
Model Product	Used to manage all the products in a model. To side clone and revision rules are Replicate.	—
New Part/Part Revision	ECO will release the parts connected with this relationship.	Disposition (Field Return) Disposition (In Field) Disposition (In Process) Disposition (In Stock) Disposition (On Order) Specific Description of Change Where Used Component Reference
New Specification/ Specification Revision	ECO will release the CAD drawing, CAD model, or drawing print connected with this relationship.	Disposition (Field Return) Disposition (In Field) Disposition (In Process) Disposition (In Stock) Disposition (On Order) Specific Description of Change
Object Route	Identifies an object to be routed or to be included in a route template. To side revision rule is Replicate.	Route Base Policy Route Base Purpose Route Base State
Option List	List of options for indicated feature.	Sequence Order
Option Rule	Connects the Option Rule object to the corresponding Option object.	—
Ordered Item	List of items ordered for a configuration.	Order Quantity Unit Price
Organization Context	Connects application parts to a location <u><i>Deprecated in version 10.5.</i></u>	—
Organization Data Document	Connects data documents to organizations.	—
Organization Goal	The organization responsible for this goal.	—
Organization List Item	Identifies the Organization objects assigned to an Organization List. From side revision and clone rules are Replicate.	—

Relationship	Description	Attribute
Organization Location	Connects a location to its organization (company or business unit). From and To side revision rules are Float.	Billing Address Headquarters Site Manufacturing Site Shipping Address
Organization Region	Identifies regions for an organization.	—
Organization Representative	Indicates employees of a company, business unit, or department who are allowed to maintain the company's profile and add users. To side revision rule is Replicate. <i><u>This relationship was originally named Company Representative and therefore its symbolic name is relationship_CompanyRepresentative.</u></i>	—
Organization Specification Office	Identifies the spec offices for an organization.	—
Organization Template	Identifies the templates created for a company, for example RFQ templates, line item templates, and workspace templates. To side revision rule is Replicate. <i><u>This relationship was originally named Organization RTS Template and therefore its symbolic name is relationship_OrganizationRTSTemplate.</u></i>	—
Originating Requirement	The master document containing all the customer requirements.	Commitment Level
Originating Specification Template	Identifies the originating template for a spec. To side revision and clone rules are Replicate.	—
Owning Organization	Identifies the organization that owns a route template.	—
Package Holder	Identifies the holder object for a company's packages.	—
Package RFQ	Identifies the latest revisions of all Request To Supplier business objects that are grouped in the package. To side revision rule is Float. <i><u>This relationship was originally named Package RTS and therefore its symbolic name is relationship_PackageRTS.</u></i>	—
Packing Report	Links one or many Document objects to one Report Holder object to download a package of files. <i><u>Deprecated in version 10.5.</u></i>	—
Parent Markup	Lets users create markups to the BOM before submitting an Engineering Change Request. This relationship describes all the current markups against a specific assembly.	—
Part Family Member	Identifies the parts in a part family.	—

Relationship	Description	Attribute
Part Family Reference Document	Identifies the documents and specifications for a part family. To side revision rule is Float.	—
Part Request	Connects a manufacturing assessment or quotation to a part. Requests that a supplier quote a particular part.	—
Part Revision	Connects part masters to parts.	Effectivity Date Lot Effectivity Plant ID Serial Effectivity
Part Rule	Connects a Part Rule object to its Part object.	—
Part Specification	Connects objects that specify information about a part, such as CAD drawings, CAD models, and drawing prints. Prevent duplicates is turned on.	—
Pass Through Line Item	Connects Supplier Line Item and Line Item.	—
Pass Through RFQ	Identifies the pass-through RFQs for a quotation. Connects RFQ Quotation and RFQ objects.	—
Pending Line Item Attribute Group	Temporary relationship used to indicate that an Attribute Group is intended for a Supplier Line Item that has not been created yet. From side revision and clone rules are Replicate.	Attribute Filter Default Selection
Pending Version	Identifies the version of a business object revision created to make changes that should not go in effect for the main revision yet.	—
Plan Deliverable	Associates deliverables to a plan. From and To side revision rules are Float.	Actual Completion Date Scheduled Completion Date Sequence Order Submitted Date
Plan Focus	Associates an organization to its quality plan(s). From and To side revision rules are Float.	—
Plan Location	Associates a Part Quality Plan to a location. From and To side revision rules are Float.	—
Plan Manager	Associate a person to manage a plan. From and To side revision rules are Float.	—
Planned Feature	Represents a feature planned for a software release. From and To side revision rules are Float.	Commitment Level
Precise BOM	Generates the Precise BOM. The top-level part number assigned to the configuration will be related to the Product Configuration through the intermediate GBOM object. The relationship Precise BOM will connect the Product Configuration to the GBOM object. All components generated will be related using the EBOM relationship.	Find Number Quantity
Primary Image	Indicates the primary image for a product line, product, model, or feature. From side clone and revision rules are Replicate.	—

Relationship	Description	Attribute
Primary Process	Connects a part to a process.	—
Product Build	The delivery product for all builds.	—
Product Configuration	Connects a product to each of its product configurations.	Comment
Product Configuration Build	Relates a product configuration and a build.	
Product Line Models	Relates product lines to its models. A product line can have one or many models.	—
Product Requirement	Identifies the user requirements for a product.	—
Product Specification	Identifies the technical specs for a product.	—
Product Version	Provides visibility for all the versions of a product.	Sequence Order
Products	Connects products and models.	—
Program Project	Identifies the project management objects in a program.	—
Project Access Key	Connects Project Access Lists to Project Management, Risk, Financials, Quality, and Assessment objects.	—
Project Access List	Connects Project Access Lists to Project Concept, Project Space, and Project Template objects.	—
Project Assessment	Identifies assessments for a project space.	—
Project Financial Item	Identifies financial items for a project space. Prevent duplicates is turned on.	—
Project Folders	<u>Deprecated in version 9.5.0.0.</u>	—
Project Members	Identifies the project members for a project space, workspace, or project concept. To side revision rule is Replicate. <u><i>This relationship is used in an expression access so if its name changes, the expression must change on the policy. See Administrative Objects Used in Expression (Filter) Accesses.</i></u>	—
Project Membership	Identifies a person who is a project member. A project can be associated to a person or a predefined group of people. <u><i>This relationship is used in an expression access so if its name changes, the expression must change on the policy. See Administrative Objects Used in Expression (Filter) Accesses.</i></u>	—
Project Question	Identifies questions for a Template Project.	—
Project Release	Connects a project to a software release. To side revision rule is Float.	—
Project Requirement	Requirement managed by Project.	—

Relationship	Description	Attribute
Project Route	Identifies the route creator. To side revision rule is Replicate.	—
Project Route Templates	Identifies the route templates for a project space.	Route Access
Project Task	Identifies the person or route task user who must complete the task. To side revision rule is Replicate.	—
Proposed Markup	Lets users associate markup objects to an ECR to start the ECR process.	—
Publish	Connects the event object for the event being subscribed to with the Publish Subscribe object.	—
Publish Subscribe	Connects the object being subscribed to with its Publish Subscribe object. From side revision rule is Float.	—
Purchase Class	Links Plant Scorecard to a Purchase Class. From and To side revision rules are Float.	Supplier Classification
Purchase Class Manager	Associate a person to manage a purchase class. From and To side revision rules are Float.	—
Purchase Class Member	Associate a purchase class to its member parts. From and To side revision rules are Float.	—
Purchase Subclass	Creates hierarchy of purchase classifications. From and To side revision rules are Float.	—
Pushed Subscription	Indicates the Subscription Events that one user has set for another user. Currently, it is assumed that the subscribed object's owner sets all pushed subscriptions.	—
Quality	Identifies quality objects for a project space.	—
Quality Metric	Connects quality objects to quality metric objects that measure the quality.	Comments Defects Per Million Opportunities Defects Per Unit Lower Specification Limit Mean Metric Source Originator Sigma Standard Deviation Title Upper Specification Limit
Quality Plan Part	Associates a Part Quality Plan to a part. From and To side revision rules are Float.	—
Question	Projects will be created by a wizard based screen that will allow the user to answer questions that will decide the tasks that must be added to the project (Survey Templates)	Task Transfer

Relationship	Description	Attribute
Quote	Connects Requests for Quotes to the quotation object.	—
Raised Against ECR	Connects the ECR to the original Part/Revision or Specification/revision. Hidden. Prevent duplicates is turned on.	—
Rate Period	Defines conversion rates at a given start effectivity date. Allows duplicates.	Effectivity Date Rate
Recommended Suppliers	Connects companies to technical specs. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Reference Document	Lets people attach supporting documentation—such as a process specification or test plan—to a business object, such as a part, part family, project, ECR, line item, RFQ, quotation, risk, assessment, financial item, quality, business goal, technical spec, etc. These documents may or may not require change control. From side clone and revision rules are Replicate. Prevent duplicates is turned on. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Document Classification
Region Owns	Identifies the company region that owns technical specs and SCOs. To side revision and clone rules are Replicate.	—
Regional Template	Identifies the regional template for a spec. To side revision and clone rules are Replicate.	—
Related Concept Project	A project space supports this concept project.	—
Replaces Part	Not currently used.	Replacement Scope
Reply	Identifies a message that is a reply to another message. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Reporting Organization	Identifies Issues for an organization.	—

Relationship	Description	Attribute
Request for Quote	Connects Requests for Quotes submitted by a customer to the organization the customer belongs to.	—
Request Impact Analysis	Impact Analysis done for a request. To side revision rule is Float. <i>Deprecated in version 10.0.1.0.</i>	—
Request Part Obsolescence	ECR is requesting obsolescence of the parts connected with this relationship. Prevent duplicates is turned on.	Disposition (Field Return) Disposition (In Field) Disposition (In Process) Disposition (In Stock) Disposition (On Order) Specific Description of Change
Request Part Revision	ECR is requesting a revision to the parts connected with this relationship. Prevent duplicates is turned on.	Disposition (Field Return) Disposition (In Field) Disposition (In Process) Disposition (In Stock) Disposition (On Order) Specific Description of Change Where Used Component Reference
Request Specification Revision	ECR is requesting a revision to the CAD drawings, CAD models, or drawing prints connected with this relationship. Prevent duplicates is turned on.	Specific Description of Change
Requirement Baseline	Connect Requirements and Requirement Baselines.	—
Requirement Group	Connect Requirements and Requirement Groups.	—
Requirement Satisfied By	Identifies the software feature that satisfies a user requirement.	—
Requirement Specification	Connects requirement specifications to a requirement. The specification can be in the form of a specification, document, product documentation, or process.	—
Requirement Use Case	The decomposed use cases for the requirement.	—
Requirement Validation	The test case that is used to validate the attached user requirement.	—
Requires	Connects two options where one option must be present if the other is included but not vice versa.	Comment
Resolved To	Connects Issues to any of the following: Issue Incident, Task, Document, Risk, Requirement, Use Case, Test Case, Organization, Bug, Request, Erratum, Duplicate, Defect, User Requirement, Company, Business Unit, Department.	—

Relationship	Description	Attribute
Resource	Points to a resource used by an option. <i>Deprecated in 10.5.</i>	Comment Resource Operation Resource
Resource Limit	Points to resource that is limited for a product.	—
Resource Usage	Points to a fixed resource used by a feature list, option rule.	—
Responsible Sales Engineer	Identifies the Sales Engineer who is the point person for a supplier company.	—
RFQ for Package Quotation	Identifies the last revision of the Request To Supplier business object used to collectively quote all line items of all other Request To Supplier business objects in the package. To side revision rule is Float. <i>This relationship was originally named RTS for Package Quotation and therefore its symbolic name is relationship_RTSforPackageQuotation.</i>	—
RFQ Holder	Identifies the holder object for a company's RFQs.	—
RFQ Line Item Template	Connects a line item template to a specific RFQ template. From side revision rule is Replicate.	Attribute Filter
RFQ Quotation	Identifies each supplier's RFQ Quotation business object for a given RFQ business object. <i>This relationship was originally named RTS Quotation and therefore its symbolic name is relationship_RTSQuotation.</i>	Award Status Percentage
RFQ Supplier	Connects an RFQ supplier part or RFQ to an organization. From side revision and clone rules are Replicate. <i>This relationship was originally named RTS Supplier and therefore its symbolic name is relationship_RTSSupplier.</i>	Recipient
RFQ Template	Defines the template chosen while creating the RFQ business object. From side revision rule is Replicate. <i>This relationship was originally named RTS Template and therefore its symbolic name is relationship_RTSTemplate.</i>	—
RFQ Template Buyer Desk Association	Identifies the Buyer Desks assigned to an RFQ template. To side revision rule is Replicate.	—
Right Expression	Indicates the right expression of the compatibility rule for an optional feature. From side clone rule is Replicate.	—

Relationship	Description	Attribute
Risk	Identifies risks for a project management, software feature, or software release. <u><i>This relationship is used in an expression access so if its name changes, the expression must change on the policy. See Administrative Objects Used in Expression (Filter) Accesses.</i></u>	—
Risk Affected Items	Connects Risks to Task Management items. <u><i>Deprecated in version 10.5.</i></u>	—
Risk Item	Connects Risks to Task Management items.	—
Risk RPN	Indicates the RPN value stored for the risk. A risk can have many RPN.	Effective Date Originator Risk Impact Risk Probability Risk RPN Value Status
Role Person	A person can belong to many roles and a role can have many persons. Duplicates not allowed. <u><i>Deprecated in version 10.5.</i></u>	—
Route Node	Identifies the person or route task user (role, group, or unassigned) associated with each node in routes and route templates. To side revision rule is Replicate. From and To side clone rules are Replicate.	Actual Completion Date Allow Delegation Approval Status Approver's Responsibility Assignee Set Due Date Comments Date Offset From Due Date Offset Parallel Node Procession Rule Reviewers Comments Review Comments Needed Review Task Route Action Route Branch Route Instructions Route Node ID Route Sequence Route Task User Route Task User Company Scheduled Completion Date Task Requirement Template Task Title
Route Scope	Establishes the data organization level from which a route's content and participants can be selected.	—

Relationship	Description	Attribute
Route Task	The tasks for a route. To side revision rule is Replicate.	—
Route Templates	Identifies the route templates available for a company, person, or workspace. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
RTS Removed Supplier	Identifies suppliers that are not bidding on the current round, but did bid on the previous round. From side revision and clone rules are Replicate.	—
Rule Extension	Indicates an extension rule for a product.	—
Sales Order	Connects sales orders to the organization that placed them.	—
Score By	Relationship from Score By object to a Scorecard. From and To side revision rules are Float.	—
Score For	Relationship from Score For object to a Scorecard. From and To side revision rules are Float.	—
Selected Options	Decomposed list of selected features and options after a configuration.	List Price Quantity
Selection Criteria	Connects Template Filter to Business Unit, Region, or Specification Office.	
Ships To	The location a location ships to. From and To side revision rules are Float.	—
Similar Part	Not currently used.	—
Software Architecture	The architecture of the software, which serves as a reference while implementing the software. From and To side revision rules are Float.	—
Software Baseline	The baseline of the software release at a particular point of time in its development cycle. From and To side revision rules are Float.	—
Software Build	Connects Software Build object to the respective Software Release object. From side revision rule is Float.	—
Software Dependency	This software release gets implemented as part of ECO release. From and To side revision rules are Float.	—
Software Distribution	Connects the Software Distribution object to the respective Software Build object. From side revision rule is Float.	—

Relationship	Description	Attribute
Software Requirement Specification	The software requirement specification for the software, which serves as a startup document for the rest of the specifications of the software. From and To side revision rules are Float. <i>Deprecated in version 10.0.1.0.</i>	—
Source File	The source code files.	—
Spare Part	Identifies spares for parts. From side revision rule is Replicate. Prevent Duplicates is turned on.	End Effectivity Quantity Start Effectivity
Specification Section	Identifies the sections of a specification. From side revision and clone rules are Replicate.	—
Specification Template	Connects from Specification to Technical Specification Template to identify the parent for Specification.	—
Standard Component	Identifies a substitute for a part.	—
Standard Item	Identifies standard configurations for a Configurable Product. Standard configurations are configurations created by the host company that should be made available to multiple customer organizations.	—
Sub Directories	The source code directory structure.	—
Sub Goal	The sub goals of a given business goal.	—
Sub Product Lines	The sub product lines for a product line.	—
Sub Region	Identifies sub-regions within organization regions.	—
Sub Requirement	Indicates subrequirements that make up a user requirement. <i>This relationship was originally named Requirement Breakdown and therefore its symbolic name is relationship_RequirementBreakdown.</i>	—
Sub Route	Identifies a route within a route. To side revision rule is Replicate.	Parent Route Node
Sub Test Case	The decomposed test cases.	—
Sub Use Case	The decomposed use cases.	—
Sub Vaults	Identifies a category within a category. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—

Relationship	Description	Attribute
Subscribed Item	Connects the event object for the subscription to the object subscribed to.	—
Subscribed Person	Connects the event object for the subscription to the person who is subscribing.	—
Subsidiary	Indicates that one company is a subsidiary of another company. From and To side revision rules are Float.	—
Substitute	Lets users substitute preferred parts in the Manufacturing Bill of Material.	Parent Part ID
Substituted Component	Identifies a part that has been substituted.	—
Subtask	Builds a project structure (Project Schedule) by connecting projects and project tasks to other project tasks.	Project Task WBS Sequence Order
Supplier	Indicates that one organization is a supplier for another organization. From and To side revision rules are Float.	—
Supplier Buyer Line Item	Used just to navigate between the buyer-specified line item information and the supplier's line item response. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Supplier Line Item	Identifies the supplier's response to all line items the buyer wants quoted. To side revision rule is Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Annual Part Quantity Currency Effectivity Date No Bid Reason Supplier Response Unit of Measure Unit Price
Supplier Line Item Price Period	Stores additional price periods for a supplier line item response. Allows duplicates.	Currency Unit Price Effectivity Date
Supplier Part Request	Connects a quotation or manufacturing assessment to an RFQ supplier part. Indicates which part/supplier pair the quote is for.	—
Supplier Response	Connects a quotation or RFQ quotation to an organization. From side revision rule is Replicate. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—

Relationship	Description	Attribute
Supplies	Connects an organization or location to the Purchase Class(es) it supplies. From and To side revision rules are Float.	Supplier Classification
Supply Responsibility	Connects an organization to a part or technical spec to indicate the supplier and the supplied part/spec. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Agreed Unit Price Comments RTS ID Show Sub Components Show Target Cost Source Qualification Status Source Selection Status Sub Component Level
Supported File Format	Identifies the kind of file formats used by an organization. From and To side revision rules are Float.	—
Supported Organization	Connects specification offices to business units.	—
Task	Tasks associated with a risk or software release.	—
Task Deliverable	Lets project leader and assigned owners attach parts, documents, or any other object to a task.	End State Start State
Task Escalation Message	Defines how task escalation messages are to be determined. Allow Duplicates is turned on.	Day Delta Hours Delta Recipient
Task Sub Route	Identifies subroutes created for a task.	—
Team Membership	A team has a set of person members. <i>Deprecated in version 10.5.</i>	—
Technical Specification	Connects one specification to other reference specification. This is a self-relationship.	—
Template Filter	Connects a route template to a template filter.	—
Testing Responsibility	Indicates the responsibility of an organization for testing of a part or technical spec. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Agreed Unit Price Comments RTS ID Show Sub Components Show Target Cost Source Qualification Status Source Selection Status Sub Component Level

Relationship	Description	Attribute
Thread	Connects the object being discussed to the thread object that represents all the messages about the object. For example, if a discussion is about a document, the Thread relationship would connect the document object to the thread object. From side revision rule is Float. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
UDA Miscellaneous	Stores miscellaneous user-defined attribute information for an RFQ, line item, supplier line item, package, or technical spec. Allows duplicates.	Default Selection UDA Mandatory UDA Name Value (String)
UDA Non-Recurring Cost	Stores non-recurring cost user-defined attribute information. Allows duplicates.	Currency Default Selection UDA Name Value (Real)
UDA Recurring Cost	Stores recurring cost user-defined attribute information. Allows duplicates.	Currency Default Selection UDA Name Value (Real)
Use Case Validation	The test case that is used to validate the attached use case.	—
User Requirement	The customer or internal requirement that gets implemented as part of the software release. From and To side revision rules are Float.	Commitment Level
Vaulted Documents Rev2	Copy of Vaulted Documents, which has been renamed Vaulted Objects. Modification of the original to allow for revisions/versions. Identifies the items in a folder: documents, document classification and documents. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—

Relationship	Description	Attribute
Vaulted Objects	Identifies the items in a folder. Any type of object can be added to a folder. To side revision rule is float. <i>This relationship was originally named Vaulted Documents and therefore its symbolic name is relationship_VaultedDocuments.</i> <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Version	A master document or document sheet maintains versions. This relationship connects Product Documentation, Specification, Process, Documents, User Manual objects to Version Document objects. Revisions tracked using Matrix revision history. Versions tracked using the version relationships. This allows documents to be versioned independent of their revision, e.g., A,A:1,A:2,A:3,A:4 B,B:1,B:2,B3... Each revision of the document can continue to be versioned independently until the document is “released.” <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	—
Viewable	This relationship captures the derivation of a viewable object from its CAD Model or Drawing. There can be multiple viewables associated with a CAD Model. The assumption is that a given viewable has only one CAD Model associated with it.	—
Viewable Subcomponent	This relationship captures the structural dependencies between 3D viewables, much as CAD Subcomponent captures the relationships between assemblies and components.	Bounding Box Reference Designator Renamed From Spatial Location
Work Place	Identifies the company location where a person works. From side revision rule is Float.	—
Workspace Buyer Desk	Associates a Buyer Desk with a Team Central workspace.	—
Workspace Template	Indicates the template used to create a workspace.	—
Workspace Template Member	Identifies the members who belong to a workspace template.	Project Access Create Route Folder Access

Roles

If the role description does not specify an application that uses the role, then the role is used by more than one application. For an illustration of the type hierarchy for roles, see [Role Type Inheritance](#) in Chapter 3.

Role	Description
Access Grantor	Used for granting access to objects, hidden. A custom-code role for granting access privileges. Access grantor persons have the password “shadowsecret”.
Accounting Clerk	Used in Product Central. Receives payment for sales orders and then promote sales orders to the Payment Received state.
Administration Manager	Persons must belong to this role to use the administrative wizards in Matrix Navigator and to use the Reload Cache tool.
Advanced Quality Engineer	Used in Supplier Central. Responsible for Quality Plan Creation and Management Process.
Author	Used in Document Central. Can create, revise, modify or delete objects as determined by the AuthorCreateTypesList configuration settings.
Business Manager	Used in Program Central. Administrators of business goals. Business goals feature is only available to host company users. Child of Project User.
Buyer	Used in the Supplier Central. Employees or representatives of a buyer company.
Buyer Administrator	Used in Sourcing Central. Creates RFQ and line item templates.
Company Contact	Main contact for a company.
Company Representative	Used in several applications. Can access the Administration pages to edit their company and employee profiles.
Component Engineer	Used in Engineering Central. Maintains manufacturer equivalent parts. Child of Employee. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Customer	Used in Product Central. Users in the customer company who use the Product Configurator Web pages to configure and order products. Parent of Customer Representative.
Customer Representative	Used in several applications. Users in the customer or supplier company who will request for users to be added.

Role	Description
Design Engineer	Used in Engineering Central and Product Central. Its child is the Senior Design Engineer role. In Engineering Central, user who creates and revises parts, drawing prints, and ECR/ECOs. In Product Central, user who creates rules for a product (such as compatibility rules, inclusion rules, rule extensions, and resource), adds parts, and reviews and tests the product before it is released. People who need to test the Configure Product feature in Product Central should be assigned to the Product Manager or Design Engineer role. Persons with either of these roles can create configurations for products in any state, not just those that are released. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Document Control	Used in Document Central.
Document Control Manager	Used in Document Central.
Document Management Administrator	Used in Document Central. Administrative users who can set configuration and system settings using the configuration tool.
ECR Chairman	Used in Engineering Central. Each Change Board group must have exactly one ECR Chairman and one ECR Chairman can belong to many ECR Change Boards. As an ECR enters the Review state, an action fires that automatically routes the ECR to the person in the company that satisfies the following criteria: – The person must be a member of the group indicated in the Change Board attribute on the Product Line object that is attached to the ECR being promoted, AND; – The person must also have the ECR Chairman role. The system is thus performing a dynamic group/role intersection. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
ECR Coordinator	Used in Engineering Central. Each Change Board group must have exactly one ECR Coordinator and one ECR Coordinator can belong to many ECR Change Boards. As an ECR enters the Submit state, an action fires that automatically routes the ECR to the person in the company that satisfies the following criteria: – The person must be a member of the group which is indicated in the Change Board attribute on the Product Line object that is attached to the ECR being promoted, AND; – The person must also have the ECR Coordinator role. The system is thus performing a dynamic group/role intersection. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
ECR Evaluator	Used in Engineering Central. Evaluates an ECR before it is sent to the ECR Chairman.

Role	Description
Employee	Used in many suites. Parent of most roles. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Exchange User	Used in Team Central. All employees added to Team Central are assigned the role Exchange User. Parent of Company Representative.
External Program Lead	Used in Program Central. Manages programs for a given organization. Separates external users from Employee. Child of External Project User.
External Project Administrator	Used in Program Central. Manages project templates for a given organization. Separates external users from Employee. Child of External Project User.
External Project Lead	Used in Program Central. Creates projects. Separates external users from Employee. Child of External Project User.
External Project User	Used in Program Central. Enables Program Central application for non-host company employees. Child of Exchange User.
Financial Category Administrator	Used in Program Central. Administers the Financial Template Categories.
Financial Reviewer	Used in Program Central. Hidden, not currently used. Maintains and manages project budgets.
Global Template Administrator	Used in Specification Central. Defines and manages Global Technical Specification Templates. Child of Specification User.
Global User	Used in Team Central, hidden. All users are assigned to this role in order to do “global” access grants. This is the top-level role in the role hierarchy and is the parent of Employee, Supplier, Customer, and Exchange User.
Incident Entry Clerk	Used in Product Central. Enters incident data into the system, including bugs, erratas, requests, and duplicates.
Issue Manager	Used in Product Central. Connects issues to objects and assigns users to look at the issue.
Limited Author	Used in Document Central. Creates, revises, modifies, or deletes a limited set of business types as determined by the LimitedAuthorCreateTypesList configuration settings.
Manufacturing Engineer	Used in Engineering Central. Reviews, plans, and promotes parts before they are released. Its child role is Senior Manufacturing Engineer. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Marketing Manager	Used in Product Central. Defines the marking definition for products, enters the Marketing Name, Text and images for all product content.
Order Entry Clerk	Used in Product Central. Validates product configurations (if the process is not automated) and generates the precise BOM for product configurations. Order Entry Clerks also may enter the order into the MRP system and enter the promised delivery date for a sales order, as assigned by the MRP system.

Role	Description
Organization Manager	Used in several applications. Adds business units, companies, and persons. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Part Family Coordinator	Used in Engineering Central. Can create, modify, and delete Part Families.
Plant Representative	Used in Supplier Central. Host Company Plant location personnel.
Product Manager	Used in Product Central. Can add features and options for a product, review and test the product structure and rules, sign off on a product before it is released, update pricing for options, and determine when products and product configurations should no longer be offered for sale. People who need to test the Configure Product feature in Product Central should be assigned to the Product Manager or Design Engineer role. Persons with either of these roles can create configurations for products in any state, not just those that are released.
Product Obsolescence Manager	Used in several applications. Promote parts to the Obsolete state and reinstates obsolete parts. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Production Support Clerk	Used in Product Central. Monitors sales orders as they move through the manufacturing system and promotes sale orders to the Shipping state.
Program Lead	Used in Program Central. Child of Project User.
Project Administrator	Used in Program Central. Creates project templates. Child of Project User.
Project Assessor	Used in Program Central. Hidden, not currently used. Child of Project User.
Project Lead	Used in Program Central. Primary person responsible for a project space. Child of Project User.
Project User	Used in Program Central. Creates project concepts. Parent role for Project Lead, Project Administrator, Program Lead, Financial Category Administrator, Financial Reviewer, Project Assessor.
Quality Administrator	Used in Supplier Central. Manages business goals.
Quality Manager	Used in Supplier Central. Creates Quality Plan Templates and manages processes.
Release Manager	Used in Document Central. Owns all objects in Released state.
Requirement Manager	Used in Product Central. Manages the requirements structure.
Reviewer	Used in Document Central. Reviews documents (access documents in Review state), grants signatures for approval, etc.
Sales Engineer	Used in Sourcing Central. Receives Request for Quotes (RFQs), submits quotations, and handles issues related to RFQs.

Role	Description
Senior Design Engineer	Used in Engineering Central. Performs all of the tasks related to the parent role, Design Engineer, plus has the added responsibility of approving parts, drawing prints, and ECR/ECOs. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Senior Manufacturing Engineer	Used in Engineering Central. Performs all of the tasks related to the parent role, Manufacturing Engineer, plus has the added responsibility of approving parts before they are released.
Senior Quality Engineer	Used in Product Central. Person belonging to the Software Quality Assurance team who ensures that output from the Software Engineering group (software functionality, bug fixes, and documentation) is of the desired quality levels. Has the authority to promote tested features and incidents.
Shipping Clerk	Used in Product Central. Ships ordered items to customers and promotes sales orders to the Shipped state.
Software CCB Coordinator	Used in Product Central. Heads the Change Control Board and coordinates the meetings.
Software Engineer	Used in Product Central. User on the software development team.
Software Engineering Manager	Used in Product Central. Responsible for an engineering team and creates Software Feature objects.
Software Project Manager	Used in Product Central. Responsible for a software project.
Software Quality Engineer	Used in Product Central. User who is on the software quality team and who ensures that the output is of the desired quality levels.
Software Quality Manager	Used in Product Central. Heads the software quality team.
Software Release Engineer	Used in Product Central. Creates Distribution File objects.
Software Release Manager	Used in Product Central. Creates and approves Distribution File objects.
Specification Manager	Used in Specification Central. Defines new Technical Specifications. Child of Specification User.
Specification Office Manager	Used in Specification Central. Manages SCO and Technical Specifications. Child of Specification User.
Specification Restricted User	Used in Specification Central. Manages highly-restricted specifications. Child of Specification User.
Specification User	Used in Specification Central. Parent of Specification Manager, Specification Office Manager, Template Administrator, Global Template Administrator, Specification Restricted User.
Supplier	Used in Supplier Central. Employees or representatives of a supplier company.
Supplier Development Manager	Used in Supplier Central. Manages Supplier Development process.

Role	Description
Supplier Engineer	Used in Supplier Central. Supplier who performs engineering activities. Child of Supplier role. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
Supplier Quality Engineer	Used in Supplier Central. Performs WBS Tasks assigned by Supplier Representative.
Supplier Quote Analyst	Used in Supplier Central. Required role for Sourcing RFQ notifications
Supplier Representative	Used in Supplier Central. Employees or representatives of a supplier company who request for users to be added. Child of Supplier role. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>
System Conversion Manager	Used in all applications. Has special permission to override and promote data in order to convert it from a legacy system.
System Engineer	Used in Product Central. Defines all product content.
System Transition Manager	Used in all applications. Has special permission to correct data resulting from legacy system conversion.
Technical Writer	Used in Product Central. Corrects documentation errors, as tracked in Erratum objects.
Template Administrator	Used in Specification Central. Defines and manages Technical Specification Templates. Child of Specification User
Template Header Administrator	Used in Specification Central. Manages the Global Header section attributes. Child of Specification User.

Stores

Store	Description
STORE	Use this store for testing the generic and viewer formats along with the file checkin, checkout, edit, and view commands. You may need to adjust these definitions by editing the schema scripts to include pertinent data from the database in which it will be installed and pertinent path and host data.

Types

A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z

Business Object Type	Description	Assigned Attributes
2D Viewable	A kind of viewable that has only 2D geometry. Examples are Gifs, Jpegs and so on. Each such viewable is typically viewable as a standalone document. Child of Viewable.	All attributes inherited from Viewable.
3D Viewable	A kind of viewable that has 3D geometry. Examples are JT, IGES, VRML, and STL files. Some of these viewables (corresponding to and derived from assembly models) depend on some of the other viewables in order to be visualized. Child of Viewable.	All attributes inherited from Viewable.
ADHOC CATEGORY	The abstract type that represents a place where users reference their objects. <i>Deprecated in version 10.0.0.0.</i>	Originator
Application Part	Represents application context MEP, used as an intermediate object when an application context MEP is created.	Inherits attributes from part, except Spare Part attribute.
Architecture	The architecture of the software, which serves as a reference while implementing the software.	Notes Originator Synopsis
Assembly Technical Specifications	An instantiable type that can also be sub-typed further by customers depending on the products the customer has available. Child of Technical Specifications	Attributes inherited from Technical Specifications
Assessment	Assessment of the current health of a project based on an assessor's evaluation.	Assessment Comments Assessment Status Resource Status Resource Comments Schedule Status Schedule Comments Finance Status Finance Comments Risk Status Risk Comments Originator
Attribute Group	Abstract type that is sub-typed by host companies to define logical groups of attributes to be filled in together as an extension of the standard attributes for a type.	Attribute Filter Primary Key Secondary Keys

Business Object Type	Description	Assigned Attributes
Baseline	Tracks the user requirements and features that will be delivered by a software release. There can be multiple baselines, with added or deleted features or requirements as the release is being developed. Differences between baselines can be evaluated.	Baseline Text Notes Originator Synopsis
Baseline Log	Maintains a log of baseline revisions.	Originator
Benefit Item	The Benefit Cost for a specific Financial Item in a Ledger Account. Child of Financials	All attributes inherited from Financials plus: Actual Benefit Estimated Benefit Ledger Account Number Planned Benefit
Bolt Part	Child of Fastener Part.	All attributes inherited from Part.
Book	Used for grouping together documents.	Access Type Created On Originator
Bookshelf	A collection of books.	Access Type Created On Originator
Boolean Compatibility Rule	A rule for a configurable product that specifies which feature/option pairs cannot be included together and which pairs must be included together based on a comparison operator and the value of two expressions. The comparison operator—which can be incompatible, co-dependent, or requires—determines how the two expressions are interpreted relative to each other. The left and right expressions can be any valid Boolean expression that includes feature/option pairs as the operands.	Comment Comparison Operator Contextual Error Left Expression Originator Right Expression
Bracket Part	Child of Mechanical Part.	All attributes inherited from Part.
Budgetary Estimate	A document from a supplier indicating the approximate cost of an item. Child of Quotation.	All attributes inherited from Quotation.
Bug	An incident that is determined to be an error in the software code. Child of Incident.	All attributes inherited from Incident.
Build Specification	Describes the build-related specifications. Child of Product Specification.	All attributes inherited from Product Specification.
Builds	Abstract Describes the Build type hierarchy.	Actual Build Date Notes Objective Originator Planned Build Date Serial Effectivity Serial Number

Business Object Type	Description	Assigned Attributes
Business Goal	Represents the hierarchical business goals of an organization.	Business Benefit Comments Effective Date Goal Originator Measure of Success Operational Definition Title
Business Unit	Any business unit that has persons who will be using the database, including a business unit within your own company, within a supplier company, and within a customer company. Also see Company object. Child of Organization.	All attributes inherited from Organization.
Buyer Desk	A logical place to consolidate parts, suppliers, and processes.	Comments Originator Primary Key Secondary Keys
CAD Drawing	CAD drawing for a part. Child of DOCUMENTS.	All attributes inherited from DOCUMENTS.
CAD Model	A document generated by an MCAD or ECAD tool that specifies a part. A given part object can have multiple such CAD Models. In addition, a given CAD Model could potentially correspond to multiple Parts. Each customer site will have business practices that govern this kind of interactions between CAD Models and Parts. Child of DOCUMENTS. Parent of MCAD Model and ECAD Model.	All attributes inherited from DOCUMENTS.
Capacitor Part	Child of Electrical Part.	All attributes inherited from Part.
Capital Expenditure Part	Child of Other Part.	All attributes inherited from Part.
Certification		Originator
Change Management	Abstract Parent of EBOM Markup. <u>Deprecated in version 10.5</u>	—
Characteristic	Abstract super type of the Characteristic types.	Title
Chemical Characteristic	Represents a Chemical Characteristic. Child of Characteristic.	All attributes inherited from Characteristic plus: Value

Business Object Type	Description	Assigned Attributes
Circuit Card Part	Child of Electrical Part. <i>In version 9512, the name and symbolic name for this type was renamed to correct a misspelling. If you refer to this type in any programs, you must change the symbolic name to reflect the corrected, updated name, which is type_CircuitCardPart.</i>	All attributes inherited from Part.
Company	Any company that has persons who will be using the database, including your own company, supplier companies, and customer companies. A Company business object is installed with the framework that is named Company Name. You should change the name of this business object to the name of your company, making sure it has the exact same name as the group that represents your company employees. For more information, see Defining Your Company (The Host Company) in Chapter 6. Child of Organization.	All attributes inherited from Organization. Secondary Vaults
Configurable Feature	Hidden A characteristic of a product that the customer must make a decision about. The decision affects how the product will be built. Each possible decision is an option. Think of a feature as a question the customer needs to answer. The possible answers are the options. Configurable features can be tangible characteristics, like monitor size, or intangible characteristics, like destination country. Child of Features. <i>Deprecated in 10.0.1.0.</i>	Display Text Feature Category Originator
Configurable Product	An assemble-to-order kind of product in which customers choose the features and options they want and the product is manufactured according to customer specifications. The object is the top level of a configurator structure that contains rules. Child of Products.	Base Price Currency Unit End Effectivity Originator Responsible Design Engineer Responsible Product Manager Start Effectivity Web Availability
Configuration File	Object created by the Release Manager to check in all the files that are related to the specific Software Build object.	Originator
Connector Part	Child of Electrical Part.	All attributes inherited from Part.
CoOwned List	Intermediate object that identifies co-owners of object.	Originator

Business Object Type	Description	Assigned Attributes
Cost Item	Cost Items for a specific Financial Item in a Ledger Account (expense or Investment). Child of Financials.	All attributes inherited from Financials plus: Ledger Account Number Planned Cost Estimated Cost Actual Cost
Currency Conversion	Defines exchange rates to go from one currency to another.	—
Customer Requirement Specification	Originating specification written by sales or customer.	All attributes inherited from Requirement Specification.
Customer Response Requirement Specification	The specification response sent to the customer.	All attributes inherited from Requirement Specification.
Customer Specification	Customer Specification or RFP. Child of Requirement Specification.	All attributes inherited from Requirement Specification.
Data Document	Attached to Request for Quote (RFQ) by the customer. Can be any document related to the RFQ, such as a specifications document, Bill of Materials, drawing, etc.	Designated User Error Report File Contents FTP Directory FTP Host Governing ECO Originator
Defect	Incident identified as a defect. Child of Incident.	All attributes inherited from Incident.
Deliverable	Abstract Any task that is required, such as a document, meeting, or action. Parent of Deliverable1, Deliverable2, Deliverable3	Originator Status
Deliverable1, 2, 3	Sample types for deliverables. Rename as appropriate to fit deliverables for a plan. Children of Deliverable	All attributes inherited from Deliverable.
Department	A department within a company. Child of Organization.	All attributes inherited from Organization.
Design Process	Design processes the supplier company is capable of. Child of Process.	Originator
Design Specification	Contains details about how a particular aspect (usually feature) of the software will be implemented. Child of Specification.	All attributes inherited from Specification.
Directed Supplier	This is a connection object for customer who directs part to a supplier.	Originator Responsibility

Business Object Type	Description	Assigned Attributes
Distribution File	Object that is created by the Release Manager to check in all the files that are related to the specific Software Build. <i>This type was originally named Software Distribution so its symbolic name is type_SoftwareDistribution.</i>	Notes Originator Platform
Document	An object that represents a single checked in file.	Checkin Reason Designated User File Version Language Originator Primary Key Secondary Keys Title
DOCUMENT CLASSIFICATION	The abstract type that is the parent of library, bookshelf, book, and any other containers that may be defined.	Approver Created On Originator Title
Document Management	Abstract Parent of Document and Project Plan. <i>Deprecated in version 10.5.</i>	—
Document Part	Child of Other Part.	All attributes inherited from Part.
Document Sheet	Used for storing document sheets.	Created On Originator Sheet Number Title
DOCUMENTS	Type created to group all documents.	Access Type Designated User Originator Title Is Version Object Move Files To Version Suspend Versioning Language Checkin Reason
Drawing Print	A viewable image of drawings that does not require CAD and is the electronic equivalent to the paper-based drawing blueprint. Child of Documents.	All attributes inherited from Documents.
Duplicate	An incident that has been determined to be a duplicate of an incident reported already. Child of Incident.	All attributes inherited from Incident.
EBOM Markup	Object that represents markups to an EBOM.	Originator

Business Object Type	Description	Assigned Attributes
ECAD Model	A document generated by an ECAD tool that specifies a part. Child of CAD Model.	All attributes inherited from CAD Model.
ECO	Engineering Change Orders specify the exact changes made to the new revisions of the objects in order to satisfy an approved ECR.	Branch To Originator Priority Reason For Cancel Release Distribution Group Responsible Design Engineer Responsible Manufacturing Engineer
ECR	Engineering Change Requests identify problems or required changes relating to existing business objects and request that a change be made.	Category of Change ECR Evaluator ECR Originator Organization General Description of Change Originator Primary Key Reason For Change Responsible Design Engineer Reviewers Comments Secondary Keys
Effort	Used to estimate and capture effort for all parts.	Actual Effort Estimated Effort
Electrical Part	Child of Hardware Part.	All attributes inherited from Part.
Erratum	Describes a correction to be done in the documentation. Child of Incident.	All attributes inherited from Incident.
eService Number Generator	Each object type that requires a unique numbering sequence for its automatic names has an eService Number Generator object. The object specifies the number to be used in the name of the next object created that uses that numbering sequence. An object generator program increments the number by one each time an object is created.	eService Next Number
eService Object Generator	Each object type that is named automatically has an eService Object Generator object. The object contains attributes that define the naming convention for the automatically-generated names and other attributes that control the program that creates the names. For information on configuring the automatic naming process, see Configuring Automatic Business Object Naming in Chapter 6.	eService Name Prefix eService Name Suffix eService Processing Time Limit eService Retry Count eService Retry Delay eService Safety Policy eService Safety Vault

Business Object Type	Description	Assigned Attributes
eService Trigger Program Parameters	A business object that represents a trigger program that should be executed when a trigger event occurs. Each of these business objects, which are called by the Trigger Manager program, stores the name of a trigger program to be executed, values for the program parameters, and the order in which the programs are to be executed. For more information, see Configuring Automatic Business Rules (Triggers) in Chapter 6.	eService Constructor Arguments eService Method Name eService Program Name eService Sequence Number eService Program Argument <i>n</i> eService Program Argument Description <i>n</i> eService Target States
Event	An event that a person subscribes to for an object.	Event Type Originator
Extruded Part	Child of Mechanical Part.	All attributes inherited from Part.
Fastener Part	Child of Mechanical Part.	All attributes inherited from Part.
Feature	Feature type that will be shipped with the product. Child of Features. <i>This type was originally named Configurable Feature and therefore its symbolic name is type_ConfigurableFeature.</i>	All attributes inherited from Features.
Feature List	Intermediate business object that enables product specific rules to be applied to feature/options.	Comment Default Selection Expression Feature Selection Type List Price Maximum Quantity Minimum Quantity Originator Rule Type
Feature Requirement Specification	Requirement specification for a feature. Child of Requirement Specification.	All attributes inherited from Requirement Specification.
Feature Rule	Every Configurable Feature object has a Feature Rule object that connects the feature to the product. The Feature Rule object specifies the feature selection type and inclusion rule for the feature it is connected to.	Comment Expression Feature Selection Type Originator Rule Type
Feature Specification	Specifications related to features. Child of Product Specification.	All attributes inherited from Product Specification.
Features	Abstract Defines the feature hierarchy.	Difficulty Feature Category Feature Classification Marketing Feature Marketing Name Marketing Text Notes Originator Synopsis Technical Feature

Business Object Type	Description	Assigned Attributes
File Format	The file formats used in the company.	File Extension Originator
Final Quotation	A document from a supplier indicating the exact, contractually binding cost of an item. Child of Quotation.	All attributes inherited from Quotation.
Financial Benefit Category	Stores all pre-defined Benefit Categories.	—
Financial Cost Category	Stores all pre-defined Cost Categories.	—
Financial Item	The top-level Financial Entry in a Ledger Account.	Attributes inherited from Financials plus: Actual Benefit Actual Cost Actual Net Benefit Benefit Interval Benefit Interval End Date Benefit Interval Start Date Cost Interval Cost Interval End Date Cost Interval Start Date Estimated Benefit Estimated Cost Estimated Net Benefit Planned Benefit Planned Cost Planned Net Benefit Total Budget Cost
Financials	Abstract type storing all the financial type necessary to build the financial spreadsheets. Parent of Financial Item, Benefit Item, Cost Item, Interval Item Data, Financial Cost Category, Financial Benefit Category	Notes Originator
Fixed Resource	A resource for a product that can be consumed or supplied by the product's options. For example, a computer product might need a fixed resource for Available Card Slots. Options such as video cards and sound cards use up this resource. Some options, such as an expansion card, might add to the resource.	Comment Initial Resource Originator Resource ID Resource Maximum Resource Minimum
Formula Characteristic	Represents a Formula Characteristic.	All attributes inherited from Characteristic plus: Value

Business Object Type	Description	Assigned Attributes
Formula Technical Specification	An instantiable type that can also be sub-typed further by customers depending on the formula the customer has available. Child of Technical Specification	Attributes inherited from Technical Specification and: Appearance Color Density General Requirements Kind of Product Odour pH Shelf Life Storage Conditions Value
Frame Part	Child of Mechanical Part.	All attributes inherited from Part.
Functional Specification	Contains development/programming details about a particular aspect (usually feature) of the software. Child of Specification.	All attributes inherited from Specification.
Further Chemical Physical Characteristic	Represents a Further Chemical Physical Characteristic. Child of Characteristic.	All attributes inherited from Characteristic plus: Tolerance Upper Tolerance Lower Units Value
GBOM	Intermediate business object that enables product specific rules to be applied to parts.	Comment Expression Originator Rule Type
Generic Characteristic	Represents Generic Characteristics. Child of Characteristic.	All attributes inherited from Characteristic plus: Average Tolerance Upper Tolerance Lower Units
Generic Document	Generic Document	Approver Designated User Originator Release Type Title
Hardware Build	Describes the hardware builds. Child of Builds.	All attributes inherited from Builds.
Hardware Part	Child of Part.	All attributes inherited from Part.
Hardware Product	Describe the hardware products provided by a company. Child of Products.	All attributes inherited from Products.
Holder	Holder that identifies a company's ECRs, RFQs, packages, and currency conversions.	Originator

Business Object Type	Description	Assigned Attributes
HTTP Request	Holds the search criteria, if the Query type is Expand.	Find Type Request Parameters Search Type Title Usage
HTTP Request Holder	Holder Object for HTTP Request and Person.	—
Image	Provides the ability to manage marketing images for products, features and models.	Image Horizontal Size Image UOM Image Vertical Size Marketing Name Marketing Text Originator
Impact Analysis	A study of the effect of a request or bug on the software product, usually performed by an engineer.	Designated User Estimated lost effort Estimated Schedule Impact Estimated total effort Impact Analysis Effort Implementation Effort Life cycle cost issues Originator Other Requirements Affected Other Tasks Affected Prioritization Benefit Priority Quality Impact Relative Cost Relative Penalty Relative Risk Severity Validation Effort

Business Object Type	Description	Assigned Attributes
Inbox Task	A request that a project member perform some activity—such as review, comment, or approve—related to a document or object that is routed to the person. When people create routes, they specify who they want to send the item to and the tasks people should complete when they receive it.	Actual Completion Date Allow Delegation Approval Status Approvers Responsibility Assignee Set Due Date Comments Date Offset From Due Date Offset Escalation Recipients Originator Review Comments Needed Reviewers Comments Review Task Route Action Route Instructions Route Node ID Route Task User Route Task User Company Scheduled Completion Date Task Comments Needed Task Requirement Template Task Title
Incident	An issue that needs to be addressed in the software code, design, or documentation. After a new incident is evaluated, it is changed to a bug, a request, an errata, or a duplicate.	Change Board Closed By Closed Date Comments Error Category Error Navigation Incident Disposition Origination Originator Platform Priority Product Reason for Closure Regression Regression Version Severity Synopsis Test Effort Time of Occurrence Version Workaround
Initial Quotation	A document from a supplier indicating the accurate (but not contractually binding) cost of an item. Child of Quotation.	All attributes inherited from Quotation.
Interval Item Data	Placeholder type containing no attributes. Child of Financials.	—

Business Object Type	Description	Assigned Attributes
Issue	An object that represents a problem, questions, or consideration regarding any other object. For example, an issue attached to a Request for Quote might raise questions about the type of material used, testing requirements, etc.	Originator Escalation Required Estimated Start Date Estimated End Date Actual Start Date Actual Finish Date Priority Resolution Recommendation Action Taken Resolution Statement Steps To Reproduce Public Issue Classification Issue Category
Issue Category	Categorize the type of issue.	Originator
Issue Classification	Classify the type of issue.	Originator
Library	A collection of bookshelves.	Originator Created On
Line Item	The item number that a buyer wants quoted for a specific Request To Supplier business object. <i>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</i>	Entered Name Primary Key Secondary Keys
Line Item Template	Defines how a user is supposed to fill in attributes from an Attribute Group business object when associated to a Line Item or Supplier Line Item business object.	Attribute Group Type Company Attributes Line Item Template Scope Primary Key Secondary Keys User Symbolic Name
Location	Physical site with an address.	Address 1 Address 2 City Country FTP Directory FTP Host Location Code Originator Postal Code Primary Key Region State/Region
Location Equivalent Object	Represents location context MEP, used as an intermediate object when location specific MEP is created.	Originator

Business Object Type	Description	Assigned Attributes
Machined Part	Child of Mechanical Part.	All attributes inherited from Part.
Manufacturing Assessment	A document from a supplier indicating the manufacturing feasibility of an item's design.	Comments Cost Assessment Lead Time Assessment Manufacturability Originator Quality Assessment Quote Requested by Date Request Disposition Responsibility Supplier Response Tooling Assessment
Manufacturing Process	Manufacturing processes the supplier company is capable of. Child of Process.	Originator
Markup	Object that represents the markup for an object, such as a drawing or document. Child of DOCUMENTS.	All attributes inherited from DOCUMENTS.
MCAD Assembly	Represents an assembly model in the CAD tool. It is distinct in that it references multiple other objects of type "Assembly" or "Component" and needs these files present in order to be read by the MCAD tool. Its lifecycle typically depends on the lifecycle of its constituent components. Child of MCAD Model.	All attributes inherited from CAD Model.
MCAD Assembly Instance	This is very much like an assembly (as reflected by the subtyping). It differs from the assembly in that it owes its existence and can be modified only in the presence of the assembly of which it is an instance. Child of MCAD Assembly.	All attributes inherited from CAD Model.
MCAD Component	Represents a component model in the CAD tool. It is distinct in that the MCAD tool can load it all by itself and is therefore a self-contained model. Child of MCAD Model.	All attributes inherited from CAD Model.
MCAD Component Instance	This is very much like a component (as reflected by the subtyping). It differs from the component in that it owes its existence and can be modified only in the presence of the component of which it is an instance. Child of MCAD Component.	All attributes inherited from CAD Model.
MCAD Model	Documents generated by an MCAD tool. Child of CAD Model.	All attributes inherited from CAD Model.
MCAD Parametrized Assembly Model	This type represents a parameterized assembly. Assemblies are parameterized differently from components and therefore need a separate type to represent them. Child of MCAD Parametrized Model.	All attributes inherited from CAD Model.

Business Object Type	Description	Assigned Attributes
MCAD Parametrized Component Model	This type represents a parameterized component. Child of MCAD Parametrized Model.	All attributes inherited from CAD Model.
MCAD Parametrized Model	This represents a common concept in MCAD tools. This model is a modeling construct and does not correspond to something actually made on the shop floor. It is a model driven by parameters (which could be geometric dimensions of the model or other things). It is distinct in that it is never directly included as part of an existing model structure. Child of MCAD Model.	All attributes inherited from CAD Model.
Mechanical Part	Child of Hardware Part.	All attributes inherited from Part.
Media	Object created by a Quality Engineer for storage of the Distribution File object, which can be tape, CD-ROM, etc.	Label Originator Unit Price
Meeting	Object that represents a specific meeting. <i><u>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</u></i>	Conference Call Access Code Conference Call Number Meeting Duration Meeting ID Meeting Key Meeting Location Meeting Site Name Meeting Start Date Time On-line Meeting Instructions On-line Meeting Provider Originator Title
Member List	Distribution list.	Originator
Message	An object that represents a posted message that can be related to any type of object, such as a file. <i><u>This admin object is used in one or more filter expressions for policy accesses. If the admin object name is changed, any expressions that include it must be changed to reflect the new name. See Administrative Objects Used in Expression (Filter) Accesses.</u></i>	Count Next Message Originator Sequence Order Subject Synopsis
Microbiological Characteristic	Represents a Microbiological Characteristic Child of Characteristic.	All attributes inherited from Characteristic plus: Absence Of Requirement Total Count
Model	Models are used to describe a company's catalog.	Marketing Name Marketing Text Originator
Molded Part	Child of Mechanical Part.	All attributes inherited from Part.
Nut Part	Child of Fastener Part.	All attributes inherited from Part.

Business Object Type	Description	Assigned Attributes
Option	See Configurable Feature .	Display Text Originator
Option Rule	Every Option object has an Option Rule object that connects the option to its feature. The Option Rule object contains attributes for the option such as quantity rules and list price.	Comment Default Selection Expression List Price Maximum Quantity Minimum Quantity Originator Rule Type
Organization	The Organization object keeps track of the complete profile of the companies added. The profile consists of the company name, company ID, Cage code, DUNS number. Other details could be the location and person the company is associated with, the company's capabilities, and the software the company is using. Parent of Company and Location.	Address Cage Code City Country Default Policy Division DUNS Number File Site File Store Symbolic Name FTP Directory FTP Host Meeting Site ID Meeting Sitename Organization Fax Number Organization ID Organization Name Organization Phone Number Originator Postal Code Primary Key State/Region Web Site
Organization List	Used to collect one or more Organization objects together so they can be easily retrieved for future use.	Primary Key Secondary Keys
Other Part	Child of Part.	All attributes inherited from Part.
Package	Used to group one or more Request To Supplier business objects.	Allow User Edit Co-Owner Comments Primary Key Secondary Keys
Packaging Characteristic	Represents a Packaging Characteristic. Child of Characteristic.	All attributes inherited from Characteristic plus: Max Packaging Height Packed Requirement Packed On Date Packaging Size

Business Object Type	Description	Assigned Attributes
Packaging Technical Specification	An instantiable type that can also be sub-typed further by customers depending on the products the customer has available. Child of Technical Specification	Attributes from Technical Specification
Part	An item that can be inventoried and of which a physical product is composed. Most companies use parts as the basis from which they specify their products and product configurations.	Classification Effectivity Date Estimated Cost Lead Time Material Category Originator Production Make Buy Code Service Make Buy Code Spare Part Target Cost Unit of Measure Weight
Part Family	A logical grouping of parts that a company can define. Parts within the family can be managed as a whole. It permits product specifications to be associated with the family (rather than each individual part).	Part Family Name Generator On Part Family Base Number Part Family Prefix Pattern Part Family Suffix Pattern Part Family Last Used Sequence Part Family Pattern Separator Part Family Sequence Pattern
Part Master	<u><i>Part Master and MBOM are not currently being used by the applications and are included in the AEF to indicate the direction that we plan to head from a schema perspective.</i></u>	Originator
Part Quality Plan	A list of deliverables that a buyer expects a supplier company to provide when supplying a part. Child of Project Management.	All attributes inherited from Project Management.
Part Quality Plan Template	A template used to create new part quality plans. Child of Project Management.	All attributes inherited from Project Management.
Part Rule	In Product Option Manager, every Part object has a Part Rule object that connects the part to an option or directly to the product. The Part Rule object specifies the inclusion rule for the part it is connected to.	Comment Expression Originator Rule Type

Business Object Type	Description	Assigned Attributes
Person	This object keeps track of the Person object profile and contact details.	Absence Delegate Absence End Date Absence Start Date Address City Country Email Address Fax Number File Store Symbolic Name First Name Home Phone Number Host Meetings Last Login Date Last Name Login Type Meeting Key Meeting Password Meeting Username Middle Name Originator Pager Number Postal Code Preferences State/Region Subscription Level Web Site Work Phone Number
Personal Folder	A personal folder is used for a user to store objects. It is visible only to the owner of the folder. Child of ADHOC CATEGORY <i>Deprecated in version 10.0.0.0.</i>	Originator
Physical Characteristic	Represents a Physical Characteristic. Child of Characteristic.	All attributes inherited from Characteristic plus: Value
Plan	Abstract Parent of Part Quality Plan.	Originator Scheduled Completion Date Status
Plant Scorecard	Scores by a host plant location for a supplier location for a specific period. Child of Scorecard.	All attributes inherited from Scorecard. Metric Integer 01 through 15
Process	All the processes of which the supplier company is capable. Parent to Manufacturing Process, Design Process, Testing Process.	Originator

Business Object Type	Description	Assigned Attributes
Process Technical Specification	An instantiable type that can also be sub-typed further by customers depending on the products the customer has available. Child of Technical Specification	Attributes inherited from Technical Specification
Product	The product that consists of software. <i>Deprecated in 10.0.1.0.</i>	Notes Originator Synopsis
Product Configuration	A unique list of features and options ordered by a customer.	Configurable Product Name Marketing Name Marketing Text Ordered Options Originator Precise BOM Validation Status
Product Documentation	The product documentation.	Checkin Reason File Version Language Notes Originator Title Version Version Date
Product Features	Abstract This is the feature type that will be shipped with the product. Child of Features.	All attributes inherited from Features.
Product Line	Holds information used to set attributes in the ECR when it is created. This object handles the mapping from product line to change board for routing purposes.	Change Board Marketing Name Marketing Text Originator
Product Requirement Specification	Child of Requirement Specification Specification for product requirements.	All attributes inherited from Requirement Specification.
Product Specification	Child of Specification Parent of System Specification, Feature Specification, Software Specification, Build Specification, Test Specification, Use Case Specification	Attributes from Specifications and: Capacity Diameter Height Length Wall Thickness Weight Width

Business Object Type	Description	Assigned Attributes
Products	Abstract Defines the product hierarchy.	Base Price Current Version End Effectivity Is Version Marketing Text Marketing Name Notes Originator Responsible Design Engineer Responsible Product Manager Start Effectivity Synopsis Web Availability
Program	Project for identifying sub projects or grouping of projects Grouping of projects by sponsoring program (most companies use programs to group projects – not product lines or products)	Notes Originator Task Actual Duration Task Actual Finish Date Task Actual Start Date Task Estimated Duration Task Estimated Finish Date Task Estimated Start Date Title
Project Access List	Project Access List	—
Project Concept	A concept or idea prior to starting a real project A new child type of Project Management. This type will be used for new Product ideas that have a different life cycle than Project. Child of Project Management.	All attributes inherited from Project Management File Store Symbolic Name Project Visibility
Project Folder	A container for one or more files that are routed to specified project members. The route creator names and describes the folder when the route is created. <i>Deprecated in version 9.5.0.0.</i>	Originator
Project Management	Top level abstract type for managing projects and tasks Abstract type. Parent of Project Concept, Project Space, Project Template, and Task Management.	Constraint Date Note Originator Percent Complete Project Function Project Type Schedule Duration Units Task Actual Duration Task Actual Finish Date Task Actual Start Date Task Estimated Duration Task Estimated Finish Date Task Estimated Start Date Title

Business Object Type	Description	Assigned Attributes
Project Member	An object that represents a person who has been added to a project.	Create Folder Create Route Create Vault Host Meetings Originator Project Access
Project Space	Any kind of business project managed in Program Central. Child of Project Management	Inherits all attributes from Project Management Baseline Current End Date Baseline Current Start Date Baseline Initial Start Date Baseline Initial End Date File Store Symbolic Name Escalation Date Recurrence Interval Send Reminder
Project Template	A Template Project for creating new projects Project templates are owned at the company/ business unit level. Project Templates are predefined projects with WBS structure, suggested deliverables, durations, etc. Project Templates will be created and managed by company representatives. Users will be allowed to create a project by selecting predefined project templates. The user will be presented with a set of questions to determine what WBS structure to copy from. Child of Project Management.	All attributes inherited from Project Management File Store Symbolic Name Project Visibility
Public Folder	Lets a user store documents. <i>Deprecated in version 10.0.0.0.</i>	Originator
Publish Subscribe	This object type serves as a connector between the objects and the events that users can subscribe to. For example, when a user subscribes to be notified whenever a route is created for a project, then a Publish Subscribe object is created that connects the project object with the event object for creating routes.	Originator
Purchase Class	Classification for purchased goods.	Originator
Purchase Class Scorecard	Scores by a host organization for a purchase class for a specific period. Child of Scorecard.	All attributes inherited from Scorecard.

Business Object Type	Description	Assigned Attributes
Quality	The quality plan, goal, and metrics for a project.	Comments Constraints Defect Definition Goal Operational Definition Opportunity Originator Out of Bounds Problem Statement Quality Type
Quality Metric	The metrics data for a quality object.	Originator
Question	Ability to filter project tasks based on questions A Project Template is a survey of questions that determine the mandatory or optional task structure to be used in a project structure. A set of questions determines what task structure to use. Question can determine if the Task is optional or mandatory.	Originator
Quotation	Documents that convey some form of pricing or cost information as entered by the supplier. Abstract Parent to Initial Quotation, Budgetary Estimate, Final Quotation.	Annual Part Quantity Comments Container Cost Lead Time Production Lead Time Prototype Lead Time Tooling Originator Production Purpose Quote Requested by Date Request Disposition Responsibility Sourcing Product Sourcing Spares Supplier Response Tooling Cost Unit Price
Rate Period	Used as a single end point for one or more Rate Period connections form a Currency Conversion business object.	—

Business Object Type	Description	Assigned Attributes
Raw Material Technical Specifications	An instantiable type that can also be sub-typed further by customers depending on the products the customer has available. In the CPG industry, some examples of Raw Material sub-types would be Sugar Raw Material or Salt Raw Materials. Child of Technical Specifications	Attributes from Technical Specifications and: Appearance CAS Number Color Density EINECS Number General Requirements Max Palette Height Odour Origin Palette Size PH Raw Material Package Safety Description Safety Symbol Safety Warning Shelf Life Storage Conditions
Region	Region of a company.	Originator
Request	A change that someone has asked for in the software design. Child of Incident.	All attributes inherited from Incident.
Request for Quote	A request by a customer for a quotation from a supplier company. Specifies details about the item or work that is being requested and invites quotes, specifies a "Quote Requested By" date.	Comments Error Report Originator Quote Requested by Date Requested Work Result Submitted Date
Requirement	Propose a change to or introduce a new requirement for product.	Designated User Difficulty Estimated Cost Notes Originator Priority Requirement Category Sponsoring Customer Synopsis User Requirement Classification User Requirement Importance
Requirement Baseline	Used to manage baselines.	Originator

Business Object Type	Description	Assigned Attributes
Requirement Specification	Originating specification written by sales or customer. Child of Specification. <i>This type was originally named Software Requirement Specification and therefore its symbolic name is type_SoftwareRequirementSpecification.</i>	All attributes inherited from Specification.
Resistor Part	Child of Electrical Part.	All attributes inherited from Part.
RFQ	A person in a buyer organization creates an RFQ to make suppliers aware of the opportunity to respond to the buyer organization about work on parts or the feasibility of their manufacture. The main purpose of an RFQ is to act as a package for other items and to record the status of the package as a whole. <i>This type was originally named Request To Supplier and therefore its symbolic name is type_RequestToSupplier.</i>	Allow User Edit Annual Part Quantity Comments Co-Owner Currency Default Responses Default Responsibility Last Legacy Update Legacy System Identifier Legacy System Source Line Item Roll-up Calculation Formulas Line Item Roll-up Formulas Originator Primary Key Production Purpose Purchase Decision Quotation Response Review Quote Requested by Date Secondary Keys Sourcing Product Sourcing Spares Unit Price Calculation Formula Unit Price Formula
RFQ Quotation	Defines a supplier's response to all line items for a RFQ business object. <i>This type was originally named RTS Quotation and therefore its symbolic name is type_RTSQuotation.</i>	Allow User Edit Co-Owner Comments Primary Key Secondary Keys

Business Object Type	Description	Assigned Attributes
RFQ Template	Defines mandatory and optional information that must be provided by the buyer in an instantiated RFQ and provided by the suppliers in their respective RFQ Quotation instances. <i>This type was originally named RTS Template and therefore its symbolic name is type_RTSTemplate.</i>	Company Buyer Attributes Company Supplier Attributes Extend Unit Price Formula Line Item Roll-up Calculation Formulas Line Item Roll-up Formulas Quotation Header Roll-up Calculation Formulas Quotation Header Roll-up Formulas Primary Key Secondary Keys Unit Price Calculation Formula Unit Price Formula
Risk	Represents a risk for a project, software feature, or release. Within the context of Program Central, Risk objects capture, track, and manage project risks from identification to resolution. During the risk process, Program Central captures risk details and tracks the mitigation process.	Abatement Plan Actual End Date Actual Start Date Comments Estimated End Date Estimated Start Date Measure of Success Notes Originator Risk Category Risk Factor Risk Impact Risk Probability Risk Visibility Risk Type Title
Route	A path that a person defines for an object. The path can send the object to one person at a time or to more than one person. The person who creates the route specifies the task each person should perform when the person receives the object.	Co-Owners Current Route Node Originator Restrict Members Route Base Purpose Route Completion Action Route Instructions Route Status Sub Route Visibility
Route Task User	This type supports the Unassigned Tasks feature. Both Route and Route Templates support Unassigned Tasks and Assignment of Roles/Groups to tasks.	Originator Route Task User Title
Route Template	A route that is saved (people, sequence, tasks, instructions, etc.) and can be reused when other routes are created.	Co-Owners Originator Restrict Members Route Base Purpose Route Instructions Task Edit Setting

Business Object Type	Description	Assigned Attributes
RPN	The Risk Priority Number (RPN) is used to evaluate and prioritize risks associated with a risk/issue. The formula for risks is the product of Probability and Impact. For example a probability can be 1,3,5 and impact can be 1,3, 5. These 2 elements determine each issue's associated RPN by multiplying the attribute ratings for probability and impact.	Originator
RTS Quotation Archive		Originator
Rule Extension	An object that executes a program to check the validity of a product configuration. Use rule extensions to perform a validity check that can't be performed using any other rules supplied by Product Option Manager.	Comment Originator Program Rule Program Rule Argument RuleExtension ID
Safety and Regulatory Characteristic	Represents a Safety and Regulatory Characteristic. Child of Characteristic.	All attributes inherited from Characteristic plus: Symbol Warning Description
Sales Order	An order for one or more configurable products.	Order Quantity Order Recipient Originator Promised Delivery Date Requested Delivery Date Ship To Total Cost
SCO	Specification Change Order (SCO) used for issuance/obsolescence of the spec.	Originator Reason For Change Render Success Flag Render Error Message Rendition In Process
Scorecard	Abstract Scores by an object (such as a location) for an object (such as a supplier) for a specific period. Parent of Plant Scorecard, Purchase Class Scorecard, and Supplier Scorecard.	Originator Period Period Type Value Type
Screw Part	Child of Fastener Part.	All attributes inherited from Part.
Service Product	Services Product type. Child of Products.	All attributes inherited from Products.
Sketch	Holds documents that explain the change requested in an ECR. Child of DOCUMENTS.	All attributes inherited from DOCUMENTS.
Software Build	The process of preparing the software for release to the customer. The end result of the process is also referred to as a build. Created by Software Manager. Child of Build.	Inherits attributes from Builds plus: Notes Objective Originator

Business Object Type	Description	Assigned Attributes
Software Feature	Represents one feature in a software product. Child of Features.	Inherits attributes from Features plus: Actual Time Estimated Max Estimated Min Feature Classification Notes Originator Synopsis
Software Part	Child of Part.	All attributes inherited from Part.
Software Platform	The type of computer system that will run the software (e.g. Windows, UNIX).	Hardware Vendor Originator Software Vendor
Software Product	Describes the software products provided by a company. Child of Products.	All attributes inherited from Products.
Software Release	A software release gets implemented either as part of a product release or an independent software release.	Actual Time Estimated Max Estimated Min Notes Originator Synopsis
Software Specification	Specifications for software. Child of Product Specification.	All attributes inherited from Product Specification.
Source Code Directory	Directory build by integration or user to manage source code files.	Originator Notes Synopsis
Source Directory		Originator
Source File	Source File for the affected software code.	Designated User Notes Originator Synopsis
Specification	Abstract The abstract Specification type, which serves as the base type for Functional Specification, Design Specification and Test Specification. Parent of Functional Specification, Design Specification, Requirement Specification, Product Specification, Customer Specification, Customer Requirement Specification, Customer Response Requirement Specification, Product Requirement Specification, Feature Requirement Specification, System Requirement Specification, System Specification, Feature Specification, Software Specification, Build Specification, Test Specification, Use Case Specification Child of Document	Actual Time Checkin Reason Designated User Estimated Max Estimated Min File Version Language Notes Objective Originator Synopsis Title Version Version Date

Business Object Type	Description	Assigned Attributes
Specification Office	Specification office for a company.	Originator
Specification Section	Defines the spec view section for the specification.	Attribute Filter Originator Section Access Sequence Order Section Type Table Name Title
Specification Template Holder	Identifies specification templates.	Originator
Standards Manual	A Manual, Section, or Page that displays content.	Manual Title Originator Release Release Date
Substitute	A substitute for a part.	Find Number Originator Quantity Reference Designator
Supplier Development Plan	A development plan for a supplier. Child of Plan	All attributes inherited from Plan.
Supplier Line Item	Relates a part and supplier to RFQ per quote. <u><i>This type was originally named RTS Supplier Part and therefore its symbolic name is type_RTSSupplierPart.</i></u>	Originator Primary Key Responsibility Secondary Keys
Supplier Line Item Price Period	Used as a single end point for one or more Supplier Line Item Price Period connections from a Supplier Line Item business object.	Primary Key Secondary Keys
Supplier Scorecard	Scores by host organization for a supplier organization for a specific period. Child of Scorecard.	All attributes inherited from Scorecard. Metric Real 01 through 10
System Requirement Specification	Child of Requirement Specification	All attributes inherited from Requirement Specification.
System Specification	Child of Product Specification	All attributes inherited from Product Specification.
Task	Specifies some specific work that must be completed. Child of Task Management	Synopsis Task Estimated Duration Maximum Task Estimated Duration Minimum All other attributes inherited from Task Management
Task Escalation Message	Used as a single “end point” for one or more Task Escalation Message connections from a Workspace business object.	—

Business Object Type	Description	Assigned Attributes
Task Management	Identifies the type of tasks that define the project structure Abstract type This type will be used to dynamically understand the WBS structure requirements for a Project.	Task Requirement Baseline Initial Start Date Baseline Current Start Date Baseline Initial End Date Baseline Current End Date All other attributes inherited from Project Management
Technical Specification	Abstract type for managing technical specs	Approved Date Authorized Temporary Specification Claimed Attribute List Effectivity Date Expiration Date Global Attribute List Hidden Attribute List Is Template Issued Date Legacy Local Attribute List Lower Limit Originator Properties Form Name Properties Template Form Name Regional Attribute List Render Error Message Render File Name Render Language Render Success Flag Security Classification Specification Category Specification View Form Name Template Level Title Tolerance Unit of Measure Upper Limit
Template Filter		Originator Relate Object Type
Test Case	Used to validate a software feature or a user requirement or a use case based on the object to which it is attached.	Actual Completion Date Designated User Estimated Completion Date Notes Originator Synopsis Validation Comments Validation Status
Test Specification	Contains details about the testing of a particular aspect (usually feature) of the software. A test specification is used to validate a software feature before it gets released. Child of Specification.	All attributes inherited from Specification.

Business Object Type	Description	Assigned Attributes
Testing Process	Testing processes the supplier company is capable of. Child of Process.	Originator
Thread	An object that connects all the messages for a discussion and connects the object being discussed. For example, if a discussion includes five messages about a document, the five message objects connect to the thread object and the document object connects to the thread object.	Next Message Originator
Tooling Part	Child of Other Part.	All attributes inherited from Part.
Transistor Part	Child of Electrical Part.	All attributes inherited from Part.
UDA	User-Defined Attribute. Used as a single end point for one or more UDA cost/miscellaneous connections form a business object.	—
Unit Conversion	Used to store unit-of-measure conversion rates.	Rate
URL	Allows the creation of hyperlinked external URLs. The URL may or may not be a Website. It could be an internal URL link to a document. The NAME field is used to store the label of the URL. The Revision field is auto-numbered. The Description field is used to store the long URL link.	Link URL Originator
Use Case	A use case depicts the way the end customer will use a software feature. Use cases can be used to validate a software feature.	Actual Time Designated User Estimated Max Estimated Min Notes Originator Synopsis
Use Case Specification	Specification that defines a use case. Child of Product Specification.	All attributes inherited from Product Specification.
User Manual	Documents that describe the product.	Checkin Reason Designated User File Version Language Notes Objective Originator Synopsis Title Version Version Date
User Requirement	Child of Requirement	Attributes inherited from Requirement and: User Requirement Classification

Business Object Type	Description	Assigned Attributes
Version Document	Stores version of files checked into DOCUMENTS.	Checkin Reason File Version Originator Title Version Version Date
Viewable	A viewable document that is used primarily for visualization and collaboration purposes. It is typically derived from a CAD Model. Users will never directly edit the contents of a viewable model. It is a much more lightweight representation of the CAD Model which can be viewed by custom viewing tools. Parent of 2D Viewable and 3D Viewable.	Originator Viewable Base Filename Viewable Base Format Viewable State Viewing Tool
Washer Part	Child of Fastener Part.	All attributes inherited from Part.
Workspace	Any kind of business project that a company needs to collaborate on using Team Central. <u><i>This type was originally named Project and therefore its symbolic name is type_Project.</i></u>	Originator Primary Key Project Name Scope Title
Workspace Template	Defines a common and repeatable way for creating new workspace objects.	Originator
Workspace Vault	A set of files that are related and that concern a particular project. <u><i>This type was originally named Project Vault and therefore its symbolic name is type_ProjectVault.</i></u>	Access Type Count Originator Project Security Vault

Vaults

Vault	Description
eService Administration	Contains the business objects that support the MatrixOne applications, such as the objects that manage automatic object naming and triggers. You can rename this vault but do not delete it.
eService Production	Contains the business objects for the production environment for the host company. Rename this vault as needed and use it as the default vault. Also make this the safety vault for object generators.
eService Sample	Contains all sample data installed with the framework. Sample data is useful for learning the schema and application, and for testing. <i>Never store data that you want to keep in this vault.</i> None of the sample data is needed to run the application in a production environment. You can remove the sample data from the installation process. For information on how to configure the installation process, see <i>The Installation Process</i> in the <i>Matrix PLM Platform Installation Guide</i>. You can also delete the eService Sample vault. To delete a vault, you must first delete all business objects from it using the <code>clear vault</code> MQL command. Then use the <code>delete vault</code> command.

Wizards

The framework includes administrative wizards. To access the wizards, log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user (as defined in the person definition). Choose Administration Manager from the role bar and then choose the wizard from the toolset.

Wizard	Description
eService Change Administration Property	Lets you add, update, delete, and transfer the symbolic name property for an administrative object, as well as the five properties that facilitate upgrades. Use when you want to use a custom (non-framework) administrative object in place of a framework object. Also use to register custom objects using your own registration program. For more information, see the following sections: Viewing Administrative Properties , Merging Schema , and Registering Your Own Administrative Objects .
eService Change State Property	Use to update state name properties whenever you change, add, or delete states for framework policies. For information, see Configuring State Names in Chapter 6.
eService Language	Use to change the base language and to add language aliases for all admin objects. <i>This wizard has been deprecated and should no longer be used.</i>
eService Register Application	Use when you are adding or removing the applications available to users. Also use this wizard to change the menu name for an application. For information, see Adding Suites, Applications, Features, and Pages in Appendix B. <i>This wizard should only be used for version 8 and 9 applications.</i>
eService Register Feature	Use when you are adding or removing the tasks available to users. Also use this wizard to change the menu name for a feature. For information, see Adding Suites, Applications, Features, and Pages in Appendix B. <i>This wizard should only be used for version 8 and 9 applications.</i>
eService Register Page Access	Use to control which roles have access to particular MatrixOne application pages. For information, see Adding Suites, Applications, Features, and Pages in Appendix B. <i>This wizard should only be used for version 8 and 9 applications.</i>
eService Register Suite	Use when you are adding or removing the suites available to users. Also use this wizard to change the menu name for a suite. For information, see Adding Suites, Applications, Features, and Pages in Appendix B. <i>This wizard should only be used for version 8 and 9 applications.</i>
eService Register Viewer	Use to register a 2D/3D viewer so people can launch the viewer from a MatrixOne application and view documents in the viewer. Also use to assign formats to viewers. For information, see Setting Up 2D and 3D Viewers in Chapter 6.

Policy Access Tables

This section lists the accesses for each state in each policy installed with the framework. The table for each policy includes other important information about the policy, such as whether it is hidden, revisionable, and versionable, and what types the policy governs. Look in the footer row of each table for any special notes about the policy.

The tables below do not include signature requirements defined to promote and demote between states. Use MQL or Business Modeler to get signature information.

Every policy gives All access to Shadow Agent, System Conversion Manager, and System Transition Manager in every state. Shadow Agent is used to perform actions that the current user may not have access to do (using push and pop context). The System Manager accesses are used for data conversions from legacy systems.

To see the automation programs that run throughout each lifecycle, see the *Matrix Business Administrator Guide* for your MatrixOne application.

If you rename objects that are referenced in policy expression accesses, you will need to update any expressions that reference the object names. This is true even if the expression uses symbolic names during installation because once the installation is complete, the symbolic names will have been replaced by the actual object names.

Admin Specification Section	eService Trigger Program	Program	Software Distribution
Application Part	Policy	Project Access List	Software Feature
Assessment	Event	Project Concept	Software Functional Specification
Attribute Group	File Format	Project Folder	Software Platform
Baseline Log	Financial Categories	Project Member	Software Release
Basic Specification Section	Financial Items	Project Release	Software Requirement Specification
Build	Holder	Project Risk	Software Task
Build Archive	HTTP Request	Project Space	Software Test Case
Business Goal	HTTP Request Holder	Project Task	Software Test Specification
Buyer Desk	Image	Project Template	Source Code
CAD Drawing	Impact	Public Folder	Source File
CAD Model	Inbox Task	Publish Subscribe	Sourcing ECR
Certification	Incident	Purchase Class	Sourcing Template
Configurable Feature	Interval Item Data	Purchase Class Scorecard	Obsolete
Configurable Product	Issue	Quality	Specification
Configuration File	Issue Categorization	Quality Metric	Specification Office
Container	Legacy Technical Specification	Question	Specification Template
Container Rev2	Line Item	Rate Period	Holder
Controlled Design Release	Line Item Template	Region	Standards Manual
Controlled Design Release Rev2	Location	Request For Quote	Substitute
Controlled Production Release	Location Equivalent	Requirement	Supplier Development Plan
Controlled Production Release Rev2	Manufacturer Equivalent	Response	Supplier Line Item
Conversion	Markup	Restricted Legacy	Supplier Line Item Price Period
CoOwned List	Media	Technical Specification	Supplier Scorecard
Data Document	Meeting	Restricted Technical Specification	Suspended
Deliverable	Member List	RFQ	Task Escalation Message
Development Part	Message	RFQ Cancelled	Technical Specification
Directed Supplier	Model	RFQ History Version	Technical Specification Template
Document	Option	RFQ Pending Version	Template Filter
Document Sheet	Organization	RFQ Quotation	Template Workspace Vault
Drawing Print	Organization List	RFQ Template	Test Case
Duplicate	Package	Risk	Thread
EBOM Markup	Part Family	Route	UDA
EC Part	Part Master	Route Task User	URL
ECO (Standard)	Part Quality Plan	Route Template	Use Case
ECR (Rejected)	Part Quality Plan Template	RPN	Version
ECR (Standard)	Person	RTS Quotation Archive	Version Document
ECR Supporting Document	Personal Folder	Rule	Viewable Policy
Effort	Plan	Sales Order	Workspace
eService Object Generator	Plant Scorecard	SCO	Workspace Template
	Process	Scorecard	Workspace Vaults
	Product	Software Architecture	
	Product Configuration	Software Build	
	Product Documentation	Software Design Specification	
	Product Feature		
	Product Line		

Admin Specification Section		
Revision	-	
Types	Specification Section	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Specification Manager	read,create,viewform,show	read,modify,create,fromconnect,tconnect,execute,viewform,show
Specification Restricted User	read,create,viewform,show	read,modify,create,fromconnect,tconnect,execute,viewform,show
Specification Office Manager	read,checkout,checkin,lock,unlock,create,fromconnect,toconnect,execute,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,toconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Global User	read,fromconnect,tconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwms].businessobject.name	read,modify,fromconnect,tconnect,todisconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwms].businessobject.name
Global Template Administrator	read,checkout,lock,unlock,create,fromconnect,tconnect,execute,show	read,modify,checkout,lock,unlock,create,fromconnect,tconnect,todisconnect,execute,show
Template Header Administrator	read,checkout,lock,unlock,create,fromconnect,tconnect,execute,show	read,modify,checkout,lock,unlock,create,fromconnect,tconnect,todisconnect,execute,show
Template Administrator	read,checkout,lock,unlock,create,fromconnect,tconnect,execute,show	read,modify,checkout,lock,unlock,create,fromconnect,tconnect,todisconnect,execute,show

Admin Specification Section

System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Application Part

Revision	A,B,C...		
Types	Application Part		
Hidden	FALSE		
States	Preliminary	Review	Complete
Revisionable	FALSE	FALSE	TRUE
Versionable	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show
Owner	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect
Senior Design Engineer	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect
Manufacturing Engineer	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect
Component Engineer	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect	read, show, create, from connect, to connect, from disconnect, to disconnect
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all

Assessment

Revision	
Types	Assessment
Hidden	FALSE
States	Exists
Revisionable	TRUE

Assessment	
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	fromconnect,fromdisconnect
Owner	read,modify,create,fromconnect,toconnect,fromdisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Public Read	read,checkout,show filter: program[emxAssessment PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true
Public Modify	modify,delete,changeowner,promote,demote,changename,change policy,fromconnect,fromdisconnect,todisconnect filter: program[emxAssessment PROJECT_ASSESSOR PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true

Attribute Group	
Revision	
Types	Attribute Group
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	toconnect,todisconnect
Owner	all
Supplier*	none
Buyer*	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
System Transition Manager	all
System Conversion Manager	all
*Accesses to the Buyer and Supplier roles are granted as-needed based on the value of the Line Item template object's User Symbolic Name and Line Item Template Scope attributes.	

Attribute Group	
Shadow Agent	all
Buyer Administrator	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Customer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	none
*Accesses to the Buyer and Supplier roles are granted as-needed based on the value of the Line Item template object's User Symbolic Name and Line Item Template Scope attributes.	

Baseline Log	
Revision	
Types	Baseline Log
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,modify,delete,create,toconnect,todisconnect,show
Owner	show
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Basic Specification Section		
Revision	-	
Types	Specification Section	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE

Basic Specification Section		
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	all
Specification Manager	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show
Specification Office Manager	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeaname,changepolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeaname,changepolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Global User	read,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwms].businessobject.name	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwms].businessobject.name
Global Template Administrator	read,checkout,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,modify,checkout,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show
Template Administrator	read,checkout,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,modify,checkout,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Build						
Revision	1,2,3,...					
Types	Build					
Hidden	FALSE					
States	Plan	Build	Internal Test	External Test	Complete	Release
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Build						
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none
Owner	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,revise,tconnect,todisconnect,execute,show	read,revise,tconnect,todisconnect,execute,show,promote,demote
System Conversion Manager	all	all	all	all	all	all
System Transition Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Employee	read,modify,promote,demote,fromconnect,tconnect,fromdisconnect,todisconnect,execute,,show	read,tconnect,todisconnect,execute,show	read,modify,promote,demote,fromconnect,tconnect,fromdisconnect,todisconnect,execute,show	read,modify,checkout,promote,demote,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,tconnect,execute,show	read,demote,revise,tconnect,execute,show
Company Name	read,checkout,tconnect,todisconnect,execute,viewform,show	read,tconnect,todisconnect,execute,show	none	read,tconnect,todisconnect,execute,show	read,tconnect,execute,show	read,tconnect,execute,show
Software Engineering Manager	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'	none	none	none	none	none

Build						
System Engineer	changeowner,create	read,toconnect,todisconnect,execute,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	read,promote,demote,revise,toconnect,todisconnect,execute,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	none
Software Engineer Manager	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	read,toconnect,todisconnect,execute,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	read,promote,demote,revise,toconnect,todisconnect,execute,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	none
Product Manager	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,toconnect,todisconnect,execute,show	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	changeowner,create filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,create,changeowner,promote,demote,revise,toconnect,todisconnect,execute,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	none
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	none

Build						
Software Release Manager	none	read,modify,delete,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant, revoke,show filter:program[emx Organization -method hasRDOrole -construct \${OBJECTID} role_SoftwareReleaseManager] == 'I'	read,modify,delete,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant, revoke,show filter:program[emx Organization -method hasRDOrole -construct \${OBJECTID} role_SoftwareReleaseManager] == 'I'	read,modify,delete,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant, revoke,show filter:program[emx Organization -method hasRDOrole -construct \${OBJECTID} role_SoftwareReleaseManager] == 'I'	read,modify,delete,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant, revoke,show filter:program[emx Organization -method hasRDOrole -construct \${OBJECTID} role_SoftwareReleaseManager] == 'I'	none
Global User	none	none	none	none	none	none

Build Archive			
Revision	1,2,3,...		
Types	Software Build		
Hidden	FALSE		
States	Request Archive	Archive	Obsolete
Revisionable	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show
Owner	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Business Goal			
Revision			
Types	Business Goal		
Hidden	FALSE		
States	Created	Active	Complete
Revisionable	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	none	none	none
Owner	all	all	read,checkout,promote,demote,show
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Employee	read,checkout,show	read,checkout,show	read,checkout,show
Business Manager	all	all	read,checkout,promote,demote,show

Buyer Desk	
Revision	-
Types	Buyer Desk
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Public	read,show
System Transition Manager	all
System Conversion Manager	all

Buyer Desk	
Shadow Agent	all
Buyer	all filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Buyer Administrator	all filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Customer	all filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]

CAD Drawing				
Revision	A,B,C,...			
Types	CAD Drawing			
Hidden	FALSE			
States	Preliminary	Review	Approved	Release
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	TRUE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changegetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,checkout,changeowner,demote,schedule,revise,changevault,execute,viewform,show	changeowner,demote,changevault,show	changeowner,changevault,show
Company Name	read,checkout,create,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all

CAD Drawing				
Supplier Representative	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierRepresentativ e] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierRepresentativ e] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierRepresentativ e] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierRepresentativ e] == '1'
Supplier Engineer	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierEngineer] == '1'
Supplier Quality Engineer	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierQualityEngine er] == '1'	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierQualityEngine er] == '1'	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierQualityEngine er] == '1'	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_SupplierQualityEngine er] == '1'
Employee	read,checkout,create,revise, toconnect,todisconnect,exec ute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1'	'read,checkout,revise,toconn ect,todisconnect,execute,vie wform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1'	'read,checkout,revise,toconn ect,todisconnect,execute,vie wform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1'	'read,checkout,revise,toconn ect,todisconnect,execute,vie wform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1'
Manufacturing Engineer	none	read,checkout,promote,dem ote,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_ManufacturingEnginee r] == '1'	'read,promote,demote,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_ManufacturingEnginee r] == '1'	'none

CAD Model				
Revision	A,B,C,...			
Types	CAD Model			
Hidden	FALSE			
States	Preliminary	Review	Approved	Release
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	TRUE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show

CAD Model				
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,checkout,changeowner,demote,schedule,revise,changevault,execute,viewform,show	changeowner,demote,changevault,show	changeowner,changevault,show
Company Name	read,checkout,create,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all
Supplier Representative	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierRepresentative] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierRepresentative] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierRepresentative] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierRepresentative] == '1'
Supplier Engineer	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierEngineer] == '1'
Supplier Quality Engineer	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierQualityEngineer] == '1'	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierQualityEngineer] == '1'	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierQualityEngineer] == '1'	read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SupplierQualityEngineer] == '1'
Employee	read,checkout,create,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_Employee] == '1'	'read,checkout,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_Employee] == '1'	'read,checkout,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_Employee] == '1'	'read,checkout,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_Employee] == '1'
Manufacturing Engineer	none	read,checkout,promote,demote,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ManufacturingEngineer] == '1'	'read,promote,demote,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ManufacturingEngineer] == '1'	'none

Certification	
Revision	-
Types	Certification
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,checkout,show
Owner	read,modify,checkout,checkin,changeowner,changevault,tocconnect,todisconnect,show
Quality Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Supplier Development Manager	toconnect,todisconnect
Buyer Administrator	toconnect,todisconnect

Characteristic		
Revision	0,1,2...	
Types	Characteristic-	
Hidden	FALSE	
States	Draft	Active
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none

Characteristic		
Owner	read, modify, delete, check out, check in, schedule, lock, execute, unlock, freeze, thaw, revise, promote, revise, grant, enable, disable, override, changename, changetype, changeowner, changepolicy, revoke, changevault, fromconnect, toconnect, fromdisconnect, todisconnect, viewform, modifyform, show	read, checkout,demote,show
Specification User	read, modify, checkin, checkout, lock, unlock, show	read, checkout, show
Specification Manager	read, modify, checkin, checkout, lock, unlock, create, show	read, checkout, show
Specification Office Manager	read, modify, delete, check out, check in, schedule, lock, execute, unlock, freeze, thaw, promote, grant, enable, disable, override, changename, changetype, changeowner, changepolicy, revoke, chnagevault, fromconnect, toconnect, fromdisconnect, todisconnect, viewform, modifyform, show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read, checkout, demote, show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Configurable Feature		
Revision	-	
Types	Feature	
Hidden	FALSE	
States	Preliminary	Release
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	read,show	read,show
Owner	all	changeowner,show
Employee	read,create,show	read,show
Company Name	read,create,show	read,show

Configurable Feature		
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Product Manager	all	read,modify,toconnect,todisconnect,show
Design Engineer	all	read,modify,toconnect,todisconnect,show

Configurable Product						
Revision	A,B,C,...					
Types	Configurable Product	Products				
Hidden	FALSE					
States	Preliminary	Product Management	Design Engineering	Review	Release	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none
Owner	all	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Employee	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,create,revise,show
Deprecated in version 10.0.1.0.						

Configurable Product						
Marketing Manager	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	none	none	none
Senior Design Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	none	none	none
Software Engineering Manager	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	none	none	none
Software Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	none	none	none
System Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,demote,revise,tconnect,todisconnect,execute,viewform,show
Company Name	read,create,revise,show	read,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Deprecated in version 10.0.1.0.						

Configurable Product						
Product Manager	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,demote,revise,tconnect,todisconnect,execute,viewform,show
Design Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,revise,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changevault,changeename,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeename,changeepolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeename,changeepolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	none
Customer	none	none	read,tconnect,show	none	read,fromconnect,fromdisconnect,show	none
Public Read	none	none	none	read,checkout,tconnect,todisconnect,execute,viewform,show	read,checkout,tconnect,todisconnect,execute,viewform,show filter: to[Company Product].businessobject.from[Employee].businessobject.name == context.user	none
Sales Engineer	none	none	none	none	read,fromconnect,fromdisconnect,show	none
Deprecated in version 10.0.1.0.						

Configuration File	
Revision	1,2,3,...
Types	Configuration File
Hidden	FALSE
States	Base
Revisionable	TRUE
Versionable	FALSE

Configuration File	
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Container					
Revision					
Types	DOCUMENT CLASSIFICATION				
Hidden	TRUE				
States	CREATE	WIP	REVIEW	RELEASED	OBSOLETE
Revisionable	FALSE	FALSE	FALSE	TRUE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	TRUE	FALSE	FALSE
Checkout History	FALSE	TRUE	TRUE	FALSE	FALSE
Public	read,checkout,show	read,checkout,show	read,checkout,show	read,checkout,revise,show	read,checkout,show
Owner	all	all	read,checkout,show	read,checkout,revise,show	read,checkout,show
Reviewer	read,checkout,show	read,checkout,show	all	none	none
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Deprecated and hidden in version 10.0.0.0.					

Container Rev2					
Revision	A-Z,AA-ZZ				
Types	DOCUMENT CLASSIFICATION				
Hidden	FALSE				
States	Create	Review	Approved	Locked	Obsolete
Revisionable	FALSE	FALSE	FALSE	TRUE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changename,change policy,changetype,toconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,checkout,grant,revoke,show	read,checkout,promote,demote,fromconnect,t oconnect,fromdisconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,revise,grant,revoke,show	read,checkout,promote,demote,grant,revoke,show
Access Grantor	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changename,change policy,changetype,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,checkout,grant,revoke,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	none	none
Reviewer	read,checkout,show	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changename,change policy,changetype,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	none	none	none
Shadow Agent	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
System Transition Manager	all	all	all	all	all

Container Rev2					
Employee	none	none	read,checkout,toconnect,show	read,checkout,show	read,checkout,show
Release Manager	none	none	none	read,checkout,revise,show	none

Controlled Design Release					
Revision					
Types	Generic Document				
Hidden	TRUE				
States	CREATE	WIP	REVIEW	RELEASED	OBSOLETE
Revisionable	FALSE	FALSE	FALSE	TRUE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	TRUE	FALSE	FALSE
Checkout History	FALSE	TRUE	TRUE	FALSE	FALSE
Public	read,checkout,show	read,checkout,show	read,checkout,show	read,checkout,revise,show	read,checkout,show
Owner	all	all	read,checkout,show	read,checkout,revise,show	read,checkout,show
Reviewer	read,checkout,show	read,checkout,show	all	none	none
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
<i>Deprecated and hidden in version 10.0.0.0.</i>					

Controlled Design Release Rev2					
Revision	A-Z				
Types	Generic Document				
Hidden	FALSE				
States	Create	WIP	Review	Released	Obsolete
Revisionable	FALSE	FALSE	FALSE	TRUE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE

Controlled Design Release Rev2					
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	read,checkout,grant,revoke,show	read,checkout,promote,demote,revise,toconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,grant,revoke,show
Access Grantor	all	all	read,checkout,grant,revoke,show	read,checkout,revise,tconnect,todisconnect,grant,revoke,show	read,checkout,grant,revoke,show
Public Delete	delete,fromdisconnect,todisconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromdisconnect,todisconnect,delete] ~~ true	delete,fromdisconnect,todisconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromdisconnect,todisconnect,delete] ~~ true	none	todisconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[ToDisconnect] ~~ true	none
Public Add	fromconnect,toconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromconnect,tconnect] ~~ true	fromconnect,toconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromconnect,tconnect] ~~ true	none	toconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[ToConnect] ~~ true	none
Public Modify	modify,checkin,lock,unlock,changeowner,promote,demote,changevault,changechange,changechange filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[checkin,modify,lock,unlock,promote,demote,changechange,changechange,changechange] ~~ true	modify,checkin,lock,unlock,changeowner,promote,demote,changevault,changechange,changechange filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[checkin,modify,lock,unlock,promote,demote,changechange,changechange,changechange] ~~ true	none	none	none
Public Read	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true
Reviewer	read,checkout,show	read,checkout,show	all	none	none
Shadow Agent	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
System Transition Manager	all	all	all	all	all

Controlled Design Release Rev2					
Public Remove	fromdisconnect,todisconnect	fromdisconnect,todisconnect	fromdisconnect,todisconnect	fromdisconnect,todisconnect	fromdisconnect,todisconnect
Release Manager	none	none	none	read,checkout,revise,show	none

Controlled Production Release					
Revision					
Types	Generic Document				
Hidden	TRUE				
States	CREATE	WIP	REVIEW	RELEASED	OBSOLETE
Revisionable	FALSE	FALSE	FALSE	TRUE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	TRUE	FALSE	FALSE
Checkout History	FALSE	TRUE	TRUE	FALSE	FALSE
Public	read,checkout,show	read,checkout,show	read,checkout,show	read,checkout,revise,show	read,checkout,show
Owner	all	all	read,checkout,show	read,checkout,revise,show	read,checkout,show
Reviewer	read,checkout,show	read,checkout,show	all	none	none
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
<i>Deprecated and hidden in version 10.0.0.0.</i>					

Controlled Production Release Rev2					
Revision	1,2,3,...				
Types	Generic Document				
Hidden	FALSE				
States	Create	WIP	Review	Released	Obsolete
Revisionable	FALSE	FALSE	FALSE	TRUE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE

Controlled Production Release Rev2					
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	read,checkout,grant,revoke,show	read,checkout,promote,demote,revise,toconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,grant,revoke,show
Access Grantor	all	all	read,checkout,grant,revoke,show	read,checkout,revise,tconnect,todisconnect,grant,revoke,show	read,checkout,grant,revoke,show
Public Delete	delete,fromdisconnect,todisconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromdisconnect,todisconnect,delete] ~~ true	delete,fromdisconnect,todisconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromdisconnect,todisconnect,delete] ~~ true	none	todisconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[ToDisconnect] ~~ true	none
Public Add	fromconnect,toconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromconnect,tconnect] ~~ true	fromconnect,toconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[fromconnect,tconnect] ~~ true	none	toconnect filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[ToConnect] ~~ true	none
Public Modify	modify,checkin,lock,unlock,changeowner,promote,demote,changevault,changechange,changechange filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[checkin,modify,lock,unlock,promote,demote,changechange,changechange,changechange,changechange] ~~ true	modify,checkin,lock,unlock,changeowner,promote,demote,changevault,changechange,changechange filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[checkin,modify,lock,unlock,promote,demote,changechange,changechange,changechange,changechange] ~~ true	none	none	none
Public Read	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true	read,checkout,show filter: attribute[AccessType] ~~ Inherited AND relationship[Has Documents].from.current.access[read,show,checkout] ~~ true
Reviewer	read,checkout,show	read,checkout,show	all	none	none
Shadow Agent	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
System Transition Manager	all	all	all	all	all
Release Manager	none	none	none	read,checkout,revise,show	none

Conversion	
Revision	
Types	Currency Conversion Unit Conversion
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Supplier	show
Buyer	show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer Administrator	read,modify,create,fromconnect,toconnect,fromdisconnect,todisconnect

CoOwned List	
Revision	-
Types	CoOwned List
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all

CoOwned List	
Employee	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Specification Manager	read,checkout,checkin,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show
Specification Office Manager	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeName,changePolicy,changeType,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Global User	read,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwNs].businessobject.name
Global Template Administrator	read,checkout,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show
Template Administrator	read,checkout,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Data Document				
Revision	-			
Types	Data Document			
Hidden	FALSE			
States	Not Submitted	Submitted	In Process	Completed
Revisionable	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show
Owner	all	all	all	all

Data Document				
Customer	all	read,modify,checkout,check in,lock,unlock,changeowner ,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,from connect,toconnect,fromdisc onnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,fromconnect,toconnect ,fromdisconnect,todisconnect,show	read,fromconnect,toconnect ,fromdisconnect,todisconnect,show
Customer Representative	all	read,modify,checkout,check in,lock,unlock,changeowner ,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,from connect,toconnect,fromdisc onnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,fromconnect,toconnect ,fromdisconnect,todisconnect,show	read,fromconnect,toconnect ,fromdisconnect,todisconnect,show
Sales Engineer	read,fromconnect,toconnect ,fromdisconnect,todisconnect,show	read,modify,checkout,check in,lock,unlock,changeowner ,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,from connect,toconnect,fromdisc onnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,check in,lock,unlock,changeowner ,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,check in,lock,unlock,changeowner ,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show
Shadow Agent	all	all	all	all
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all

Deliverable				
Revision	1,2,3,...			
Types	Deliverable			
Hidden	FALSE			
States	Draft	Active	Review	Complete
Revisionable	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none
Owner	read,modify,changeowner,s how	read,modify,changeowner,s how	read,modify,changeowner,s how	all

Deliverable				
Supplier Development Manager	all	all	all	all
Advanced Quality Engineer	all	all	all	all
Quality Manager	all	all	all	all
Shadow Agent	all	all	all	all
System Conversion Manager	all	all	all	all
System Transition Manager	all	all	all	all
Company Name	read,checkout,show	read,checkout,show	read,checkout,show	read,checkout,show
Employee	read,checkout,show	read,checkout,show	read,checkout,show	read,checkout,show
Supplier Representative	none	read,checkout,checkin,promote,show	read,checkout,show	read,checkout,show
Supplier Engineer	none	read,checkout,checkin,show	read,checkout,show	read,checkout,show
Buyer	none	read,modify,checkout,checkin,lock,unlock,show	read,modify,checkout,checkin,lock,unlock,show	none

Development Part			
Revision	A,B,C,...		
Types	Part		
Hidden	FALSE		
States	Create	Peer Review	Complete
Revisionable	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	none	none	none
Owner	all	all	all
Company Name	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify
Senior Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify
Senior Manufacturing Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify

Development Part			
Organization Manager	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_OrganizationManager] == '1'	'read,changeowner,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_OrganizationManager] == '1'	'read,changeowner,demote,revise,changevault,changetype,changepolicy,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_OrganizationManager] == '1'
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Buyer	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show
Employee	read,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1'	'read,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1'	'read,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1'
Supplier	read,show	read,show	read,show
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show

Directed Supplier	
Revision	-
Types	Directed Supplier
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Supplier Development Manager	all
Buyer Administrator	all
Quality Manager	all
Shadow Agent	all
System Conversion Manager	all

Directed Supplier	
System Transition Manager	all
Company Name	read,show
Employee	read,show

Document	
Revision	0,1,2,...
Types	Document
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	create
Owner	all
Shadow Agent	all
System Transition Manager	all
System Conversion Manager	all
Buyer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key] attribute[Secondary Keys] == " context.user.property[Company Key].value ~~ attribute[Secondary Keys]
Supplier	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key] attribute[Secondary Keys] == " context.user.property[Company Key].value ~~ attribute[Secondary Keys]
Access Grantor	read,modify,delete,checkout,checkin,lock,unlock,revise,from connect,toconnect,fromdisconnect,todisconnect,grant,revoke, show
Workspace Lead Grantor	promote,demote
Public Modify	fromconnect filter: current.access[modify] ~~ true
Public Read	read,checkout,show filter: ((attribute[Access Type] == Inherited) && (to[Vaulted Documents Rev2].from.current.access[checkout] ~~ true)) (to[Company Documents].from.to[Document Holder].from.to[Supplier].from.attribute[Primary Key].value ~~ context.user.property[Company Key].value)

Document	
Buyer Administrator	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key] attribute[Secondary Keys] == " context.user.property[Company Key].value ~~ attribute[Secondary Keys]
Global User	read,modify,checkout,checkin,lock,unlock,revise,show filter: (((to[Message Attachments].id != ") (to[Meeting Attachments].id != ") && ((to[Message Attachments].from.type == 'Message' && to[Message Attachments].from.current.access[read] == true && (!to[Vaulted Objects].id != ") (to[Meeting Attachments].from.type == 'Meeting' && to[Meeting Attachments].from.current.access[read] == true && (!to[Vaulted Objects].id != ")"))))
Customer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Employee	read, checkout, execute, show
Supplier Modify	read, modify, checkout, checkin, lock, execute, unlock, revise, and show. filter: (to[Task Deliverable].from.to[Project Access Key].from.from[Project Access List].to.type ~~ "Part Quality Plan") && (to[Task Deliverable].from.to[Assigned Tasks].from.name ~~ context.user)
Sales Engineer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Issue Manager	read, modify, toconnect, fromconnect, show

Document Sheet			
Revision	A-Z		
Types	Document Sheet		
Hidden	FALSE		
States	WIP	ACTIVE	IN-ACTIVE
Revisionable	FALSE	TRUE	FALSE
Versionable	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	FALSE	FALSE	FALSE
Public	read,checkout,show	read,checkout,revise,show	read,checkout,show
Owner	all	read,checkout,revise,show	read,checkout,show
Reviewer	all	none	none

Document Sheet			
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all

Drawing Print				
Revision	A,B,C,...			
Types	Drawing Print			
Hidden	FALSE			
States	Preliminary	Review	Approved	Release
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	TRUE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,checkout,changeowner,demote,schedule,revise,changevault,execute,viewform,show	changeowner,demote,changevault,show	changeowner,changevault,show
Supplier Representative	read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierRepresentative] == '1'	'read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierRepresentative] == '1'	'read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierRepresentative] == '1'	'read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierRepresentative] == '1'
Supplier Engineer	read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierEngineer] == '1'	'read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierEngineer] == '1'
Supplier Quality Engineer	read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierQualityEngineer] == '1'	read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierQualityEngineer] == '1'	read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierQualityEngineer] == '1'	read,checkout,show filter: program[emxOrganization-method hasRDOrole-construct \${OBJECTID}\role_SupplierQualityEngineer] == '1'

Drawing Print				
Employee	read,checkout,create,revise,t oconnect,todisconnect,execut e,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1	'read,checkout,revise,toconne ct,todisconnect,execute,view form,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1	'read,checkout,revise,toconne ct,todisconnect,execute,view form,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1	'read,checkout,revise,toconne ct,todisconnect,execute,view form,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_Employee] == '1
Company Name	read,checkout,create,revise,t oconnect,todisconnect,execut e,viewform,show	read,checkout,revise,toconne ct,todisconnect,execute,view form,show	read,checkout,revise,toconne ct,todisconnect,execute,view form,show	read,checkout,revise,toconne ct,todisconnect,execute,view form,show
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all
Manufacturing Engineer	none	read,checkout,promote,demo te,schedule,execute,viewfor m,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_ManufacturingEngineer] == '1	'read,promote,demote,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\ role_ManufacturingEngineer] == '1	'none

Duplicate		
Revision	0,1,2,3,...	
Types	Duplicate	
Hidden	FALSE	
States	Opened	Closed
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	read,modify,changeowner,promote, create,fromconnect,toconnect,from disconnect,todisconnect,viewform,s how	read,delete,changeowner,fromdisco nnect,todisconnect,viewform,show
Company Name	read,modify,changeowner,promote, create,fromconnect,toconnect,from disconnect,todisconnect,viewform,s how	read,delete,changeowner,fromdisco nnect,todisconnect,viewform,show
System Transition Manager	all	all

Duplicate		
System Conversion Manager	all	all
Shadow Agent	all	all
Employee	read,modify,changeowner,promote,create,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,delete,changeowner,fromdisconnect,todisconnect,viewform,show

EBOM Markup				
Revision	-			
Types	EBOM Markup			
Hidden	FALSE			
States	Work In Progress	Review	Approved	Rejected
Revisionable	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	all	all	all	all
Owner	all	all	all	all
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all

EC Part					
Revision	1,2,3,...				
Types	Part				
Hidden	FALSE				
States	Preliminary	Review	Approved	Release	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	TRUE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE

EC Part					
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,checkout,changeowner,demote,schedule,revise,changevault,execute,viewform,show	changeowner,demote,changevault,show	changeowner,changevault,show	changeowner,changevault,show
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show
Design Engineer	read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_DesignEngineer] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_DesignEngineer] == '1'	'none filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_DesignEngineer] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_DesignEngineer] == '1'	'read,revise,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_DesignEngineer] == '1'
Senior Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	'read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify
Senior Manufacturing Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	'read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify
Organization Manager	read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_OrganizationManager] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_OrganizationManager] == '1'	'none filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_OrganizationManager] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_OrganizationManager] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_OrganizationManager] == '1'
Buyer	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show
Buyer Administrator	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show

EC Part					
Employee	read,checkout,create,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1	'read,checkout,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1	'read,checkout,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1	'read,checkout,revise,toconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1	'read,checkout,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_Employee] == '1'
Company Name	read,checkout,create,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,revise,toconnect,todisconnect,execute,viewform,show	read,checkout,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Manufacturing Engineer	none	read,modify,checkout,promote,demote,schedule,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ManufacturingEngineer] == '1	'read,promote,demote,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ManufacturingEngineer] == '1	'none	none
Advanced Quality Engineer	none	none	read, checkout, execute, toconnect, todisconnect, viewform, show	read, checkout, execute, toconnect, todisconnect, viewform, show	none

EC Part					
Supplier	none	none	none	read,show filter: context.user ~~ to[Manufacturing Responsibility].from.fr om[Employee].to.name context.user ~~ to[Design Responsibility].from.fr om[Employee].to.name context.user ~~ to[Testing Responsibility].from.fr om[Employee].to.name context.user ~~ to[Supply Responsibility].from.fr om[Employee].to.name context.user ~~ to[Line Item Object].from.to[Suppli er Buyer Line Item].from.to[Supplier Line Item].from.from[Suppli er Response].to.from[Em ployee].to.name	read,show filter: context.user ~~ to[Manufacturing Responsibility].from.fr om[Employee].to.name context.user ~~ to[Design Responsibility].from.fr om[Employee].to.name context.user ~~ to[Testing Responsibility].from.fr om[Employee].to.name context.user ~~ to[Supply Responsibility].from.fr om[Employee].to.name context.user ~~ to[Line Item Object].from.to[Suppli er Buyer Line Item].from.to[Supplier Line Item].from.from[Suppli er Response].to.from[Em ployee].to.name
Product Obsolescence Manager	none	none	none	promote filter: program[emxOrganizat ion -method hasRDOrole -construct \$\{OBJECTID\} role_ProductObsolesce nceManager] == 'I'	'demote filter: program[emxOrganizat ion -method hasRDOrole -construct \$\{OBJECTID\} role_ProductObsolesce nceManager] == 'I'

ECO (Standard)							
Revision	-						
Types	ECO						
Hidden	FALSE						
States	Create	Define Components	Design Work	Review	Release	Implemented	Cancelled
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show	read	read,show

ECO (Standard)							
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,changevault,execute,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,changevault,execute,modifyform,viewform,show	changeowner,changevault,show	read,changeowner,demote,changevault,show	read,show
Employee	read,checkout,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1	'read,checkout,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1	'read,checkout,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1	'read,checkout,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1	'read,checkout,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1	'read,checkout,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1	'none
Company Name	read,checkout,execute,viewform,show	read,checkout,execute,viewform,show	read,checkout,execute,viewform,show	read,checkout,execute,viewform,show	read,checkout,viewform,show	read,checkout,show	none
ECR Coordinator	changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRCoordinator] == '1	'changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRCoordinator] == '1	'changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRCoordinator] == '1	'changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRCoordinator] == '1	'none	none	none
ECR Chairman	changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRChairman] == '1	'changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRChairman] == '1	'changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRChairman] == '1	'changeowner,fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_ECRChairman] == '1	'none	none	none
System Transition Manager	all	all	all	all	all	all	all

ECO (Standard)							
System Conversion Manager	all	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all	all

ECR (Rejected)	
Revision	-
Types	ECR
Hidden	FALSE
States	Rejected
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	read,modify,delete,checkout,changeowner,changevault,change policy,fromdisconnect,todisconnect,execute,modifyform,viewform,show
Access Grantor	read,grant,revoke,show
Employee	read,checkout,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_Employee] == '1'
Company Name	read,checkout,execute,viewform,show
ECR Coordinator	read,modify,delete,checkout,changeowner,change policy,from disconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_ECRCoordinator] == '1'
ECR Chairman	read,modify,delete,checkout,changeowner,change policy,from disconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_ECRChairman] == '1'
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

ECR (Standard)	
Revision	-
Types	ECR
Hidden	FALSE

ECR (Standard)						
States	Create	Submit	Evaluate	Review	Plan ECO	Complete
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show	read,show
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,create,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,promote,demote,schedule,changevault,change policy,execute,viewform,show	read,checkout,changeowner,promote,changevault,toconnect,todisconnect,execute,viewform,show	changeowner,changevault,show
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show
Employee	read,checkout,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1'	'read,checkout,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1'	'read,checkout,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1'	'read,checkout,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1'	'read,checkout,tconnect,todisconnect,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1'	'read,checkout,tconnect,todisconnect,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\role_Employee] == '1'
Company Name	read,checkout,tconnect,todisconnect,execute,viewform,show	read,checkout,tconnect,todisconnect,execute,viewform,show	read,checkout,tconnect,todisconnect,execute,viewform,show	read,checkout,tconnect,todisconnect,execute,viewform,show	read,checkout,tconnect,todisconnect,viewform,show	read,checkout,tconnect,todisconnect,viewform,show
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all

ECR (Standard)						
ECR Coordinator	none	read,modify,changeowner,promote,demote,schedule,override,fromconnect,fromdisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRCoordinator] == '1	'read,modify,changeowner,promote,demote,schedule,override,fromconnect,fromdisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRCoordinator] == '1	'read,modify,changeowner,promote,demote,schedule,change policy,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRCoordinator] == '1	'read,changeowner, demote,schedule,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRCoordinator] == '1	'none
ECR Chairman	none	fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRChairman] == '1	'fromconnect,fromdisconnect filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRChairman] == '1	'read,modify,changeowner,promote,demote,schedule,override,change policy,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRChairman] == '1	'read,changeowner, demote,schedule,show filter: program[emxOrganization -method hasRDOrole -construct \$\{OBJECTID\} role_ECRChairman] == '1	'none
Change Boards	none	none	none	modify,promote	none	none

ECR Supporting Document		
Revision	-	
Types	Sketch	
Hidden	FALSE	
States	Preliminary	Complete
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	read,show	read,show
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,create,changevault,toconnect,todisconnect,execute,modifyform,viewform,show	changeowner,changevault,show
Employee	read,checkout,create,execute,viewform,show	read,checkout,create,execute,viewform,show
Company Name	read,checkout,create,execute,viewform,show	read,checkout,create,execute,viewform,show
System Transition Manager	all	all

ECR Supporting Document		
System Conversion Manager	all	all
Shadow Agent	all	all

Effort	
Revision	-
Types	Effort
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Employee	read,modify,create,toconnect,todisconnect,show
Company Name	read,modify,create,toconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

eService Object Generator	
Revision	
Types	eService Object Generator eService Number Generator
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Administration Manager	all

eService Object Generator	
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

eService Trigger Program Policy		
Revision		
Types	eService Trigger Program Parameters	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	read,show	read,show
Owner	all	all
Administration Manager	all	all
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Event		
Revision		
Types	Event	
Hidden	FALSE	
States	Active	Inactive
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	all	all
Owner	all	all

Event		
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

File Format	
Revision	-
Types	File Format
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Buyer	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Supplier Representative	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Organization Manager	toconnect,todisconnect
Customer Representative	toconnect,todisconnect
Company Name	read,show
Supplier Engineer	read,show
Supplier Development Manager	read,toconnect,todisconnect,show
Buyer Administrator	read,toconnect,todisconnect,show
Employee	read,show

Financial Categories	
Revision	
Types	Financial Cost Category Financial Benefit Category
Hidden	FALSE

Financial Categories	
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	read,modify,delete,changeowner,promote,demote,changevault,changechange,changechange,fromconnect,toconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Company Name	read,show
Project Administrator	modify,delete,changeowner,promote,demote,create,changevault,changechange,changechange,fromconnect,toconnect,fromdisconnect,todisconnect filter: program[emxFinancialTemplateCategory -method hasAccess -construct \${OBJECTID\}] == true
Employee	read,show

Financial Items		
Revision		
Types	Financial Item Cost Item Benefit Item	
Hidden	FALSE	
States	Plan	Plan Frozen
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	read,modify,create,fromconnect,toc onnect,show	show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Financial Items		
Public Read	read,checkout,show filter: program[emxFinancialItem PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,show filter: program[emxFinancialItem PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true
Public Modify	modify,delete,changeowner,promote, demote,changevault,change name,change policy,fromconnect,toconnect,fromdisconnect,todisconnect filter: program[emxFinancialItem FINANCIAL_REVIEWER PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	modify,changeowner,changevault,change name,fromconnect,toconnect filter: program[emxFinancialItem FINANCIAL_REVIEWER PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true
Public Delete	none	promote,demote filter: program[emxFinancialItem FINANCIAL_REVIEWER -method hasAccess -construct \${OBJECTID\}] == true

Holder	
Revision	
Types	Holder
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer	read,modify,create,fromconnect,toconnect,fromdisconnect,show
Supplier	read,modify,create,fromconnect,toconnect,fromdisconnect,show
Buyer Administrator	read,modify,create,fromconnect,toconnect,fromdisconnect,show
Customer	read,modify,create,fromconnect,toconnect,fromdisconnect,show

Holder	
Sales Engineer	read,modify,create,fromconnect,toconnect,fromdisconnect,show
Design Engineer	read,modify,create,fromconnect,toconnect,fromdisconnect,show
ECR Coordinator	read,modify,create,fromconnect,toconnect,fromdisconnect,show
ECR Chairman	read,modify,create,fromconnect,toconnect,fromdisconnect,show
ECR Evaluator	read,modify,create,fromconnect,toconnect,fromdisconnect,show

HTTP Request	
Revision	-
Types	HTTP Request
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Global User	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

HTTP Request Holder	
Revision	-
Types	HTTP Request Holder
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE

HTTP Request Holder	
Checkout History	TRUE
Public	none
Owner	all
Global User	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Image	
Revision	-
Types	Image
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Product Manager	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show
System Engineer	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show
Software Engineer Manager	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show
Marketing Manager	read,modify,checkout,checkin,lock,unlock,changeowner,create,fromconnect,fromdisconnect,execute,viewform,show
Global User	read,checkout,show
Company Name	read,checkout,toconnect,todisconnect,execute,viewform,show
Employee	read,checkout,toconnect,todisconnect,execute,viewform,show,fromconnect, fromdisconnect
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Impact		
Revision	A,B,C,...	
Types	Impact Analysis	
Hidden	FALSE	
States	Create	Complete
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	read,demote,viewform,show
Company Name	read,modify,delete,promote,create,changename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,modify,checkout,changename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Employee	read,viewform,show	read,viewform,show
System Engineer	read,viewform,show	read,viewform,show
Product Manager	read,viewform,show	read,viewform,show
Global User	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changename,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Impact Analysis].businessobject.to[Assigned Incident].businessobject.name context.user ~~ to[Impact Analysis].businessobject.to[Governed By].businessobject.owner	read,demote,viewform,show filter: context.user ~~ to[Impact Analysis].businessobject.to[Assigned Incident].businessobject.name context.user ~~ to[Impact Analysis].businessobject.to[Governed By].businessobject.owner

Inbox Task			
Revision			
Types	Inbox Task		
Hidden	FALSE		
States	Assigned	Review	Complete
Revisionable	FALSE	TRUE	FALSE
Versionable	TRUE	TRUE	TRUE

Inbox Task			
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	all	all	all
Owner	all	all	all
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all

Incident							
Revision	0,1,2,3,...						
Types	Incident						
Hidden	FALSE						
States	Submit	Evaluate	Review	Assign	Implement	Validate	Close
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none	none
Owner	all	all	all	all	all	read,checkout,promote,demote,fromconnect,viewform,show	read,checkout,fromconnect,viewform,show
Company Name	read,modify,delete,checkout,checkin,changeowner,promote,create,changename,change policy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,viewform,show	read,checkout,fromconnect,viewform,show	read,checkout,fromconnect,tocconnect,todisconnect,viewform,show	read,checkout,fromconnect,tocconnect,todisconnect,viewform,show	read,checkout,fromconnect,tocconnect,todisconnect,viewform,show	read,checkout,fromconnect,tocconnect,todisconnect,viewform,show	read,checkout,fromconnect,tocconnect,todisconnect,viewform,show
System Transition Manager	all	all	all	all	all	all	all
<i>This lifecycle contains several branches and signature requirements.</i>							

Incident							
System Conversion Manager	all	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all	all
Employee	checkout,promote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform	read,checkout,fromconnect,fromdisconnect,viewform,show	read,checkout,fromconnect,fromdisconnect,viewform,show	read,checkout,fromconnect,fromdisconnect,viewform,show	read,checkout,fromconnect,fromdisconnect,viewform,show	read,checkout,fromconnect,fromdisconnect,viewform,show	read,checkout,fromconnect,viewform,show
Incident Entry Clerk	checkout,promote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform	none	none	none	none	none	none
Global User	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,fromdisconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,fromdisconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,fromdisconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,fromdisconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,fromdisconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,fromdisconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name	read,checkout,fromconnect,viewform,show filter: context.user ~~~to[Assigned Incident].businessobject.name context.user ~~~to[Governed By].businessobject.owner context.user ~~~to[Governed By].businessobject.name
This lifecycle contains several branches and signature requirements.							

Incident							
Product Manager	read,modify,delete,checkout,checkin,changeowner,promote,create,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	toconnect,todisconnect,fromconnect,fromdisconnect	nonetoconnect,todisconnect,fromconnect,fromdisconnect	nonetoconnect,todisconnect,fromconnect,fromdisconnect	nonetoconnect,todisconnect,fromconnect,fromdisconnect	nonetoconnect,todisconnect,fromconnect,fromdisconnect	none
System Engineer	read,modify,delete,checkout,checkin,changeowner,promote,create,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	fromconnect,fromdisconnect	fromconnect,fromdisconnect	fromconnect,fromdisconnect	fromconnect,fromdisconnect	fromconnect,fromdisconnect	none
Software Quality Engineering	none	read,modify,checkout,checkin,show	none	read,modify,checkout,checkin,show	read,modify,checkout,checkin,show	read,modify,checkout,checkin,changeowner,promote,demote,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,checkout,demote,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show
Software CCB Coordinator	none	read,modify,checkout,checkin,promote,demote,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,checkout,checkin,promote,demote,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	none	none	none	none
Software Project Manager	none	read,modify,checkout,checkin,changeowner,promote,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	none	none	none	none	none

This lifecycle contains several branches and signature requirements.

Incident							
Change Boards	none	none	read,modify,checkout,checkin,changeowner,promote,revise,change name,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	none	none	none	none
Software Change Board	none	none	read,modify,checkout,checkin,changeowner,promote,demote,revise,change name,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	none	none	none	none
Software Engineering	none	none	none	read,checkout,promote,viewform,show	read,modify,checkout,checkin,changeowner,promote,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,viewform,show	none	none
Documentation	none	none	none	read,checkout,viewform,show	read,modify,checkout,checkin,changeowner,promote,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,viewform,show	none	none
Software Quality Engineer	none	none	none	none	none	none	modify
This lifecycle contains several branches and signature requirements.							

Interval Item Data	
Revision	
Types	Interval Item Data
Hidden	FALSE

Interval Item Data	
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	all
Owner	read,modify,changeowner,promote,demote,changevault,change name,change policy,fromconnect,toconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Company Name	read,show
Employee	read,show

Issue					
Revision	—				
Types	Issue				
Hidden	FALSE				
States	Create	Assign	Active	Review	Closed
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all except demote	all except delete, change policy, change vault, changetype	all except delete, change policy, change vault, changetype	all except delete, change policy, change vault, changetype	read,show, demote
Employee	read, show, create, fromconnect, toconnect	all except delete, changeowner, changetype filter: "context.user == to\[Assigned Issue\].from.name"	all except delete, change policy, changeowner, changetype filter: "context.user == to\[Assigned Issue\].from.name"	all except delete, change policy, changeowner, changetype filter: "context.user == to\[Assigned Issue\].from.name"	read,show
Issue Manager	all except demote	all except delete, change policy, change vault, changetype	all except delete, change policy, change vault, changetype	all except delete, change policy, change vault, changetype	read,show

Issue					
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all

Issue Categorization	
Revision	—
Types	Issue Category, Issue Classification
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	FALSE
Public	all
Owner	all
Issue Manager	all
Employee	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer	read,modify,delete,create,revise,fromconnect,toconnect,from disconnect,todisconnect,show filter: ((to[Line Item].from.current.access[modify] == true && to[Line Item].from.current.access[fromconnect] == true && to[Line Item].from.current.access[fromdisconnect] == true) (to[Line Item Split].from.to[Line Item].from.current.access[modify] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromconnect] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromdisconnect] == true)) && (context.user.property[Company Key].value == attribute[Primary Key] attribute[Primary Key] == "")

Issue Categorization	
Access Grantor	read,grant,revoke,show
Customer	read,modify,delete,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: ((to[Line Item].from.current.access[modify] == true && to[Line Item].from.current.access[fromconnect] == true && to[Line Item].from.current.access[fromdisconnect] == true) (to[Line Item Split].from.to[Line Item].from.current.access[modify] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromconnect] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromdisconnect] == true)) && (context.user.property[Company Key].value == attribute[Primary Key] attribute[Primary Key] == ")

Legacy Technical Specification		
Revision	-	
Types	Technical Specification	
Hidden	FALSE	
States	Issued	Obsolete
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	read,checkout,checkin,lock,unlock,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,checkout,checkin,lock,unlock,fromconnect,toconnect,fromdisconnect,todisconnect,show
Specification Manager	read,checkout,checkin,toconnect,todisconnect,show	read,checkout,checkin,lock,unlock,toconnect,todisconnect,show
Specification Office Manager	delete,checkout,checkin,changeowner,promote,create,changevault,changepolicy filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	delete,checkout,checkin,changeowner,promote,create,changevault,changepolicy filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Global User	read,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwens].businessobject.name	read,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~ to[CoOwned].businessobject.to[CoOwens].businessobject.name
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
<i>This policy contains an admin property called PolicyClassification with the value Legacy.</i>		

Legacy Technical Specification		
Supplier	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobject.from[Employee].businessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobject.from[Employee].businessobject.name
Product Manager	fromconnect,toconnect,fromdisconnect,todisconnect	none
System Engineer	fromconnect,fromdisconnect	none
This policy contains an admin property called PolicyClassification with the value Legacy.		

Line Item	
Revision	1,2,3,...
Types	Line Item
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,create,show
Owner	modify,delete,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer	read,modify,delete,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: ((to[Line Item].from.current.access[modify] == true && to[Line Item].from.current.access[fromconnect] == true && to[Line Item].from.current.access[fromdisconnect] == true) (to[Line Item Split].from.to[Line Item].from.current.access[modify] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromconnect] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromdisconnect] == true)) && (context.user.property[Company Key].value == attribute[Primary Key] attribute[Primary Key] == "")

Line Item	
Access Grantor	read,grant,revoke,show
Customer	read,modify,delete,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: ((to[Line Item].from.current.access[modify] == true && to[Line Item].from.current.access[fromconnect] == true && to[Line Item].from.current.access[fromdisconnect] == true) (to[Line Item Split].from.to[Line Item].from.current.access[modify] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromconnect] == true && to[Line Item Split].from.to[Line Item].from.current.access[fromdisconnect] == true)) && (context.user.property[Company Key].value == attribute[Primary Key] attribute[Primary Key] == ")

Line Item Template		
Revision	1,2,3,...	
Types	Line Item Template	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	changenname,show	show
Buyer Administrator	read,modify,delete,promote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,demote,revise,fromconnect,tonnect,fromdisconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Buyer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	none	read,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Line Item Template		
Customer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	none	read,show

Location		
Revision	-0,1,2,....	
Types	Location	
Hidden	FALSE	
States	Inactive	Active
Revisionable	TRUE	TRUE
Versionable	TRUE	FALSE
Auto Promote	TRUE	FALSE
Checkout History	TRUE	TRUE
Public	read,show	read,show
Owner	all	all
Senior Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_SeniorDesignEngineer] == '1	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_SeniorDesignEngineer] == '1
Senior Manufacturing Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_SeniorManufacturingEngineer] == '1	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_SeniorManufacturingEngineer] == '1
Buyer	all	all
Supplier Representative	all	all
Customer Representative	all	all
Organization Manager	all	all
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Supplier Development Manager	all	all

Location		
Plant Representative	fromconnect,fromdisconnect	fromconnect,fromdisconnect
Buyer Administrator	all	all
Advanced Quality Engineer	fromconnect,fromdisconnect	fromconnect,fromdisconnect

Location Equivalent	
Revision	None
Types	Location Equivalent
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	read, show, create, from connect, to connect, from disconnect, to disconnect
Senior Design Engineer	read, show, create, from connect, to connect, from disconnect, to disconnect
Manufacturing Engineer	read, show, create, from connect, to connect, from disconnect, to disconnect
Component Engineer	read, show, create, from connect, to connect, from disconnect, to disconnect
Organization Manager	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Plant Representative	fromconnect,fromdisconnect

Manufacturer Equivalent					
Revision	1,2,3,...				
Types	Part				
Hidden	FALSE				
States	Preliminary	Review	Approved	Release	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE

Manufacturer Equivalent					
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changegetype,fromconnect,tconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,checkout,changeowner,schedule,changevault,execute,viewform,show	read,checkout,changeowner,schedule,changevault,execute,viewform,show	changeowner,show	changeowner,changevault,show
Company Name	read,checkout,revise,tconnect,todisconnect,execute,viewform,show	read,checkout,revise,tconnect,todisconnect,execute,viewform,show	read,checkout,revise,tconnect,todisconnect,execute,viewform,show	read,checkout,tconnect,todisconnect,execute,viewform,show	read,checkout,todisconnect,execute,viewform,show
Employee	read,checkout,revise,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_Employee] == '1'	read,checkout,revise,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_Employee] == '1'	read,checkout,revise,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_Employee] == '1'	read,checkout,tconnect,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_Employee] == '1'	read,checkout,todisconnect,execute,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID\} role_Employee] == '1'
Buyer Administrator	read,tconnect,todisconnect,show	read,tconnect,todisconnect,show	read,tconnect,todisconnect,show	read,tconnect,todisconnect,show	read,tconnect,todisconnect,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show
Senior Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify
Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify

Manufacturer Equivalent					
Senior Manufacturing Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	'read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify	read, show, create, fromconnect, toconnect, fromdisconnect, to disconnect, revise, modify
Organization Manager	read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_OrganizationManager] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_OrganizationManager] == '1'	'read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_OrganizationManager] == '1'	'read,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_OrganizationManager] == '1'	'read,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_OrganizationManager] == '1'
Buyer	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show
Component Engineer	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_ComponentEngineer] == '1'	'read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_ComponentEngineer] == '1'	'read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,create,revise,changevault,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_ComponentEngineer] == '1'	'read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,schedule,revise,changevault,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_ComponentEngineer] == '1'	'read,modify,delete,checkout,checkin,lock,unlock,demote,schedule,revise,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID}\ role_ComponentEngineer] == '1'

Manufacturer Equivalent					
Product Obsolescence Manager	none	none	none	read,promote,show filter: program[emxOrganiza tion -method hasRDOrole -construct \${OBJECTID\ role_ProductObsolesc enceManager] == '1	'read,demote,show filter: program[emxOrganiza tion -method hasRDOrole -construct \${OBJECTID\ role_ProductObsolesc enceManager] == '1'
Supplier	none	none	none	read,show filter: context.user ~~ to[Manufacturing Responsibility].from.f rom[Employee].to.na me context.user ~~ to[Design Responsibility].from.f rom[Employee].to.na me context.user ~~ to[Testing Responsibility].from.f rom[Employee].to.na me context.user ~~ to[Supply Responsibility].from.f rom[Employee].to.na me context.user ~~ to[Line Item Object].from.to[Suppli er Buyer Line Item].from.to[Supplier Item].from.from[Supp lier Response].to.from[Em ployee].to.name	none

Markup	
Revision	
Types	Markup
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	all
Owner	all
System Transition Manager	all

Markup	
System Conversion Manager	all
Shadow Agent	all

Media			
Revision	1,2,3,....		
Types	Media		
Hidden	FALSE		
States	Create	Validate	Complete
Revisionable	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show
Owner	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Meeting				
Revision				
Types	Meeting			
Hidden	FALSE			
States	Create	Scheduled	In Progress	Complete
Revisionable	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE

Meeting				
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	all	all	all	all
Owner	all	all	all	all
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all
Project Lead	all	none	none	none

Member List		
Revision	-	
Types	Member List	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	all
Specification Manager	read,create,fromconnect,toconnect,f romdisconnect,todisconnect,execut e,viewform,show	read,create,fromconnect,toconnect,f romdisconnect,todisconnect,execut e,viewform,show
Specification Office Manager	read,create,fromconnect,toconnect,f romdisconnect,todisconnect,execut e,viewform,show	read,create,fromconnect,toconnect,f romdisconnect,todisconnect,execut e,viewform,show
Specification User	none	read,checkout,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Employee	none	read,create,toconnect,todisconnect, show

Message		
Revision		
Types	Message	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	all
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Access Grantor	read,modify,checkout,checkin,lock,unlock,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revok e,show	read,modify,checkout,checkin,lock,unlock,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revok e,show
Global User	none	read,modify,checkout,checkin,lock,unlock,revise,fromconnect,toconne ct,show filter: (to[Reply].from.current.access[all] == true) (to[Reply].from.current.access[read] == true && to[Reply].from.current.access[from connect] == true && to[Reply].from.current.access[toco nnect] == true) (to[Message].from.current.access[al l] == true) (to[Message].from.current.access[r ead] == true && to[Message].from.current.access[fr omconnect] == true && to[Message].from.current.access[to connect] == true && to[Message].attribute[Access Type] == Inherited)

Model					
Revision					
Types	Model				
Hidden	FALSE				

Model					
States	Preliminary	Marketing	Review	Active	Inactive
Revisionable	TRUE	TRUE	TRUE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	read,modify,checkout,changeowner,demote,changevault,changeiname,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,demote,changevault,changeiname,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,demote,changevault,changeiname,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,demote,changevault,changeiname,fromdisconnect,todisconnect,execute,modifyform,viewform,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Company Name	read,checkout,toconnect,todisconnect,execute,viewform,show	read,checkout,toconnect,todisconnect,execute,viewform,show	read,checkout,toconnect,todisconnect,execute,viewform,show	read,checkout,toconnect,todisconnect,execute,viewform,show	read,checkout,todisconnect,execute,viewform,show
System Engineer	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,create,fromdisconnect,todisconnect,execute,grant,show
Marketing Manager	read,modify,checkout,fromconnect,fromdisconnect,execute,viewform,show	read,modify,checkout,fromconnect,fromdisconnect,execute,viewform,show	read,modify,checkout,fromconnect,fromdisconnect,execute,viewform,show	read,modify,checkout,fromconnect,fromdisconnect,execute,viewform,show	read,modify,checkout,fromdisconnect,execute,viewform,show
Product Manager	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,changeowner,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,show	read,modify,checkout,todisconnect,fromdisconnect,execute,viewform,show
Global User	none	none	none	read,show	read,show

Option		
Revision	-	
Types	Option	
Hidden	FALSE	
States	Preliminary	Release
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	read,show	read,show
Owner	all	changeowner,show
Employee	read,create,show	read,show
Company Name	read,create,show	read,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Product Manager	all	read,modify,toconnect,todisconnect,show
Design Engineer	all	read,modify,toconnect,todisconnect,show

Organization		
Revision	-,0,1,2,...	
Types	Organization	
Hidden	FALSE	
States	Inactive	Active
Revisionable	TRUE	TRUE
Versionable	FALSE	FALSE
Auto Promote	TRUE	FALSE
Checkout History	TRUE	TRUE
Public	read,show	read,fromconnect,fromdisconnect,show
Owner	changeowner,show	changeowner,show

Organization		
Company Name	read,create,revise,show	read,create,revise,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Buyer	all	all
Supplier	read,modify,show	read,modify,toconnect,todisconnect,show
Customer	read,modify,show	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Supplier Representative	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Customer Representative	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Organization Manager	all	all
Sales Engineer	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Supplier Development Manager	all	all
Buyer Administrator	all	all
Company Representative	read,modify,fromconnect,fromdisconnect	read,modify,fromconnect,fromdisconnect
Employee	read,create,revise,show	read,create,revise,show
System Engineer	fromconnect	fromconnect
Product Manager	none	fromconnect,fromdisconnect
Design Engineer	none	fromconnect,fromdisconnect
Advanced Quality Engineer	none	read, fromconnect, toconnect, fromdisconnect,todisconnect,show
Senior Design Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify
Manufacturing Engineer	read, show, create, fromconnect, toconnect, fromdisconnect, todisconnect, modify	read, show, create, from connect, to connect, from disconnect, to disconnect, modify
Senior Manufacturing Engineer	none	fromconnect,fromdisconnect
Component Engineer	none	fromconnect,fromdisconnect
Specification Office Manager	none	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show

Organization List	
Revision	
Types	Organization List
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Buyer	read,modify,delete,fromconnect,fromdisconnect,show filter: to[Buyer Desk Bid List].businessobject.to[Assigned Buyer].businessobject.name == context.user.name
Buyer Administrator	read,modify,delete,fromconnect,fromdisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Customer	read,modify,delete,fromconnect,fromdisconnect,show filter: to[Buyer Desk Bid List].businessobject.to[Assigned Buyer].businessobject.name == context.user.name
Grants are used to expose the Organization List to other users depending on its usage.	

Package		
Revision		
Types	Package	
Hidden	FALSE	
States	Started	Complete
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	modify,delete,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,grant,show	show

Package		
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Buyer	read,create,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	read,show	read,show
Customer	read,create,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	read,show	read,show
Route Delegation Grantor	read,modify,delete,checkout,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoked,show	read,show
Workspace Access Grantor	read,modify,delete,checkout,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoked,show	read,show
Workspace Lead Grantor	read,modify,delete,checkout,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoked,show	read,show
Workspace Member Grantor	read,modify,delete,checkout,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoked,show	read,show

Part Family		
Revision	-	
Types	Part Family	
Hidden	FALSE	
States	Exists	Obsolete
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none

Part Family		
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,changeame,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,changeame,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Company Name	read,show	read,show
Part Family Coordinator	all	read,demote,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Employee	read,show	read,show

Part Master	
Revision	
Types	Part Master
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Part Quality Plan			
Revision	-		
Types	Part Quality Plan		
Hidden	FALSE		
States	Draft	Active	Complete

Part Quality Plan			
Revisionable	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	toconnect, todisconnect	toconnect, todisconnect	none
Public Modify	modify, promote, demote, changename, changetype, fromconnect, toconnect, fromdisconnect, todisconnect filter: program[emxSupplierPartQualityPlan ADVANCED_QUALITY_ENGINEER -method hasAccess -construct \${OBJECTID}] == true	modify, promote, changename, changetype, fromconnect, toconnect, fromdisconnect, todisconnect filter: program[emxSupplierPartQualityPlan ADVANCED_QUALITY_ENGINEER -method hasAccess -construct \${OBJECTID}] == true	none
Public Read	read, checkout, execute, show filter: program[emxSupplierPartQualityPlan PROJECT_MEMBER -method hasAccess -construct \${OBJECTID}] == true	read, checkout, execute, show filter: program[emxSupplierPartQualityPlan PROJECT_MEMBER -method hasAccess -construct \${OBJECTID}] == true	read, checkout, execute, show filter: program[emxSupplierPartQualityPlan PROJECT_MEMBER -method hasAccess -construct \${OBJECTID}] == true
Owner	read, create, modify, delete, execute, promote, demote, changename, changetype, changeowner, changepolicy, fromconnect, toconnect, fromdisconnect, todisconnect, show	read, modify, promote, changename, changetype, changeowner, changepolicy, fromconnect, toconnect, fromdisconnect, todisconnect, show	read, show
Advanced Quality Engineer	none	read, show	read, show
Shadow Agent	all	all	all
System Conversion Manager	all	all	all
System Transition Manager	all	all	all

Part Quality Plan Template		
Revision	-	
Types	Part Quality Plan Template	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	FALSE	FALSE
Public	read, fromconnect, toconnect, fromdisconnect, todisconnect, show	read, fromconnect, toconnect, fromdisconnect, todisconnect, show

Part Quality Plan Template		
Owner	read, modify, delete, promote, demote, changename, changetype, changeowner, changepolicy, changevault, show	read, modify, delete, promote, demote, changename, changetype, changeowner, changepolicy, changevault, show
Quality Administrator	create, changeowner	create, changeowner
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Person		
Revision	-	
Types	Person	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Owner	all	all
Buyer	all	all
Supplier Representative	read,modify,delete,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,delete,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
Customer Representative	read,modify,delete,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,delete,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
Company Name	read,create,revise,show	read,create,revise,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Product Manager	read,modify,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show
Design Engineer	read,modify,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show
Organization Manager	all	all

Person		
Sales Engineer	read,modify,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,show
Supplier Development Manager	all	all
Buyer Administrator	all	all
Company Representative	all	all
Employee	read,create,revise,show	read,create,revise,show
Specification Office Manager	read,modify,promote,create,tocconnect,todisconnect filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,tocconnect,todisconnect,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name

Personal Folder	
Revision	
Types	Personal Folder
Hidden	FALSE
States	Active
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	FALSE
Public	none
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
<i>Deprecated in version 10.0.0.0.</i>	

Plan	
Revision	-
Types	Plan
Hidden	FALSE

Plan	
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Quality Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Company Name	read,checkout,show
Employee	read,checkout,show

Plant Scorecard	
Revision	-
Types	Plant Scorecard
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	read,modify,checkout,checkin,changeowner,changevault,tocconnect,todisconnect,show
Quality Manager	all
Plant Representative	read,modify,delete,checkout,checkin,create,fromconnect,tocconnect,fromdisconnect,todisconnect,show
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Company Name	read,checkout,show

Plant Scorecard	
Supplier Representative	read,checkout,show
Employee	read,checkout,show

Process	
Revision	-
Types	Process
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Buyer	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Supplier Representative	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Supplier Development Manager	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Buyer Administrator	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show

Product						
Revision	A,B,C,...					
Types	Product					
Hidden	FALSE					
States	Preliminary	Product Management	DesignEngineering	Review	Release	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

Product						
Software Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	none	read,modify,changeowner,promote,demote,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,show,revise filter:program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	none	none	none
Software Engineer Manager	read,modify,changeowner,promote,demote,create,revise,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	none	read,modify,changeowner,promote,demote,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,show,revise filter:program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineerManager] == '1'	none	none	none
Design Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	none	read,modify,changeowner,promote,demote,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,show,revise filter:program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	none	none	none

Product						
Senior Design Engineer	read,modify,changeowner,promote,demote,create,revise,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesign Engineer] == '1'	none	read,modify,changeowner,promote,demote,changevault,change name,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,show,revise filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesign Engineer] == '1'	none	none	none
Marketing Manager	read,modify,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,modify,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,modify,fromconnect,tocconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	none	none	none
Shadow Agent	all	all	all	all	all	all
Customer	none	none	read,show	none	none	none
Employee	read, revise, toconnect, todisconnect, fromdisconnect, show	read,tocconnect,todisconnect,execute,show	read,show	read,,tocconnect,todisconnect,show	read,execute,show	none
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Sales Engineer	none	none	none	none	none	read,show
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show

Product Configuration					
Revision	-				
Types	Product Configuration				
Hidden	FALSE				

Product Configuration					
States	Preliminary	Validate Configuration	Generate Precise BOM	Active	Inactive
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show
Owner	all	changeowner,show	changeowner,show	changeowner,show	changeowner,show
Employee	read,show,fromconnect,fromdisconnect	read,show,fromconnect,fromdisconnect	read,show,fromconnect,fromdisconnect	read,show,fromconnect,fromdisconnect	read,show,fromconnect,fromdisconnect
System Engineer	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show
Company Name	read,show	read,show	read,show	read,show	read,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Product Manager	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show

Product Configuration					
Design Engineer	all	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,to connect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,to connect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	none	none
Order Entry Clerk	all	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,to connect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,change type,fromconnect,to connect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	none	none
Customer	read,modify,promote,to connect,todisconnect,show	read,modify,promote,to connect,todisconnect,show	read,modify,toconnect,todisconnect,show	read,modify,toconnect,todisconnect,show	none
Access Grantor	none	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,to connect,fromdisconnect,todisconnect,execute,grant,revoke,show	none	none	none

Product Documentation					
Revision					
Types	Product Documentation				
Hidden	FALSE				
States	Plan	Document	Review	Approve	Supercede
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	all	all	all

Product Documentation					
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Product Feature						
Revision	1,2,3,...					
Types	Features					
Hidden	FALSE					
States	Preliminary	Product Management	Design Engineering	Review	Release	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none
Owner	all	read,modify,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,modify,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,modify,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show	read,modify,lock,unlock,changeowner,demote,schedule,override,enable,disable,create,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revok e,show	read,modify,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,grant,revoke,show

This policy has been renamed and its symbolic name reflects its original name. See [Renamed Administrative Objects](#) in Chapter 4.

Product Feature						
Product Manager	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show,execute,changeolicy filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'	read,modify,demote,fromconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_ProductManager] == '1'
System Engineer	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show,execute,changeolicy filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'	read,modify,demote,fromconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SystemEngineer] == '1'
Software Engineer	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show,execute filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'	read,modify,demote,fromconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineer] == '1'
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.						

Product Feature						
Software Engineering Manager	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show,execute filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'	read,modify,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SoftwareEngineeringManager] == '1'
Design Engineer	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show,execute filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'	read,modify,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_DesignEngineer] == '1'
Senior Design Engineer	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show,execute filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesignEngineer] == '1'	read,modify,changeowner,promote,demote,create,changevault,changeiname,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesignEngineer] == '1'	read,modify,demote,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_SeniorDesignEngineer] == '1'
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.						

Product Feature						
Marketing Manager	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show,fromconnect filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,modify,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'	read,show filter: program[emxOrganization -method hasRDOrole -construct \${OBJECTID} role_MarketingManager] == '1'
Global User	read,toconnect,todisconnect,execute,show,changeowner,create,demote,modify,prorate,FromConnect, fromdisconnect filter: context.user ~~ to[Assigned Feature].businessobject.name context.user ~~ to[Feature List To].businessobject.to[Feature List From].businessobject.to[Assigned Feature].businessobject.name	read,show,modify,changeowner,changeowner,fromconnect,fromdisconnect,todisconnect filter: context.user ~~ to[Assigned Feature].businessobject.name context.user ~~ to[Feature List To].businessobject.to[Feature List From].businessobject.to[Assigned Feature].businessobject.name	read,show,modify filter: context.user ~~ to[Assigned Feature].businessobject.name context.user ~~ to[Feature List To].businessobject.to[Feature List From].businessobject.to[Assigned Feature].businessobject.name	read,show filter: context.user ~~ to[Assigned Feature].businessobject.name context.user ~~ to[Feature List To].businessobject.to[Feature List From].businessobject.to[Assigned Feature].businessobject.name	read,show filter: context.user ~~ to[Assigned Feature].businessobject.name context.user ~~ to[Feature List To].businessobject.to[Feature List From].businessobject.to[Assigned Feature].businessobject.name	none
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Customer	none	none	none	none	read,fromconnect,fromdisconnect,show	none
Employee	read,create,show,todisconnect,fromdisconnect	read,create,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,show
Company Name	read,create,show	read,create,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show	read,show
Sales Engineer	none	none	none	none	read,fromconnect,fromdisconnect,show	none
Access Grantor	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show	read,grant,revoke,show
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.						

Product Line						
Revision	-					
Types	Product Line					
Hidden	FALSE					
States	Planned	Marketing	Define Change Board	Review	Complete	Inactive
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	read,show	read,show	none
Owner	all	read,modify,checkout,changeowner,promote,demote,changevault,change name,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,promote,demote,changevault,change name,toconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,promote,demote,changevault,change name,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,changeowner,promote,demote,changevault,change name,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Employee	read,checkout,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	fromconnect,fromdisconnect	read,checkout,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,todisconnect,execute,viewform,show
Global User	read,show	read,show	read,show	none	read,show	read,show
Product Manager	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,changeowner,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,changeowner,demote,todisconnect,execute,viewform,show
System Engineer	read,modify,checkout,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show	fromconnect,fromdisconnect	read,checkout,changeowner,demote,todisconnect,execute,viewform,show

Product Line						
Marketing Manager	read,modify,checkout,promote,demote,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	none	none	none	none
Senior Design Engineer	read,modify,checkout,promote,demote,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,tocconnect,todisconnect,execute,viewform,show	none	read,checkout,tocconnect,todisconnect,execute,viewform,show	none	none
Design Engineer	read,modify,checkout,promote,demote,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,tocconnect,todisconnect,execute,viewform,show	none	read,checkout,tocconnect,todisconnect,execute,viewform,show	none	none
Company Name	read,checkout,tocconnect,todisconnect,execute,viewform,show	none	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,todisconnect,execute,viewform,show
ECR Coordinator	read,modify,checkout,promote,demote,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,demote,tocconnect,todisconnect,execute,viewform,show	read,modify,checkout,changeowner,promote,demote,tocconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,promote,demote,tocconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,tocconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,todisconnect,execute,viewform,show
ECR Chairman	read,modify,checkout,promote,demote,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	none	modify,promote,demote	none	none	none
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all

Program		
Revision		
Types	Program	
Hidden	FALSE	

Program		
States	Active	Inactive
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	fromconnect,toconnect,fromdisconnect,todisconnect	fromdisconnect,todisconnect
Owner	read,modify,delete,changeowner,promote,demote,changevault,changeame,change policy,show	read,modify,delete,changeowner,promote,demote,changevault,changeame,change policy,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Program Lead	changeowner,create	changeowner,create
Public Read	read,show filter: program[emxProgram \$\{OBJECTID\} -method hasAccess] == true	read,show filter: program[emxProgram \$\{OBJECTID\} -method hasAccess] == true
External Program Lead	create	none

Project Access List	
Revision	
Types	Project Access List
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,delete,fromconnect,fromdisconnect,show
Owner	read,modify,changeowner,create,fromconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Project Access List	
Public Read	read,execute,show filter: program[emxCommonProjectAccessList PROJECT_READ_ACCESS -method hasAccess -construct \${OBJECTID}] == true
Access Grantor	read,modify,promote,demote,override,changename,fromconnect,execute,modifyform,viewform,grant,revoke,show

Project Concept				
Revision				
Types	Project Concept			
Hidden	FALSE			
States	Concept	Prototype	Review	Approved
Revisionable	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	FALSE	FALSE	FALSE	FALSE
Public	none	none	none	none
Owner	read,modify,delete,changeowner,promote,demote,changename,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,modify,changeowner,promote,demote,changename,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,modify,changeowner,promote,demote,changename,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,modify,changeowner,promote,demote,changename,changeopolicy,changetype,execute,show
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all
Public Read	read,checkout,execute,show filter: program[emxProjectConcept PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectConcept PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectConcept PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectConcept PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true
Public Modify	modify,promote,demote,changename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect filter: program[emxProjectConcept PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,demote,changename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect filter: program[emxProjectConcept PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,demote,changename,changetype,fromconnect,toconnect,fromdisconnect,todisconnect filter: program[emxProjectConcept PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,demote,changename,changeopolicy,changetype filter: program[emxProjectConcept PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true

Project Concept				
Project User	create	none	none	none
External Project User	create	none	none	none

Project Folder	
Revision	
Types	Project Folder
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	all
Owner	all
Shadow Agent	all
System Transition Manager	all
System Conversion Manager	all
<i>Deprecated in version 9.5.0.0.</i>	

Project Member		
Revision	-	
Types	Project Member	
Hidden	FALSE	
States	Active	Inactive
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	all	all
Owner	all	all
System Transition Manager	all	all

Project Member		
System Conversion Manager	all	all
Shadow Agent	all	all

Project Release			
Revision			
Types	Workspace		
Hidden	FALSE		
States	Plan	Active	Complete
Revisionable	TRUE	TRUE	TRUE
Versionable	FALSE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	read,show	none	none
Owner	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,fromconnect,fromdisconnect,viewform,show	read,fromconnect,fromdisconnect,viewform,show
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,fromconnect,fromdisconnect,viewform,show	read,fromconnect,fromdisconnect,viewform,show
Software Project Manager	none	read,modify,delete,checkout,checkin,changeowner,promote,create,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,modify,delete,checkout,viewform,show

Project Risk					
Revision					
Types	Risk				
Hidden	FALSE				
States	Create	Assign	Active	Review	Complete
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE

Project Risk					
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	read,modify,delete,changeowner,promote,demote,create,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,create,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Public Read	read,checkout,show filter: attribute[Risk Visibility] == Public && program[emxRisk PROJECT_MEMBER RISK_ASSIGNEE -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,show filter: attribute[Risk Visibility] == Public && program[emxRisk PROJECT_MEMBER RISK_ASSIGNEE -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,show filter: attribute[Risk Visibility] == Public && program[emxRisk PROJECT_MEMBER RISK_ASSIGNEE -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,show filter: attribute[Risk Visibility] == Public && program[emxRisk PROJECT_MEMBER RISK_ASSIGNEE -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,show filter: attribute[Risk Visibility] == Public && program[emxRisk PROJECT_MEMBER RISK_ASSIGNEE -method hasAccess -construct \${OBJECTID\}] == true
Public Modify	read,modify,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxRisk PROJECT_LEAD RISK_ASSIGNEE PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	read,modify,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxRisk PROJECT_LEAD RISK_ASSIGNEE PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	read,modify,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxRisk PROJECT_LEAD RISK_ASSIGNEE PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	read,modify,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxRisk PROJECT_LEAD RISK_ASSIGNEE PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	read,modify,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: program[emxRisk PROJECT_LEAD RISK_ASSIGNEE PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true
Public Delete	delete,changeowner filter: program[emxRisk PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	changeowner filter: program[emxRisk PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	changeowner filter: program[emxRisk PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	changeowner filter: program[emxRisk PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	changeowner filter: program[emxRisk PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true

Project Space						
Revision						
Types	Project Space					

Project Space						
Hidden	FALSE					
States	Create	Assign	Active	Review	Complete	Archive
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	toconnect,todisconnect	toconnect,todisconnect	toconnect,todisconnect	none	none	none
Owner	read,modify,delete,changeowner,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,modify,changeowner,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,changeowner,promote,demote,changeiname,changeopolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,changeowner,demote,changeopolicy,show
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Project Lead	create	none	none	none	none	none
Public Read	read,checkout,execute,show filter: program[emxProjectSpace PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectSpace PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectSpace PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectSpace PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectSpace PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,execute,show filter: program[emxProjectSpace PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true

Project Space						
Public Modify	modify,promote,demote,changenam e,changetype,from connect,toconnect, fromdisconnect,to disconnect filter: program[emxProje ctSpace PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,d emote,changenam e,changetype,from connect,toconnect, fromdisconnect,to disconnect filter: program[emxProje ctSpace PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,d emote,changenam e,changetype,from connect,toconnect, fromdisconnect,to disconnect filter: program[emxProje ctSpace PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,d emote,changenam e,changetype,from connect,toconnect, fromdisconnect,to disconnect filter: program[emxProje ctSpace PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	modify,promote,d emote,changenam e,changetype,from connect,toconnect, fromdisconnect,to disconnect filter: program[emxProje ctSpace PROJECT_LEAD -method hasAccess -construct \${OBJECTID\}] == true	none
External Project Lead	create	none	none	none	none	none

Project Task					
Revision					
Types	Task Management				
Hidden	FALSE				
States	Create	Assign	Active	Review	Complete
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	TRUE	TRUE	TRUE	TRUE	TRUE
Checkout History	FALSE	FALSE	FALSE	FALSE	FALSE
Public	create,toconnect,todisc onnect	toconnect,todisconnect	toconnect,todisconnect	none	none
Owner	read,modify,delete,cha ngeowner,promote,de mote,changename,cha ngepolicy,changetype, fromconnect,toconnect ,fromdisconnect,todisc onnect,show	read,modify,changeow ner,promote,demote,c hangename,change poli cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,show	read,modify,changeow ner,promote,demote,cr eate,changename,chan gepolicy,changetype,fr omconnect,toconnect,f romdisconnect,todisco nnect,show	read,modify,changeow ner,promote,demote,cr eate,changename,chan gepolicy,changetype,fr omconnect,toconnect,f romdisconnect,todisco nnect,show	read,modify,changeow ner,promote,demote,cr eate,changename,chan gepolicy,changetype,fr omconnect,toconnect,f romdisconnect,todisco nnect,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all

Project Template		
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Project Administrator	changeowner,create	changeowner,create
Public Read	read,show filter: program[emxProjectTemplate -method hasAccess -construct \$\{OBJECTID\}] == true	read,show filter: program[emxProjectTemplate -method hasAccess -construct \$\{OBJECTID\}] == true
External Project Administrator	create	none

Public Folder	
Revision	
Types	Public Folder
Hidden	FALSE
States	Active
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	FALSE
Public	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
<i>Deprecated in version 10.0.0.0.</i>	

Publish Subscribe		
Revision		
Types	Publish Subscribe	
Hidden	FALSE	
States	Active	Inactive
Revisionable	FALSE	FALSE

Publish Subscribe		
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	all	all
Owner	all	all
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Purchase Class	
Revision	-
Types	Purchase Class
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	read,modify,checkout,checkin,changeowner,changevault,tocnnect,todisconnect,show
Buyer Administrator	all
Supplier Development Manager	all
Quality Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Company Name	read,checkout,toconnect,todisconnect,show
Employee	read,checkout,toconnect,todisconnect,show

Purchase Class Scorecard	
Revision	-
Types	Purchase Class Scorecard
Hidden	FALSE

Purchase Class Scorecard	
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	read,modify,checkout,checkin,changeowner,changevault,toc nnect,todisconnect,show
Quality Manager	all
Supplier Development Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Company Name	read,checkout,show
Employee	read,checkout,show

Quality		
Revision		
Types	Quality	
Hidden	FALSE	
States	Initiated	Controlled
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	read,modify,create,fromconnect,toc onnect,show	show
Shadow Agent	all	all
System Conversion Manager	all	all
System Transition Manager	all	all

Quality		
Public Modify	modify,delete,changeowner,promote,demote,changevault,changeiname,changeobject,changeobjecttype,fromconnect,toconnect,fromdisconnect,todisconnect filter: program[emxQuality PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true	promote,demote filter: program[emxQuality PROJECT_LEAD PROJECT_OWNER -method hasAccess -construct \${OBJECTID\}] == true
Public Read	read,checkout,show filter: program[emxQuality PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true	read,checkout,show filter: program[emxQuality PROJECT_MEMBER -method hasAccess -construct \${OBJECTID\}] == true

Quality Metric	
Revision	
Types	Quality Metric
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,modify,delete,create,toconnect,todisconnect,show
Owner	show
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Question		
Revision		
Types	Question	
Hidden	FALSE	
States	Active	Inactive
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE

Question		
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	read,fromconnect,fromdisconnect,show	read,fromconnect,fromdisconnect,show
Owner	modify,create,toconnect,show	modify,create,toconnect,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Public Modify	modify,delete,promote,demote,change ngename,change policy,toconnect,to disconnect filter: program[emxQuestion PARENT_MODIFY -method hasAccess -construct \${OBJECTID\}] == true	modify,delete,promote,demote,cha ngename,change policy,toconnect,to disconnect filter: program[emxQuestion PARENT_MODIFY -method hasAccess -construct \${OBJECTID\}] == true

Rate Period	
Revision	
Types	Rate Period
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeown er,promote,demote,schedule,override,enable,disable,revise,cha ngevault,change name,change policy,changetype,fromconnec t,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute, modifyform,viewform,grant,revoke,show
Buyer	show
Supplier	show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer Administrator	read,modify,create,fromconnect,toconnect,fromdisconnect,to disconnect

Region	
Revision	-
Types	Region
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Employee	read,toconnect,todisconnect,show
Specification Manager	read,fromconnect,toconnect,fromdisconnect,show
Specification Office Manager	read,fromconnect,toconnect,fromdisconnect,show
Global Template Administrator	read,fromconnect,toconnect,fromdisconnect,show
Template Administrator	read,fromconnect,toconnect,fromdisconnect,show
Organization Manager	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Request For Quote					
Revision	-				
Types	Request For Quote				
Hidden	FALSE				
States	Not Submitted	Submitted	In Process	Quoted	Closed
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show
Owner	read,show	read,show	read,show	read,show	all

Request For Quote					
Customer	all	read,modify,checkout,checkin,lock,unlock,changeowner,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show
Customer Representative	all	read,modify,checkout,checkin,lock,unlock,changeowner,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show
Sales Engineer	read,fromconnect,tconnect,fromdisconnect,todisconnect,show	read,promote,fromconnect,tconnect,fromdisconnect,todisconnect,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,demote,schedule,override,enable,disable,create,revise,changevault,changeiname,changeopolicy,changetype,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,fromconnect,tconnect,fromdisconnect,todisconnect,show
Shadow Agent	all	all	all	all	all
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all

Requirement				
Revision	1,2,3,...			
Types	User Requirement	Requirement		
Hidden	FALSE			
States	Propose	Approve	Validate	Release
Revisionable	FALSE	FALSE	FALSE	TRUE

Requirement				
Versionable	FALSE	FALSE	FALSE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none
Owner	all	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show	read,checkout,changeowner,demote,revise,toconnect,todisconnect,execute,viewform,show
Shadow Agent	all	all	all	none
System Conversion Manager	all	all	all	none
System Transition Manager	all	all	all	none
Company Name	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	none
Employee	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show filter: context.user ~~to[Assigned Requirement].businessobject.name context.user.assignment ~~to[Assigned Requirement].businessobject.name context.user.assignment ~~to[Assigned Requirement].businessobject.revision	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show filter: context.user ~~to[Assigned Requirement].businessobject.name context.user.assignment ~~to[Assigned Requirement].businessobject.name context.user.assignment ~~to[Assigned Requirement].businessobject.revision	read,checkout,promote,demote,toconnect,todisconnect,execute,viewform,show filter: context.user ~~to[Assigned Requirement].businessobject.name context.user.assignment ~~to[Assigned Requirement].businessobject.name context.user.assignment ~~to[Assigned Requirement].businessobject.revision	none
ECR Coordinator	read,checkout,changeowner,demote,toconnect,todisconnect,execute,viewform,show	none	none	none
Global User	read,show	read,show	read,show	none

Requirement				
Requirement Manager	all filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision	read,checkout,promote,demote,toconnect,todisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision	read,checkout,promote,demote,toconnect,todisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision	read,checkout,changeowner,demote,create,revise,toconnect,todisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision
Product Manager	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,promote,demote,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision	read,checkout,changeowner,demote,create,revise,toconnect,todisconnect,fromconnect,fromdisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision	read,checkout,changeowner,demote,revise,toconnect,todisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.name context.user.assignment ~~ to[Assigned Requirement].businessobjec t.revision
System Engineer	fromconnect, fromdisconnect	fromconnect, fromdisconnect	fromconnect, fromdisconnect	none

Response			
Revision	-		
Types	Manufacturing Assessment	Quotation	
Hidden	FALSE		
States	Open	Review	Closed
Revisionable	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show

Response			
Owner	all	read,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,revise,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,viewform,grant,show	read,checkout,changeowner,demote,schedule,override,enable,disable,revise,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,viewform,grant,show
Buyer	all	read,checkout,checkin,lock,unlock,promote,demote,show	read,checkout,demote,show
Supplier	read,modify,checkout,checkin,show	read,checkout,show	read,checkout,show
Supplier Representative	read,modify,checkout,checkin,promote,show	read,checkout,show	read,checkout,show
Design Engineer	all	all	all
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Customer	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeepolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeepolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show
Customer Representative	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeepolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeepolicy,changetype,fromconnect,tocconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,show
Sales Engineer	all	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Supplier Engineer	read,modify,checkout,checkin,show	read,checkout,show	read,checkout,show
Buyer Administrator	all	read,checkout,checkin,lock,unlock,promote,demote,show	read,checkout,show

Restricted Legacy Technical Specification		
Revision	-	
Types	Technical Specification	
Hidden	FALSE	
States	Issued	Obsolete
Revisionable	TRUE	TRUE
<i>This policy contains an admin property called PolicyClassification with the value Restricted- Legacy.</i>		

Restricted Legacy Technical Specification		
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	read,checkout,checkin,lock,unlock,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,checkout,checkin,lock,unlock,fromconnect,tocconnect,fromdisconnect,todisconnect,show
Specification Manager	read,checkout,checkin,toconnect,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,checkout,checkin,lock,unlock,tocconnect,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)
Specification Office Manager	delete,checkout,checkin,changeowner,promote,create,changevault,changepolicy filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	delete,checkout,checkin,changeowner,promote,create,changevault,changepolicy filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Specification Restricted User	read,checkout,show	none
Global User	read,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: (context.user ~~ to[CoOwned].businessobject.to[CoOwns].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' (to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user))	read,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: (context.user ~~ to[CoOwned].businessobject.to[CoOwns].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' (to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user))
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Supplier	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobject.from[Employee].businessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobject.from[Employee].businessobject.name
Product Manager	fromconnect,toconnect,fromdisconnect,todisconnect,	
System Engineer	fromconnect,fromdisconnect	
This policy contains an admin property called PolicyClassification with the value Restricted- Legacy.		

Restricted Technical Specification					
Revision	A,B,C,...				
Types	Technical Specification				
Hidden	FALSE				
States	Draft	Peer Review	Formal Approval	Issued	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeiname,changeipolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,checkout,checkin,lock,unlock,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,checkout,create,revise,fromconnect,toconnect,todisconnect,execute,show	read,checkout,create,revise,toconnect,todisconnect,show	read,demote,toconnect,todisconnect,show
Specification Manager	read,checkout,checkin,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,checkout,create,tconnect,fromdisconnect,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,checkout,create,tconnect,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,create,toconnect,tconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,toconnect,fromdisconnect,todisconnect,execute,show
This policy contains an admin property called PolicyClassification with the value General.					

Restricted Technical Specification					
Specification Restricted User	read,checkout,checkin,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,checkout,create,tconnect,fromdisconnect,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,checkout,create,tconnect,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,checkout,create,tconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~~ 'Specification Restricted User' && to[Business Unit Owns].businessobject.from[Business Unit Employee].businessobject.name ~~ context.user)	read,tconnect,fromdisconnect,todisconnect,execute,show
Specification Office Manager	read,modify,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,changeowner,create,revise,changeowner,changetype,toconnect,todisconnect,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,checkout,checkin,lock,unlock,changeowner,demote,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
This policy contains an admin property called PolicyClassification with the value General.					

Restricted Technical Specification					
Global User	read,modify,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeowner,changeobject.to[CoOwne].businessobject.to[CoOwne].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,revise,fromconnect,toconnect,fromdisconnect,execute,show filter: (context.user ~ to[CoOwne].businessobject.to[CoOwne].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,checkout,create,revise,fromconnect,connect,todisconnect,execute,show filter: (context.user ~ to[CoOwne].businessobject.to[CoOwne].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,checkout,create,revise,toconnect,todisconnect,show filter: (context.user ~ to[CoOwne].businessobject.to[CoOwne].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,changevault,changeowner,changeobject.to[CoOwne].businessobject.to[CoOwne].businessobject.name) && (attribute[Security Classification] != 'Highly Restricted' to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)
Global Template Administrator	read,checkout,lock,unlock,create,toconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,create,toconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,create,toconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,checkout,create,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,checkout,create,todisconnect,execute,show
Template Administrator	read,checkout,lock,unlock,create,toconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,create,toconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,create,toconnect,todisconnect,execute,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,checkout,create,todisconnect,show filter: attribute[Security Classification] != 'Highly Restricted' (context.user.assignment ~ 'Specification Restricted User' && to[Business Unit Ows].businessobject.from[Business Unit Employee].businessobject.name ~ context.user)	read,checkout,create,todisconnect,show
This policy contains an admin property called PolicyClassification with the value General.					

Restricted Technical Specification					
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Supplier	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name
Product Manager	fromconnect,toconnect ,fromdisconnect,todisc onnect	fromconnect,toconnect ,fromdisconnect,todisc onnect	fromconnect,toconnect ,fromdisconnect,todisc onnect	fromconnect,toconnect ,fromdisconnect,todisc onnect	none
System Engineer	fromconnect,fromdisc onnect	fromconnect,fromdisc onnect	fromconnect,fromdisc onnect	fromconnect,fromdisc onnect	none
<i>This policy contains an admin property called PolicyClassification with the value General.</i>					

RFQ						
Revision						
Types	RFQ					
Hidden	FALSE					
States	Started	Initial Review	Sent	Response Complete	Final Review	Complete
Revisionable	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none
Owner	modify,delete,chan geowner,promote, changepolicy,from connect,fromdisco nnect,grant,voke ,show	changeowner,pro mote,demote,chan gepolicy,fromconn ect,fromdisconnect ,show	modify,changeow ner,promote,revise ,changepolicy,fro mconnect,fromdis connect,grant,repo ke,show	changeowner,pro mote,revise,chan gepolicy,show	changeowner,pro mote,demote,revise, changepolicy,show	show
System Transition Manager	all	all	all	all	all	all
Some states in this policy have been renamed and their symbolic name reflect the original name. See Renamed Administrative Objects in Chapter 4.						

RFQ						
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Buyer	read,create,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	read,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]
Buyer Administrator	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Customer	read,create,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Some states in this policy have been renamed and their symbolic name reflect the original name. See Renamed Administrative Objects in Chapter 4.						

RFQ						
Sales Engineer	read,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]	read,checkout,show filter: attribute[Secondary Keys] == " context.user.property[Company Key].value matchlist attribute[Secondary Keys] , '~' context.user.property[Company Key].value == attribute[Primary Key]
Route Delegation Grantor	read,modify,delete,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,modify,checkout,changeowner,promote,revise,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,changeowner, promote,revise,change policy,grant, revoke,show	read,changeowner, promote,demote,revise,change policy, grant, revoke,show	read,show
Workspace Access Grantor	read,modify,delete,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,modify,checkout,changeowner,promote,revise,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,changeowner, promote,revise,change policy,grant, revoke,show	read,changeowner, promote,demote,revise,change policy, grant, revoke,show	read,show
Workspace Lead Grantor	read,modify,delete,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,modify,checkout,changeowner,promote,revise,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,changeowner, promote,revise,change policy,grant, revoke,show	read,changeowner, promote,demote,revise,change policy, grant, revoke,show	read,show
Workspace Member Grantor	read,modify,delete,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,checkout,changeowner,promote,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,modify,checkout,changeowner,promote,revise,change policy,fromconnect,toconnect,fromdisconnect,todisconnect,grant, revoke,show	read,changeowner, promote,revise,change policy,grant, revoke,show	read,changeowner, promote,demote,revise,change policy, grant, revoke,show	read,show
Some states in this policy have been renamed and their symbolic name reflect the original name. See Renamed Administrative Objects in Chapter 4.						

RFQ Cancelled	
Revision	
Types	RFQ
Hidden	FALSE
States	Cancelled
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	read,modify,delete,grant,revoke,show
Buyer	read,modify,revise,changepolicy,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	read
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Customer	read,modify,revise,changepolicy,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	read
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

RFQ History Version	
Revision	
Types	RFQ
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

RFQ History Version	
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	delete,show
Buyer	read filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	read
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer Administrator	read filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Customer	read filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	read
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

RFQ Pending Version	
Revision	
Types	RFQ
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	modify,delete,promote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

RFQ Pending Version	
Buyer	read filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	none
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Customer	read filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	none
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

RFQ Quotation				
Revision				
Types	RFQ Quotation			
Hidden	FALSE			
States	Open	Review	Returned	Closed
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none
Owner	modify,changeowner,promote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,show	changeowner,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,show	changeowner,grant,show	show
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.				

RFQ Quotation				
Supplier	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]
Buyer	show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'read,show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'read,show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~
Buyer Administrator	none filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'none filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'none filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'none filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~
Customer	show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'read,show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~	'read,show filter: attribute[Secondary Keys] == " context.user.property[Comp any Key].value matchlist attribute[Secondary Keys] ,~
Sales Engineer	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]	read,show filter: attribute[Primary Key] == " context.user.property[Comp any Key].value == attribute[Primary Key]
Route Delegation Grantor	read,modify,checkout,changeowner,promote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,grant,revoke,show	read,show
Workspace Access Grantor	read,modify,checkout,changeowner,promote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,grant,revoke,show	read,show
Workspace Lead Grantor	read,modify,checkout,changeowner,promote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,grant,revoke,show	read,show
Workspace Member Grantor	read,modify,checkout,changeowner,promote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,checkout,promote,demote,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,grant,revoke,show	read,show
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.				

RFQ Template		
Revision	1,2,3,...	
Types	RFQ Template	
Hidden	FALSE	
States	Inactive	Active
Revisionable	TRUE	TRUE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	changename,show	show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Buyer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Buyer Administrator	read,modify,delete,promote,create,fromconnect,t oconnect,fromdisconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,demote,revise,fromconnect,toconnect,fromdi sconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Supplier	read,show	read,show
Access Grantor	read,grant,revoke,show	read,grant,revoke,show
Customer	none filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]	read,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Sales Engineer	read,show	read,show
Exchange User	none	read,show
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.		

Risk				
Revision				
Types	Risk			
Hidden	FALSE			

Risk				
States	Identified	Assessment	Mitigate	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none
Owner	all	all	all	all
Company Name	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Shadow Agent	all	all	all	all
Employee	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show	read,checkout,from connect,toconnect,from disconnect,todisconnect,execute,viewform,show

Route				
Revision				
Types	Route			
Hidden	FALSE			
States	Define	In Process	Complete	Archive
Revisionable	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	FALSE
Public	none	none	none	none
Owner	all	all	all	read,show
Shadow Agent	all	all	all	all

Route				
System Transition Manager	all	all	all	all
System Conversion Manager	all	all	all	all
Workspace Lead Grantor	promote,demote	promote,demote	none	none
Access Grantor	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,checkout,grant,revoke,show	read,checkout,grant,revoke,show

Route Task User	
Revision	
Types	Route Task User
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Route Template		
Revision	1,2,3,...	
Types	Route Template	
Hidden	FALSE	
States	Inactive	Active
Revisionable	TRUE	TRUE

Route Template		
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	fromconnect
Owner	all	all
Specification Office Manager	read,delete,checkout,checkin,lock,unlock,change owner,schedule,override,enable,disable,create,revise,changevault,changename,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Organization Manager	read,delete,checkout,checkin,lock,unlock,change owner,schedule,override,enable,disable,create,revise,changevault,changename,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Company Representative	read,delete,checkout,checkin,lock,unlock,change owner,schedule,override,enable,disable,create,revise,changevault,changename,change policy,change type,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all
Exchange User	none	read,show
Company Name	none	read,show
Employee	none	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
Global User	none	all filter: (to[Route Templates].from.grantor == 'Workspace Lead Grantor') (context.user == owner && revision == last.revision)
Specification Manager	none	read,toconnect,show
Template Administrator	none	read,toconnect,show
Global Template Administrator	none	read,toconnect,show

RPN	
Revision	
Types	RPN
Hidden	FALSE

RPN	
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,modify,delete,create,toconnect,todisconnect,show
Owner	show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

RTS Quotation Archive	
Revision	1,2,3,...
Types	RTS Quotation Archive
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Rule		
Revision	-	
Types	Feature Rule Option Rule Part Rule Fixed Resource Boolean Compatibility Rule Rule Extension Feature List GBOM	
Hidden	FALSE	
States	Preliminary	Release
Revisionable	FALSE	FALSE
Versionable	FALSE	FALSE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	changeowner,show
Employee	read,create,show,toconnect,todisconnect	read,show
System Engineer	all	read,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Software Engineer	all	read,modify,show
Company Name	read,create,show	read,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Rule		
Product Manager	all	read,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Design Engineer	all	read,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show

Sales Order							
Revision	-						
Types	Sales Order						
Hidden	FALSE						
States	Not Submitted	Validate	Process	Manufacture	Shipping	Shipped	Payment Received
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show	read,show	read,show
Owner	changeowner,show	changeowner,show	changeowner,show	changeowner,show	changeowner,show	changeowner,show	changeowner,show
Company Name	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show
System Transition Manager	all	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all	all
Customer	all	read,show	read,show	read,show	read,show	read,show	read,show
Order Entry Clerk	all	read,modify,promote,show	read,modify,promote,show	none	none	none	none
Employee	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show	read,create,revise,show

Sales Order							
Production Support Clerk	none	none	none	read,modify, promote,show	none	none	none
Shipping Clerk	none	none	none	none	read,modify, promote,show	none	none
Accounting Clerk	none	none	none	none	none	read,modify, promote,show	none

SCO					
Revision	-				
Types	SCO				
Hidden	FALSE				
States	Draft	Peer Review	Formal Approval	Final Check	Approved
Revisionable	FALSE	FALSE	FALSE	FALSE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	read,promote,demote,create,fromconnect,fromdisconnect,viewform,show	read,promote,demote,create,fromconnect,fromdisconnect,viewform,show	read,promote,create,show	read,toconnect,todisconnect,viewform,show
Specification Manager	read,promote,demote,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	read,promote,demote,fromconnect,fromdisconnect,viewform,show	read,promote,demote,fromconnect,fromdisconnect,viewform,show	read,promote,viewform,show	read,viewform,show
Specification User	none	none	none	none	read,checkout,show
Specification Restricted User	read,promote,demote,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,viewform,show	read,promote,demote,fromconnect,fromdisconnect,viewform,show	read,promote,demote,fromconnect,fromdisconnect,viewform,show	read,promote,viewform,show	read,checkout,viewform,show

SCO					
Specification Office Manager	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned To Specification Office],businessobject.from[Assigned Member],businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned To Specification Office],businessobject.from[Assigned Member],businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned To Specification Office],businessobject.from[Assigned Member],businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[Assigned To Specification Office],businessobject.from[Assigned Member],businessobject.name	read,changeowner,changeowner,fromconnect,connect,fromdisconnect,todisconnect,show filter: context.user ~~~to[Assigned To Specification Office],businessobject.from[Assigned Member],businessobject.name
Global User	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changeowner,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~~to[CoOwned],businessobject.to[CoOwne],businessobject.name	read,modify,promote,demote,create,fromconnect,fromdisconnect,viewform,show filter: context.user ~~~to[CoOwned],businessobject.to[CoOwne],businessobject.name	read,promote,demote,create,fromconnect,fromdisconnect,viewform,show filter: context.user ~~~to[CoOwned],businessobject.to[CoOwne],businessobject.name	read,promote,create,show filter: context.user ~~~to[CoOwned],businessobject.to[CoOwne],businessobject.name	read,fromconnect,connect,fromdisconnect,todisconnect,viewform,show filter: context.user ~~~to[CoOwned],businessobject.to[CoOwne],businessobject.name
Global Template Administrator	read,promote,demote,create,fromconnect,connect,fromdisconnect,todisconnect,execute,viewform,show	read,promote,demote,fromconnect,fromdisconnect,execute,viewform,show	read,promote,demote,fromconnect,fromdisconnect,viewform,show	read,promote,viewform,show	read,viewform,show
Template Administrator	read,promote,demote,create,fromconnect,connect,fromdisconnect,todisconnect,execute,viewform,show	read,promote,demote,fromconnect,fromdisconnect,execute,viewform,show	read,promote,demote,fromconnect,fromdisconnect,viewform,show	read,promote,viewform,show	read,viewform,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Supplier	read,show	read,show	read,show	read,show	none
Public Read	none	read,show	read,show	none	none
Employee	read,show	read,show	read,show	read,show	read,show

Scorecard	
Revision	-
Types	Scorecard
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Quality Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Company Name	read,checkout,show
Employee	read,checkout,show

Software Architecture					
Revision	1,2,3,...				
Types	Architecture				
Hidden	FALSE				
States	Plan	Working	Draft	Review	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	all	all	all
Shadow Agent	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
System Transition Manager	all	all	all	all	all

Software Architecture					
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Build								
Revision	1,2,3,...							
Types	Software Build							
Hidden	FALSE							
States	Plan	Implement	Build	Internal Test	External Test	Complete	Media	Supercede
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show	read,show	read,show	read,show
Owner	all	all	all	all	all	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all	all	all

Software Build								
Shadow Agent	all	all	all	all	all	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Design Specification					
Revision	1,2,3,...				
Types	Design Specification				
Hidden	FALSE				
States	Plan	Working	Draft	Review	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	all	all	all
Shadow Agent	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
System Transition Manager	all	all	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Distribution						
Revision	1,2,3,...					
Types	Distribution File					
Hidden	FALSE					

Software Distribution						
States	Plan	Distribute	Test	Available	Reject	Archive
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read,show	read,show	read,show	read,show	read,show	read,show
Owner	all	all	all	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Software Quality Engineering	none	none	read,checkout,show	read,checkout,show	none	none
Software Engineering	none	none	read,checkout,show	read,checkout,show	none	none

Software Feature								
Revision	1,2,3,...							
Types	Software Feature							
Hidden	FALSE							
States	Propose	Assign	Specify	Working	Review	Submit	Validate	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Software Feature								
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none	none	none
Owner	all	all	all	all	all	all	all	all
Company Name	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all	all	all
Employee	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdiscconnect,todiscconnect,execute,viewform,show

Software Functional Specification					
Revision	1,2,3,...				
Types	Functional Specification				
Hidden	FALSE				
States	Plan	Working	Draft	Review	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	all	all	all

Software Functional Specification					
Shadow Agent	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
System Transition Manager	all	all	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Platform	
Revision	1,2,3,...
Types	Software Platform
Hidden	FALSE
States	Base
Revisionable	TRUE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Release							
Revision	1,2,3,...						
Types	Software Release						
Hidden	FALSE						

Software Release							
States	Propose	Plan	Implement	Test	Final Acceptance	Release	Supercede
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	read,show	read,show
Owner	all	all	all	all	all	all	all
Company Name	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all	all
Employee	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Requirement Specification					
Revision	1,2,3,...				
Types	Requirement Specification				
Hidden	FALSE				
States	Plan	Working	Draft	Review	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE

Software Requirement Specification					
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	all	all	all	all	all
Company Name	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Employee	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Product Manager	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect
System Engineer	fromconnect,fromdisconnect	fromconnect,fromdisconnect	fromconnect,fromdisconnect	fromconnect,fromdisconnect	fromconnect,fromdisconnect

Software Task				
Revision	-			
Types	Task			
Hidden	TRUE			
States	Identify	Assign	Working	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none
Owner	all	all	all	all
Shadow Agent	all	all	all	all
System Conversion Manager	all	all	all	all

Software Task				
System Transition Manager	all	all	all	all
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Software Test Case				
Revision	1,2,3,...			
Types	Test Case			
Hidden	FALSE			
States	Propose	Validate	Complete	
Revisionable	TRUE	TRUE	TRUE	
Versionable	FALSE	FALSE	FALSE	
Auto Promote	FALSE	FALSE	FALSE	
Checkout History	TRUE	TRUE	TRUE	
Public	none	none	none	
Owner	all	all	all	
Shadow Agent	all	all	all	
System Conversion Manager	all	all	all	
System Transition Manager	all	all	all	
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	
Software Quality Engineering	all	all	all	
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	
Product Manager	fromconnect,toconnect,fromdisconnect,todisconnect,	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect	
System Engineer	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	

Software Test Specification					
Revision	1,2,3,...				
Types	Test Specification				

Software Test Specification						
Hidden	FALSE					
States	Plan	Working	Draft	Review	Complete	
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	
Public	none	none	none	none	none	
Owner	all	all	all	all	all	
Shadow Agent	all	all	all	all	all	
System Conversion Manager	all	all	all	all	all	
System Transition Manager	all	all	all	all	all	
Company Name	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	
Employee	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,checkout,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	

Source Code						
Revision	-					
Types	Source Code Directory	Source File				
Hidden	FALSE					
States	Create	Develop	In Test	Review	Release	Superceded
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none

Source Code						
Owner	all	all	all	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,changeowner,revise,tocconnect,todisconnect,execute,viewform,show	read,checkout,revise,tocconnect,todisconnect,execute,viewform,show
Product Manager	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	none	none	none	none	none
System Engineer	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	none	none	none	none	none
Software Engineer Manager	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	read,checkout,checkin,lock,unlock,changeowner,revise,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	read,checkout,checkin,lock,unlock,changeowner,revise,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	none	none	none
Software Engineer	read,checkout,checkin,lock,unlock,changeowner,create,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	read,checkout,checkin,lock,unlock,changeowner,revise,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	read,checkout,checkin,lock,unlock,changeowner,revise,fromconnect,tocconnect,fromdisconnect,todisconnect,execute,grant,show	none	none	none
Company Name	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show
Employee	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show	read,checkout,tocconnect,todisconnect,execute,viewform,show
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all

Source File						
Revision	1,2,3,...					
Types	Source File					

Source File						
Hidden	FALSE					
States	Create	Assign	Develop	Unit Test	Validate	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none	none
Owner	all	all	all	all	all	all
Company Name	read,checkout,toc nnect,todisconnect ,execute,viewform ,show	none	none	none	none	read,show
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Employee	read,checkout,toc nnect,todisconnect ,execute,viewform ,show	none	none	none	none	read,show

Source File

Software Engineer	read,modify,check out,checkin,lock,unlock,changeowner,promote,demote,revise,schedule,override,enable,disable,create,revise,changevault,changeName,changePolicy,changeType,fromconnect,tconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoked,show filter: context.user ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.revision context.user ~ to[Employee].businessobject.name context.user ~ to[Business Unit Employee].businessobject.name	read,modify,check out,changeowner,promote,demote,revise,fromconnect,tconnect,fromdisconnect,todisconnect ,execute,viewform ,show filter: context.user ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.revision context.user ~ to[Employee].businessobject.name	read,modify,check out,changeowner,promote,demote,revise,fromconnect,tconnect,fromdisconnect ,execute,viewform ,show filter: context.user ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.revision context.user ~ to[Employee].businessobject.name	read,modify,check out,changeowner,promote,demote,revise,fromconnect,tconnect,fromdisconnect ,execute,viewform ,show filter: context.user ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.revision context.user ~ to[Employee].businessobject.name	read,modify,check out,changeowner,promote,demote,revise,fromconnect,tconnect,fromdisconnect ,execute,viewform ,show filter: context.user ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.revision context.user ~ to[Employee].businessobject.name	read,modify,check out,changeowner,promote,demote,revise,fromconnect,tconnect,fromdisconnect ,execute,viewform ,show filter: context.user ~ to[Assigned Requirement].businessobject.name context.user.assign ment ~ to[Assigned Requirement].businessobject.revision context.user ~ to[Employee].businessobject.name

Sourcing ECR			
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	none	none	none
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,create,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,changevault,fromconnect,toconnect,fromdisconnect,todisconnect,execute,modifyform,viewform,show	read,changepolicy,show
Supplier	read,modify,delete,lock,unlock,promote,fromconnect,fromdisconnect,show	read,show	none
Sales Engineer	read,modify,delete,lock,unlock,promote,fromconnect,fromdisconnect,show	read,show	none
System Transition Manager	all	all	all
System Conversion Manager	all	all	all
Shadow Agent	all	all	all
Buyer	none	read,show	none
Customer	none	read,show	none

Sourcing Template Obsolete	
Revision	
Types	Line Item Template RFQ Template
Hidden	FALSE
States	Obsolete
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read
Owner	show
Buyer	todisconnect,show
Supplier	read,show
System Transition Manager	all
System Conversion Manager	all

Sourcing Template Obsolete

Shadow Agent	all
Sales Engineer	read,show

Specification					
Revision	1,2,3,...				
Types	Specification				
Hidden	FALSE				
States	Plan	Preliminary	Draft	Review	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	TRUE	TRUE	TRUE	TRUE	TRUE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	read	none	none	none	none
Owner	all	all	all	read,modify,checkout, promote,demote,fromc onnect,toconnect,from disconnect,todisconne ct,viewform,show	read,demote,toconnect ,todisconnect,show
Shadow Agent	none	all	all	all	all
System Conversion Manager	none	all	all	all	all
System Transition Manager	none	all	all	all	all
System Engineer	fromconnect, fromdisconnect	read,modify,delete,che ckout,checkin,changeo wner,promote,create,c hangername,changeполи cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,viewform,show	read,modify,delete,che ckout,checkin,changeo wner,promote,create,c hangername,changeполи cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,viewform,show	read,modify,delete,che ckout,checkin,changeo wner,promote,create,c hangername,changeполи cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,viewform,show	none
Product Manager	toconnect,todisconnect ,fromconnect,fromdisc onnect	read,modify,delete,che ckout,checkin,changeo wner,promote,create,c hangername,changeполи cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,viewform,show	read,modify,delete,che ckout,checkin,changeo wner,promote,create,c hangername,changeполи cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,viewform,show	read,modify,delete,che ckout,checkin,changeo wner,promote,create,c hangername,changeполи cy,changetype,fromco nnect,toconnect,fromd isconnect,todisconnect ,viewform,show	none
Employee	fromconnect, fromdisconnect	read,checkout,create,fr omconnect,toconnect,f romdisconnect,todisco nnect,viewform,show	read,checkout,create,fr omconnect,toconnect,f romdisconnect,todisco nnect,viewform,show	read,checkout,create,fr omconnect,toconnect,f romdisconnect,todisco nnect,viewform,show	read,demote,toconnect ,todisconnect,show

Specification Office		
Revision	-	
Types	Specification Office	
Hidden	FALSE	
States	Inactive	Active
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	all
Employee	read,toconnect,todisconnect,show	read,toconnect,todisconnect,show
Organization Manager	all	all
Specification Manager	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Specification Office Manager	read,fromconnect,toconnect,todisconnect,show	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeaname,changepolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Global Template Administrator	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Template Administrator	read,fromconnect,toconnect,fromdisconnect,todisconnect,show	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all	all
System Conversion Manager	all	all
Shadow Agent	all	all

Specification Template Holder	
Revision	-
Types	Specification Template Holder
Hidden	FALSE
States	Exists
Revisionable	FALSE

Specification Template Holder	
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Employee	read,toconnect,todisconnect,show
Organization Manager	all
Global Template Administrator	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
Template Administrator	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
Specification User	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Standards Manual	
Revision	-
Types	Standards Manual
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,checkout,show
Owner	read,modify,checkout,checkin,lock,unlock,fromconnect,tocconnect,fromdisconnect,todisconnect,show
Quality Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Substitute	
Revision	-
Types	Substitute
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	TRUE
Public	all
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Supplier Development Plan			
Revision	-		
Types	Supplier Development Plan		
Hidden	FALSE		
States	Draft	Active	Complete
Revisionable	FALSE	FALSE	FALSE
Versionable	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE
Public	none	none	none
Owner	read,modify,changeowner,show	read,modify,changeowner,show	all
Supplier Development Manager	all	all	all
Quality Manager	all	all	all
Shadow Agent	all	all	all
System Conversion Manager	all	all	all
System Transition Manager	all	all	all
Company Name	read,checkout,show	read,checkout,show	read,checkout,show
Employee	read,checkout,show	read,checkout,show	read,checkout,show

Supplier Development Plan			
Supplier Representative	none	read,checkout,checkin,show	read,checkout,show
Supplier Engineer	none	read,checkout,checkin,show	read,checkout,show

Supplier Line Item	
Revision	1,2,3,...
Types	Supplier Line Item
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	modify,delete,toconnect,todisconnect,show
Buyer	read,modify,delete,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Supplier	modify,create,fromconnect,toconnect,fromdisconnect,todisconnect filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
Access Grantor	read,grant,revoke,show
Customer	read,modify,delete,create,fromconnect,toconnect,fromdisconnect,todisconnect,show
Sales Engineer	modify,create,fromconnect,toconnect,fromdisconnect,todisconnect filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

Supplier Line Item Price Period	
Revision	
Types	Supplier Line Item Price Period

Supplier Line Item Price Period	
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,revise,changevault,changeusername,changeusername,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show
Buyer	show
Supplier	read,delete,create,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Customer	show
Sales Engineer	read,delete,create,toconnect,todisconnect,show filter: attribute[Primary Key] == " context.user.property[Company Key].value == attribute[Primary Key]

Supplier Scorecard	
Revision	-
Types	Supplier Scorecard
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	read,modify,checkout,checkin,changeowner,changevault,tconnect,todisconnect,show

Supplier Scorecard	
Quality Manager	all
Supplier Development Manager	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all
Company Name	read,checkout,show
Supplier Representative	read,checkout,show
Employee	read,checkout,show

Suspended	
Revision	
Types	RFQ Quotation
Hidden	FALSE
States	Suspended
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Buyer	changepolicy
Supplier	none
Access Grantor	read,grant,revoke,show

Task Escalation Message	
Revision	
Types	Task Escalation Message
Hidden	FALSE
States	Exists

Task Escalation Message	
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Buyer	none
Supplier	none
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Technical Specification					
Revision	A,B,C,...				
Types	Technical Specification				
Hidden	FALSE				
States	Draft	Peer Review	Formal Approval	Issued	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	read,modify,delete,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeiname,changepolicy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show	read,checkout,checkin,lock,unlock,create,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,checkout,create,revise,fromconnect,toconnect,todisconnect,execute,show	read,checkout,create,revise,toconnect,todisconnect,show	read,demote,toconnect,todisconnect,show
<i>This policy contains an admin property called PolicyClassification with the value General.</i>					

Technical Specification					
Supplier	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name	read,show filter: context.user.name ~~ to[Recommended Suppliers].businessobj ect.from[Employee].b usinessobject.name
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Product Manager	fromconnect,toconnect ,fromdisconnect,todisc onnect,	fromconnect,toconnect ,fromdisconnect,todisc onnect	fromconnect,toconnect ,fromdisconnect,todisc onnect	fromconnect,toconnect ,fromdisconnect,todisc onnect	none
System Engineer	fromconnect,fromdisc onnect,	fromconnect,fromdisc onnect,	fromconnect,fromdisc onnect,	fromconnect,fromdisc onnect,	none
<i>This policy contains an admin property called PolicyClassification with the value General.</i>					

Technical Specification Template					
Revision	A,B,C,...				
Types	Technical Specification				
Hidden	FALSE				
States	Draft	Peer Review	Formal Approval	Approved	Obsolete
Revisionable	TRUE	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none	none
Owner	read,modify,delete,che ckout,checkin,lock,unl ock,changeowner,sche dule,override,enable,di sable,create,revise,cha ngevault,change name, changepolicy,changety pe,fromconnect,tocon nect,fromdisconnect,to disconnect,freeze,thaw ,execute,modifyform,v iewform,grant,revoke, show	read,checkout,checkin, lock,unlock,create,revi se,fromconnect,toconn ect,fromdisconnect,tod isconnect,execute,sho w	read,checkout,create,r evise,fromconnect,toc onnect,todisconnect,ex ecute,show	read,checkout,create,r evise,fromconnect,toc onnect,fromdisconnect ,todisconnect,show	read,demote,toconnect ,todisconnect,show

Technical Specification Template					
Specification Manager	read,checkout,checkin,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,checkout,create,tomconnect,fromdisconnect,todisconnect,show	read,checkout,create,fromomconnect,toconnect,todisconnect,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,toconnect,fromdisconnect,todisconnect,execute,show
Specification Office Manager	read,modify,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeowner,changegettype,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changegettype,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,changeowner,changegettype,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,changeowner,promote,demote,create,revise,changeowner,changegettype,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: context.user ~~to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name	read,checkout,checkin,lock,unlock,changeowner,promote,demote,changeowner,fromconnect,toconnect,fromdisconnect,todisconnect,show filter: context.user ~~to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Specification Restricted User	none	none	none	read,checkout,show	none
Global User	read,modify,checkout,checkin,lock,unlock,changeowner,schedule,override,enable,disable,create,revise,changevault,changeowner,changegettype,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant,revoke,show filter: context.user ~~to[CoOwned].businessobject.to[CoOwne].businessobject.name	read,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show filter: context.user ~~to[CoOwned].businessobject.to[CoOwne].businessobject.name	read,checkout,create,revise,fromconnect,tomconnect,todisconnect,execute,show filter: context.user ~~to[CoOwned].businessobject.to[CoOwne].businessobject.name	read,checkout,create,revise,toconnect,fromdisconnect,todisconnect,show filter: context.user ~~to[CoOwned].businessobject.to[CoOwne].businessobject.name	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,changevault,changeowner,changegettype,changetype,fromconnect,fromdisconnect,todisconnect,freeze,thaw,modifyform,viewform,grant,revoke filter: context.user ~~to[CoOwned].businessobject.to[CoOwne].businessobject.name
Global Template Administrator	read,checkout,checkin,lock,unlock,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,checkout,create,fromomconnect,toconnect,fromdisconnect,todisconnect,show	read,checkout,create,tomconnect,todisconnect,execute,show
Template Administrator	read,checkout,lock,unlock,create,toconnect,todisconnect,execute,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,show	read,checkout,create,fromomconnect,toconnect,fromdisconnect,todisconnect,show	read,checkout,create,tomconnect,todisconnect,show
Supplier	read,show filter: context.user.name ~~to[Recommended Suppliers].businessobject.to[Employee].businessobject.name	none	none	none	none

Technical Specification Template					
System Transition Manager	all	all	all	all	all
System Conversion Manager	all	all	all	all	all
Shadow Agent	all	all	all	all	all
Product Manager	fromconnect,toconnect,fromdisconnect,todisconnect,	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect	fromconnect,toconnect,fromdisconnect,todisconnect	none
System Engineer	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	none

Template Filter	
Revision	-
Types	Template Filter
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	none
Owner	all
Employee	read,fromconnect,toconnect,fromdisconnect,todisconnect,show
Specification Office Manager	all filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Template Workspace Vault	
Revision	1,2,3,...
Types	Workspace Vault
Hidden	FALSE

Template Workspace Vault	
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,show
Owner	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Test Case				
Revision	1,2,3,...			
Types	Test Case			
Hidden	FALSE			
States	Propose	Approve	Validate	Complete
Revisionable	TRUE	TRUE	TRUE	TRUE
Versionable	TRUE	TRUE	TRUE	TRUE
Auto Promote	FALSE	FALSE	FALSE	FALSE
Checkout History	TRUE	TRUE	TRUE	TRUE
Public	none	none	none	none
Owner	all	all	all	all
Global User	read,show	read,show	read,show	read,show
Requirement Manager	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,changeowner,promote,demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Product Manager	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,changeowner,promote,demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Software Engineer Manager	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show

Test Case				
Software Engineer	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout,promote,demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Employee	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show filter: context.user ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.revision	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show filter: context.user ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.revision	read,modify,checkout,checkin,lock,unlock,changeowner,promote,demote,schedule,override,enable,disable,create,revise,changevault,change name,change policy,changetype,fromconnect,toconnect,fromdisconnect,todisconnect,freeze,thaw,execute,modifyform,viewform,grant, revoke,show filter: context.user ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.revision	read,modify,checkout,changeowner,promote,demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show filter: context.user ~~ to[Assigned Test Case].businessobject.name context.user.assignment ~~ to[Assigned Test Case].businessobject.revision
Shadow Agent	all	all	all	all
System Conversion Manager	all	all	all	all
System Transition Manager	all	all	all	all
System Engineer	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,

Thread		
Revision		
Types	Thread	
Hidden	FALSE	
States	Active	Inactive
Revisionable	FALSE	FALSE
Versionable	TRUE	TRUE
Auto Promote	FALSE	FALSE
Checkout History	TRUE	TRUE
Public	none	none
Owner	all	all
System Transition Manager	all	all

Thread		
System Conversion Manager	all	all
Shadow Agent	all	all
Access Grantor	read,modify,checkout,checkin,lock,unlock,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revok e,show	none
Global User	read,modify,checkout,checkin,lock,unlock,revise,fromconnect,toconne ct,fromdisconnect,todisconnect,sh ow filter: (to[Thread].from.current.access[all] == true) (to[Thread].from.current.access[rea d] == true && to[Thread].from.current.access[che ckout] == true)	todisconnect

UDA	
Revision	
Types	UDA
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read
Owner	read,modify,delete,create,fromconnect,toconnect,fromdiscon nect,todisconnect,execute,viewform,show
Buyer	read,create,show
Supplier	show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Customer	read,create,show
Sales Engineer	show
Modify access to the Buyer and Supplier roles are granted as-needed based on the value of the User Symbolic Name attribute.	

UDA	
Specification Office Manager	all filter: context.user ~~ to[Assigned To Specification Office].businessobject.from[Assigned Member].businessobject.name
Global Template Administrator	read,modify,fromconnect,toconnect,todisconnect,show
Specification Manager	read,modify,delete,create,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Modify access to the Buyer and Supplier roles are granted as-needed based on the value of the User Symbolic Name attribute.	

URL	
Revision	
Types	URL
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,toconnect,todisconnect,show
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Access Grantor	read,modify,promote,demote,change name,fromconnect,toconnect,fromdisconnect,todisconnect,execute,grant,revoke,show
Public Delete	delete filter: to[Link URL],from.current.access[delete] ~~ true

Use Case					
Revision	1,2,3,...				
Types	Use Case				
Hidden	FALSE				
States	Plan	Working	Draft	Review	Complete
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.					

Use Case					
Software Engineer Manager	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,revise ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,revise ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Product Manager	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, changeowner,promote, demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, changeowner,promote, demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Requirement Manager	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, promote,demote,create ,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, changeowner,promote, demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show	read,modify,checkout, changeowner,promote, demote,revise,fromconnect,toconnect,fromdisconnect,todisconnect,execute,viewform,show
Global User	read,show	none	none	none	none
System Engineer	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	fromconnect,fromdisconnect,	none
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.					

Version	
Revision	
Types	DOCUMENTS
Hidden	TRUE
States	Exists
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	FALSE
Public	delete,create,todisconnect
Owner	read,modify,checkin,toconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Public Modify	read, modify, revise
Public Read	read,checkout,show filter: to[Version].from.current.access[checkout] == true

Version Document	
Revision	
Types	Version Document
Hidden	FALSE
States	ACTIVE
Revisionable	FALSE
Versionable	FALSE
Auto Promote	FALSE
Checkout History	FALSE
Public	delete,create,todisconnect
Owner	read,modify,checkin,toconnect,show
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all
Public Read	read,checkout,show filter: to[Version].from.current.access[checkout] == true

Viewable Policy	
Revision	
Types	Viewable
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	read,checkout,show
Owner	all
System Transition Manager	all
System Conversion Manager	all
Shadow Agent	all

Workspace						
Revision						
Types	Workspace					
Hidden	FALSE					
States	Create	Assign	Active	Review	Complete	Archive
Revisionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Versionable	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Auto Promote	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
Checkout History	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Public	create	none	none	none	none	none
Owner	all	all	all	all	all	all
Project Lead	none	none	none	none	none	none
System Transition Manager	all	all	all	all	all	all
System Conversion Manager	all	all	all	all	all	all
Shadow Agent	all	all	all	all	all	all
Workspace Lead Grantor	promote,demote	promote,demote	promote,demote	promote,demote	delete,promote,demote	promote,demote
Access Grantor	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.						

Workspace Template	
Revision	1,2,3,...
Types	Workspace Template
Hidden	FALSE
States	Exists
Revisionable	TRUE
Versionable	TRUE

Workspace Template	
Auto Promote	FALSE
Checkout History	TRUE
Public	read,modify,fromconnect,toconnect,fromdisconnect,todisconnect,show
Owner	all
Shadow Agent	all
System Conversion Manager	all
System Transition Manager	all

Workspace Vaults	
Revision	
Types	Workspace Vault
Hidden	FALSE
States	Exists
Revisionable	FALSE
Versionable	TRUE
Auto Promote	FALSE
Checkout History	TRUE
Public	create
Owner	all
Shadow Agent	all
System Transition Manager	all
System Conversion Manager	all
Workspace Lead Grantor	promote,demote
Access Grantor	read,modify,delete,checkout,checkin,lock,unlock,revise,fromconnect,toconnect,fromdisconnect,todisconnect,grant,revoke,show
Public Read	read,checkout,show filter: attribute[Global Read] ~~ True && (to[Data Vaults].from.current.access[checkout] ~~ true to[Sub Vaults].from.current.access[checkout] ~~ true)
This policy has been renamed and its symbolic name reflects its original name. See Renamed Administrative Objects in Chapter 4.	

Setting Up and Configuring

Setting Up

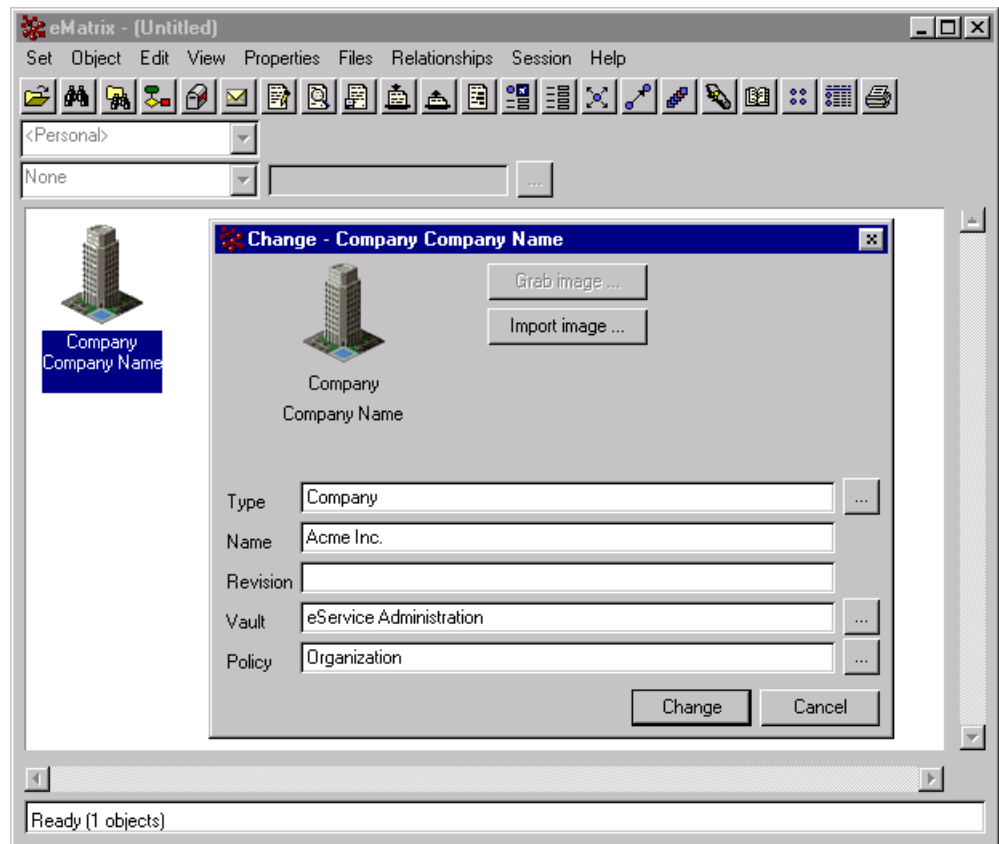
If you don't need to configure the framework or MatrixOne applications, there are only a few preliminary setup tasks that you need to perform. This section describes the setup tasks that are required, which include adding your company and its employees. Adding your company and employees lets you control employee access to the database and the MatrixOne applications. You can give different accesses to employees than to other people who need to use the applications, such as suppliers and customers.

Defining Your Company (The Host Company)

There must be a Company business object and an administrative object group defined for the host company. These objects must be derived from the objects named "Company Name" at installation. Both objects must have the exact same name.

To define a company and group for the host company

1. In Matrix Navigator, change the name of the Company business object named "Company Name" to the name of the host company.

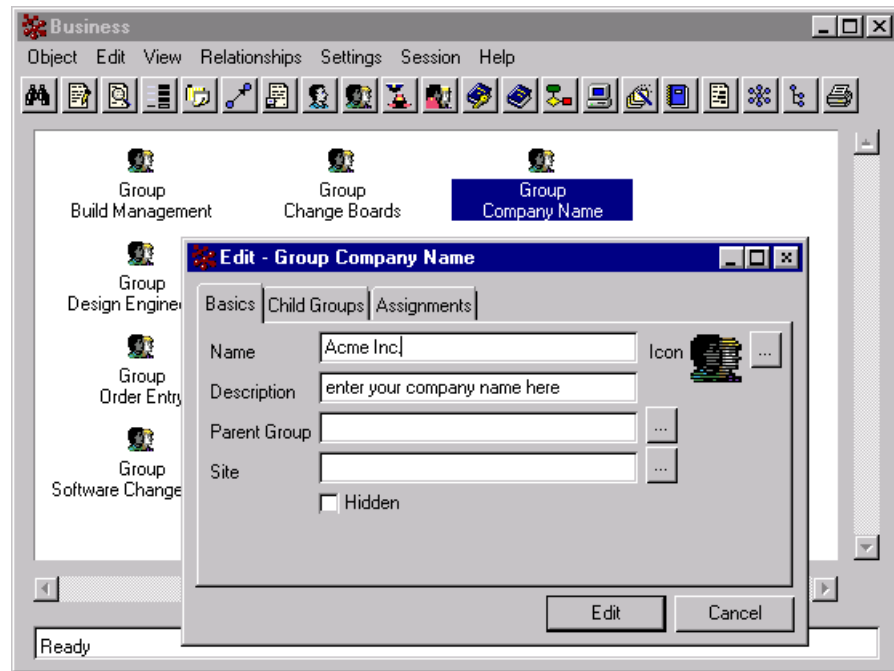


When you change the business object name, a trigger fires that changes the name of the group admin object called “Company Name” to the name you specified. To verify this change, log into Business Modeler and search for groups.

2. Before logging in as an employee of the host company, make sure you restart the application server.
3. If employees from different divisions and subsidiaries need to use the system, you can create multiple Company objects and Business Unit objects to represent the various divisions and subsidiaries. Make sure you promote any new objects of this kind to Active. To see the objects and relationships that you can use to model your company, see the data model shown in [Supplier Central Data Model Continued: Profile Management](#) in Chapter 2.

You can click Tools>Administration from the global toolbar within your MAtrixOne application to create your company structure (the above step).

4. If you created more than one Company or Business Unit object, you should use Business Modeler to create a group to represent each of these objects. Make sure the group names match the corresponding Company/Business Unit names.



Adding Employees for the Host Company

Every host company employee who needs access to the database using Matrix core applications (Navigator, Business Modeler, or MQL) must be added as a person in Business Modeler. Employees who need to log into the MatrixOne applications should also be added as Person business objects in Matrix Navigator and connected to the Company (or Business Unit) object to which they belong. All MatrixOne applications have an Administration tool that you can use to add employees. For details, see the *Common Components User Guide*. If you prefer to use Matrix Navigator, follow the below instructions. Also refer to the Administrator's guide for your application to obtain application-specific instructions for adding employees.

If the client machine is running the thick client (desktop Matrix), all .jars and .properties files must be copied from the RMI or EJB server to the thick client machine:

1. Make sure the client machine is running the same version of thick client as the RMI or EJB server.
2. Copy all .jar file from RMI-EJB/java/lib to THICK_CLIENT/java/lib.
3. Copy all .properties files from RMI-EJB/java/properties to THICK_CLIENT/java/properties.

To add an employee

1. In Business Modeler, add the person. When adding the person:
 - a) In the **Name** box of the Basics tab, enter the username you want the person to use when logging into the MatrixOne applications.

New Person

Basics | Address | Defaults | Privileges | Certificates | Assignments

Name: Meg Greene

Full Name: Meg K. Greene

Comment:

☐ Hidden

Create Cancel

- b) If you will be adding this person as a Person business object in Matrix Navigator, make a note of the exact name you enter for the person, including capitalization.
- c) In the Privileges tab, enter a Password for the person if you want the person to use a password when logging into the MatrixOne applications.
- d) Make sure **Inactive** is unchecked on the Privileges tab. Persons marked as Inactive will be unable to log in.

New Person

Basics | Address | Defaults | Privileges | Certificates | Assignments

Access: Administration Access

Password: *****

Type: ☒ Application User ☒ Full User ☐ Business Adm ☐ System Adm ☐ Inactive ☐ Trusted

Create Cancel

- e) In the **Assignments** tab, assign the person to the group that represents your company (or business unit). Assign the person to any other roles, groups, and associations as needed.

For most MatrixOne applications, the roles a person belongs to determines the tasks the person can perform in the applications. For information about how roles and groups are used by each application, refer to the administrator and user guides for that application. For instructions on how to assign access to commands in an application, see [Overview of Dynamic User Interface](#) in Chapter 8. For a description of every role and group in the framework, see [Roles](#) and [Groups](#) in Chapter 3.

There are two special roles for employees who are responsible for configuring the schema and setting up applications:

Organization Manager role—For employees who will set up customers and/or suppliers (add customer/supplier companies, add users, etc.).

Administration Manager role—For employees who will configure the framework and applications. A person must be assigned to this role to use the administration wizards that come with the framework.

2. In Matrix Navigator, create a Person business object to represent the person you added to Business Modeler in Step 1.

- a) Enter the person's name exactly as you entered it in the Name box person administrative object (Step 1a).

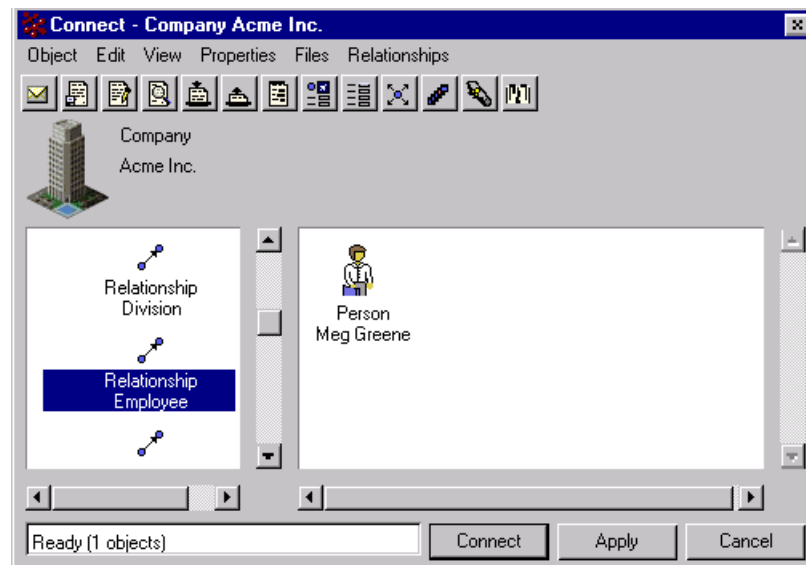
The 'Original' dialog box is shown with the following fields:

- Type: Person
- Name: Meg Greene
- Revision: -
- Vault: eService Administration
- Policy: Person

Buttons: Create, Cancel

- b) Complete as many attributes as you can.

3. In Matrix Navigator, connect the Company object (or Business Unit object) to which the person belongs to the Person object using the Employee relationship.



4. Promote the person to Active using the States browser.

Setting Up Groups

Before a group created within Matrix Business Modeler or MQL can be used within the MatrixOne applications, it must be registered and its name must be added to the StringResources properties file. For details on creating groups, see the *Matrix Business Modeler Guide* or *MQL Guide*.

To configure the group for use in MatrixOne applications

1. Register the new group with the framework using the registration wizard. For details see [Registering Your Own Administrative Objects](#).
2. Specify the group name within the appropriate `emxFrameworkStringResource.properties` file for the Web browser language or languages in use.

The naming convention for all internationalized versions of the StringResource files is `emxAPPLICATION_NAMEStringResource_LANGUAGE.properties`, where `LANGUAGE` is the standard abbreviation for the language, such as `fr` for French or `de` for German, in lowercase.

For example, the following shows how to add a group named “MatrixOne PLM” for use with a browser configured to use the English language:

- a) Open the `emxFrameworkStringResource_en.properties` file, located in `Apps/Framework/VERSION/UserInterface`.
 - b) Locate the section entitled “Group Properties.”
 - c) Append the following line to the last entry in this section:
`emxFramework.Group.MatrixOne_PLM=MatrixOne PLM`
 - d) Save and exit this file.
 - e) Redeploy the application if an exploded `.war` file is not being used.
3. Restart the Matrix Collaboration and Web/application servers.

Setting Up Vaults

The vault field is included in the search criteria and there are many configurable options available to change the behavior. The vault field is a drop-down list that contains the following list of options:

- All
- Local Vaults
- <Vault name 1>
- <Vault name 2>
-
- <Vault name n>

The first two options (All, Local Vaults) are available selections for all users. The details of these individual options are discussed in the following topics.

The drop-down list can also display a list of specific vault names available to the user for searching. The list of vaults is obtained using the “All” (`ALL_VAULT`) criteria logic. See [Definition for vault search criteria “All”](#).

When the user selects one of the available options from the vault list and clicks the Search link, the search component uses the selected vault field criteria for identifying which vaults to search. When the selection is any specific vault, the objects are searched only in the selected specific vault. When the selected option is “All” or “Local”, the vault selection is applied as described below.

Default Vault Option Selected in Search Dialog

When the general search dialog opens, one of the vault options is selected by default. This default selection is based on the following criteria and observes the following precedence:

- **Vault set in the person preference setting.** The preference is stored as the person admin property “`property_SearchDefaultVaults`.” This preference property on the Person admin object remains undefined until the user sets the default vault preference

explicitly through the personal preference page. The following table provides the list of possible values assigned to this property and the equivalent option selected in the Vault field of the search dialog.

Person Property <code>property_SearchDefaultVaults</code>	Vault Option Selected in Search Dialog
ALL_VAULTS	All
LOCAL_VAULTS	Local Vaults
DEFAULT_VAULT	Default
Vault name (symbolic name) for example: <code>vault_eServiceProduction</code>	If the property value matches one of the vaults available in the list box, that vault will be selected in the vault field. The value can have one or more vaults separated by commas.

- **System property `emxFramework.DefaultSearchVaults`.** If there is no preference is set in the Person admin object, then the property setting `emxFramework.DefaultSearchVaults` in `emxSystem.properties` is used for selecting the default vault option. The following table provides the list of possible values assigned to this system property setting and the equivalent option selected in the Vault field of the search dialog.

System Property <code>emxFramework.DefaultSearchVaults</code>	Vault Option Selected in Search Dialog
ALL_VAULTS	All
LOCAL_VAULTS	Local Vaults
DEFAULT_VAULT	Default
Vault name (symbolic name) for example: <code>vault_eServiceProduction</code>	If the property value matches with one of the vaults available in the list box, that vault will be selected in the vault field. The value can have one or more vaults separated by comma.

The default setting for this system property is:

```
emxFramework.Search.DefaultVaultPreference = ALL_VAULTS
```

- **No preferences.** If there is no preference is set for the Person admin object as well as no value set on the system property `emxFramework.Search.DefaultVaultPreference`, then the default vault option is “All.”

Vault Criteria for General Search

When the user selects one of the available options from the vault list and clicks on the search link in the bottom frame, the search component uses the vault field criteria selected to identify which vaults to search for. When the selection is any specific vault, the objects are searched for only in the selected vault. When the option selected is “All,” “Local Vaults” or “Default,” the vault selection is applied as described in the following sections.

Definition for vault search criteria “All”

When the vault criterion is specified as “All” (ALL_VAULTS), the following logic is used to obtain the list of vaults.

- **Primary Vault**—The vault of the company object to which the logged-in person object is connected with the relationship “Employee.”
- **Secondary Vault**—The list of vaults specified in the attribute [Secondary Vaults] on the company object to which the logged-in person object is connected with the relationship “Employee.” The value can be a vault list with symbolic name separated by tilde (“~”) and may have local or remote vault entries.
- **Primary and Secondary Vaults of Collaboration Partners**—The primary and secondary vaults as explained above for those Company objects that are connected to the person’s company object with relationship “Collaboration Partner.”

Definition for vault search criteria “Local Vault”

When the vault criterion is specified as “Local Vault” (LOCAL_VAULTS), the following logic is used to obtain the list of vaults.

- **Primary Vault**—The vault of the company object to which the logged-in person object is connected with the relationship “Employee.”
- **Secondary Vault (Local)**—The list of vaults specified in the attribute [Secondary Vaults] on the company object to which the logged-in person object is connected with the relationship “Employee.” The value can be a vault list with symbolic name separated by tilde (“~”) and may have local or remote vault entries. Only the local vaults defined in this list will be used for searching.

Overview of Configuring

The framework and MatrixOne applications are designed to be configured for each company that uses them. The rest of this section explains the things you should know before making changes and gives instructions for making changes.

You should perform all changes using this 2-step process:

1. Make proposed changes in the test database, with MQL scripting on so you can track all changes.
2. Create the schema scripts that will run in your pre-production and/or production database from copies of the scripts provided with the framework.

Things to Know Before Making Changes

This section describes important information about how the framework identifies and names objects. The remaining sections of the chapter, which give instructions for how to configure the framework, build on the principles described here. Please read this section before making changes or additions to the framework or applications.

How the System Identifies Administrative Objects: Symbolic Name Properties

The system identifies administrative objects using symbolic names rather than standard object names. Symbolic names have this syntax:
[administrative object type name]_[original object name with no spaces]

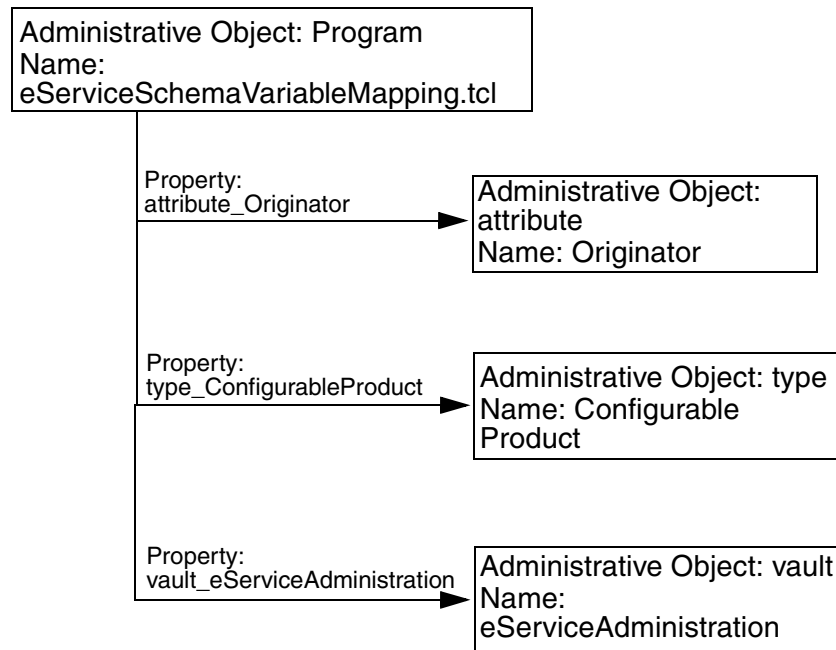
For example, the symbolic name of the Originator attribute is `attribute_Originator`. The symbolic name of the Product Manager role is `role_ProductManager`. If you change the name of the Originator attribute to `Creator`, the symbolic name remains `attribute_Originator`, allowing the system to identify the attribute even though the name has changed.

Important: Do not change the symbolic name for an administrative object. If you do, objects in the schema will no longer be able to reference the object. Additionally, new software versions will be unable to identify the object for updating.

Whenever you refer to an administrative object within a program (Tcl or JSP program), you must use the object's symbolic name and not the standard name ("`policy_Person`", not "`Person`"). You must also use the symbolic name when specifying an administrative object as an input parameter that is to be passed to a program. For example, when specifying the eService Safety Vault, which is an attribute whose value is passed to the `eServicecommonObjectGenerator.tcl` program, you must use the symbolic name for the vault.

The symbolic names for all administrative objects are stored as properties of a program object called `eServiceSchemaVariableMapping.tcl`. Each property assigns a symbolic name to an administrative object. For example, this diagram shows the properties on the program object `eServiceSchemaVariableMapping.tcl` that assign a symbolic name to an attribute, type, and vault object.

For samples of JSP code that gets the symbolic name for an admin object, see [Getting Symbolic Names](#) in Appendix A. For a sample of how to get an admin object name from a symbolic name, see [Getting Admin Object Names from Symbolic Names](#) in Appendix A.



For instructions on how to view symbolic name properties, see [Viewing Administrative Properties](#).

Tracking administrative objects using fixed symbolic names has several advantages:

- You can easily change the name of administrative objects. As long as the symbolic name doesn't change, the system can identify renamed objects. The only exception is program objects that are installed with the framework and applications. You should not change the name of these program objects, which are prefixed with "eService" or "emx".
- When you do change a name, other administrative objects in the schema that reference the object are still able to identify the object. For example, if you change the name of the Originator attribute to Creator, the types that use the attribute called Originator will now use the attribute called Creator because its symbolic name is still attribute_Originator.
- Because names can be changed without causing problems, the system changes names to prevent name collisions between objects in your existing database and objects in the framework. If the installation program finds an existing object with the same name as a framework object, it adds a prefix to the name of the framework object. The prefix is "eService" and the version number of the installation. For example, if you install the version 10.0.0.0 framework onto a database that contains a Part type, the installation program renames the Part type within the framework to "eService10000~Part".
- When upgrading to a new version, the system reliably finds the correct object to update, even if you have renamed the object and even if you have replaced the framework object with your object. As long as the symbolic name is assigned to the object you want to use, the installation program can update the object. For more information, see [Resolving Schema Similarities Between the Framework and Custom Schema](#).

Properties of Administrative Objects

Every administrative object has five properties that facilitate upgrades:

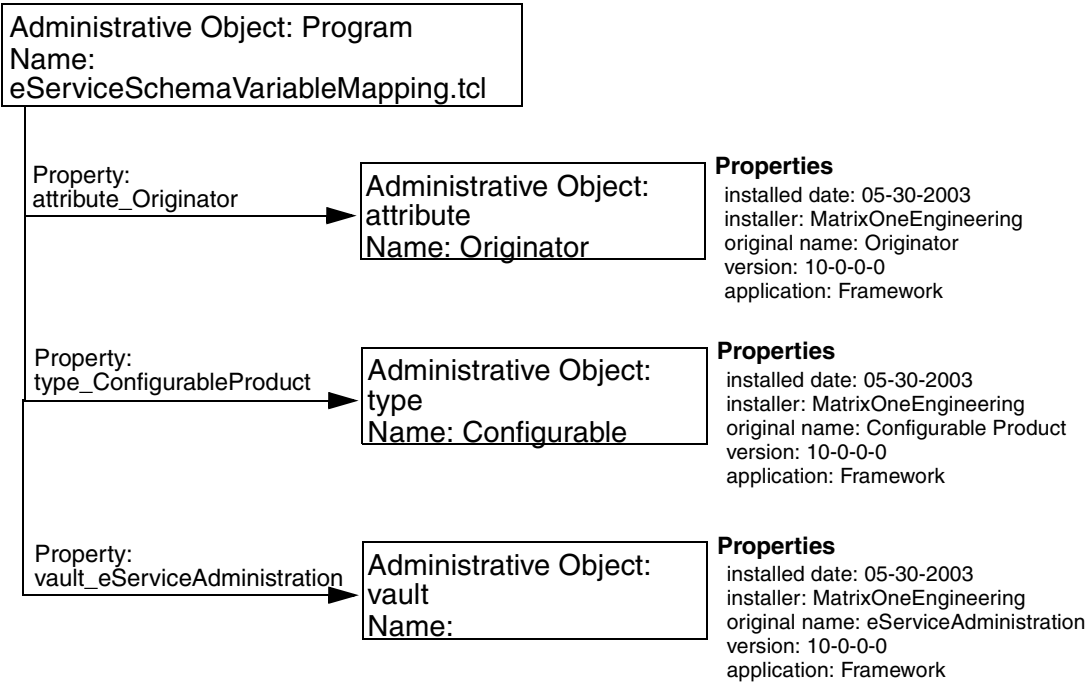
- *installed date*—current date on the system on which the install scripts ran when the object was initially installed; for example, “07-27-2003”. This date is not updated when an object is updated in subsequent releases.
- *installer*—the company name that designed the administrative object. For all objects installed with the framework and MatrixOne applications, the value is “MatrixOneEngineering”.

Never enter “MatrixOneEngineering” for the installer property in any schema that you register without explicit permission from MatrixOne Framework Engineering.

- *original name*—the intended name for the object when it was originally installed. For example, if the standard name for the Part type is changed to avoid a name collision, the original name is still Part and does not include the prefix added to the name.
- *version*—the software version number for the most recent update to the object, such as 10-0-0-0.
- *application*—the name of the application the object is installed with. For administrative objects installed with the framework (a majority of administrative objects), the value for this property is “Framework”. For administrative objects installed an application, the value is the application name with no white space, such as ProductOptionManager.

For instructions on how to view administrative properties, see [Viewing Administrative Properties](#).

The following diagram is the same as the one shown in the previous section, which shows the symbolic name properties for several administrative objects, but the five properties for each administrative object have been added.



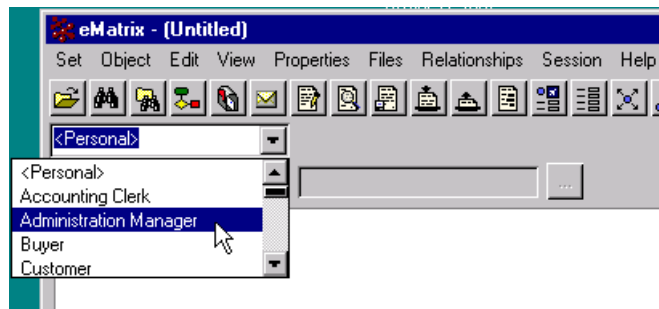
Viewing Administrative Properties

The framework contains an administrative wizard that lets you view the properties for administrative objects, including the symbolic name property and the five properties that facilitate upgrades for each object. All of these properties are described in the preceding two sections. The wizard also lets you add, delete, and modify properties (see [Registering Your Own Administrative Objects](#)).

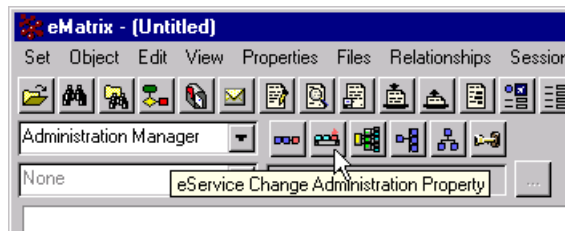
As with all administrative properties, you can list and modify these properties using MQL. The administrative wizard is a tool to help make working with the properties easier.

To view properties for an administrative object

1. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
2. Choose **Administration Manager** from the role bar.



3. From the Administration Utilities toolset, click **eService Change Administration Property**.



4. In the eService Change Administration Property wizard, make sure **eServiceSchemaVariableMapping.tcl** is selected from the Registration Program list.

5. From the **Admin Type** list, choose the type of administrative object whose properties you want to view.
For example, suppose you want to view the properties for the Originator attribute. You would choose attribute from the Admin Type list.
6. Click **Retrieve List**.
The wizard lists all administrative objects of that type. The objects that are assigned to a symbolic name property are listed in the Registered Admins list. The objects that are not assigned to a symbolic name property are listed in the Un-Registered Admins list.
7. From the **Registered Admins** list, choose the object for which you want to view properties.
8. Click **Retrieve Registration**.
All the properties for the object are listed in the lower left corner.

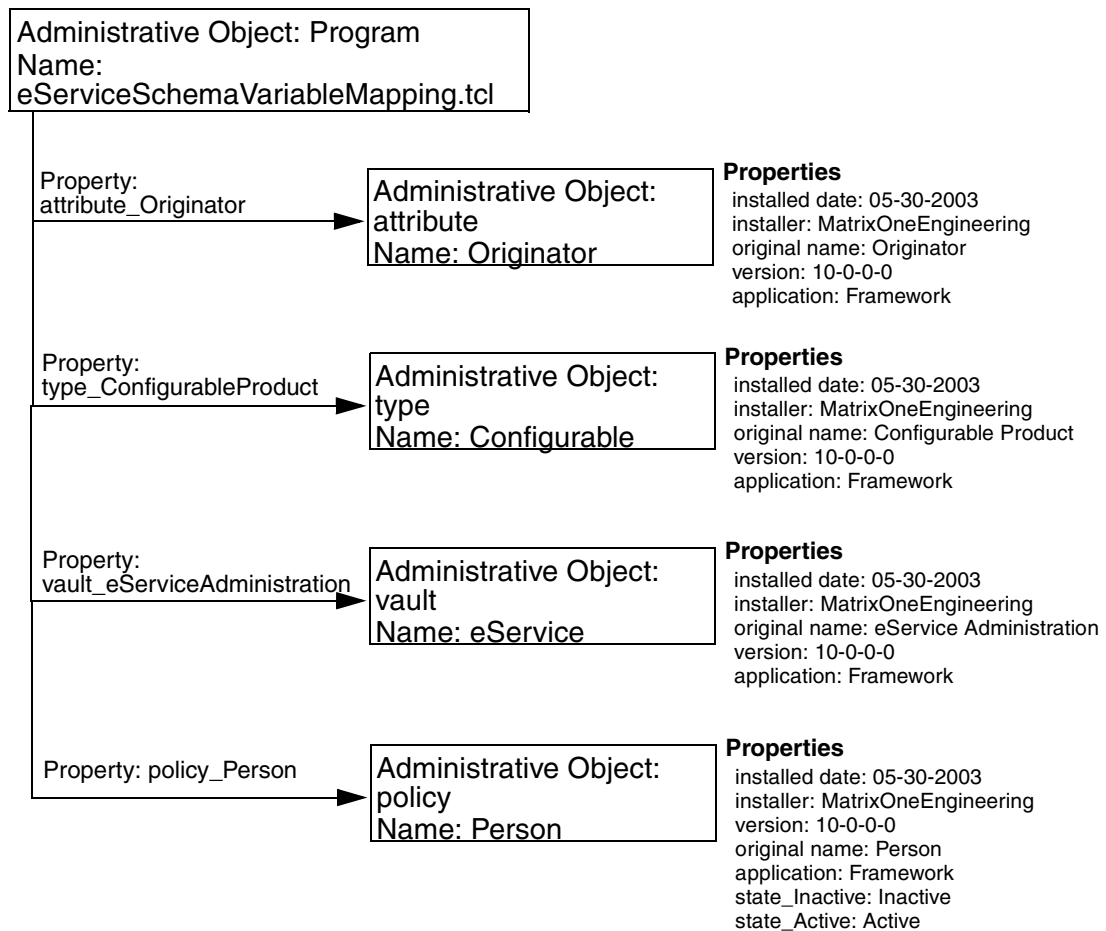
9. To view properties for another object, repeat Steps 5 through 8.

Or

To quit the wizard, click **Finish**.

How the System Identifies State Names

Like administrative objects, the system identifies policy states using symbolic names. For example, the symbolic name for the Active state of the Person policy is state_Active. Symbolic names for states are stored as properties on the policy object. This diagram, which shows all the properties for several administrative objects, is the same as the one shown in the previous section except the properties have been added for a policy.



Resolving Schema Similarities Between the Framework and Custom Schema

If you have a custom admin object that is similar to a framework object, you must decide which one to use. The objects may have the same name and/or they may have the same symbolic name. As described in “How the Installation Handles Schema Similarities” in the *Matrix PLM Platform Installation Guide*, the framework handles these two cases differently when installing. However many of the decisions and procedures for resolving the similarities are the same.

To help find similarities, look in the installation log file for admin objects that were renamed due to name collisions and MQL add and modify commands that were skipped because an object with the same symbolic name already exists in the database.

When the framework has an object that is similar to your custom object, decide whether the objects are really the same. It may be that they serve different functions and are both needed. In that case, you can rename one or both as needed.

If the objects are the same, you must decide which one to use. It’s usually most efficient to use the framework’s administrative object instead of your object. Using the framework’s object instead of your object usually involves:

- Renaming your object and the framework object so the framework object has the correct name and there are no name collisions. For example, if the version 9.0.0.0 installation program renamed the Widget type framework object due to a name collision, you could rename your Widget type to OldWidget and rename the framework's eService9000~Widget to Widget.
- Configuring the framework object so it has the attributes and other characteristics that your object has.
- Making sure any of your existing objects, such as programs and policies, that reference the object you are replacing point to the framework object instead.
- Migrating data from your admin object to the framework's (if the object's been used in production).

Another alternative is to use your object instead of the framework object. This alternative usually involves more work than using the framework object. Replacing a framework object with your own object involves:

- Pointing the symbolic name and the five administrative properties to your object instead of the framework's object. For instructions, see [Merging Schema](#).
- Configuring your object so it contains the attributes and other characteristics the framework expects it to have.
- Changing all references within the framework to your object instead of the framework object. For example, if you want to use your Widget type instead of the framework's Widget type, you would have to make sure all relationships and policies that use the framework's Widget type now use your Widget type.

Reserved Prefixes, Names, and Characters

This section contains general guidelines for entering data and names.

All Fields

The tilde (~) and pipe (|) special characters are reserved for use in the programming code for the framework and MatrixOne applications. Do not use these characters when entering data for any field for the framework or any MatrixOne application, including fields in application Web pages, Matrix Navigator, and Business Modeler. This restriction includes names, descriptions, comments, and all other attributes and properties.

Administrative Object Names

The framework and MatrixOne applications are designed to let you use your exact business terminology rather than cryptic words that have been modified to conform to the computer system limitations. Names are case-sensitive and spaces are allowed. You can use complete names rather than contractions, making the terminology in your system easier for people to understand. Generally, name lengths can be a maximum of 127 characters. Leading and trailing spaces are ignored.

You should avoid using characters that are programmatically significant to the applications. These characters are specified as follows for Administrative Object Names:

@ , * ? [] # \$ { } \ \ " < > |

These characters are specified as follows for Administrative Object Descriptions, and other related Text fields:

\$ { }

To configure the system to allow the use of the above-listed special characters, see [Special Character Properties](#) in Chapter 7.

Also, do not prefix any administrative object (types, relationships, attributes, programs, wizards, etc.) with “eService”.

Users should never use special characters for companies, business units, or persons, or for any attribute whose value is used in a Tcl or MQL script.

Program Objects

Do not prefix program objects with any of the following strings:

bid, bom, build, common, draw, engchg, prodcfg, pom, profile, softchg, softqmt, supplyacc, ui, or eService.

Do not prefix any administrative objects (types, relationships, attributes, programs, wizards, etc.) with “eService”.

Configuring the Schema

This section contains instructions for configuring the names of administrative objects, business objects, and policy states. Before configuring names, you should read the section [Things to Know Before Making Changes](#).

If you customize objects added by MatrixOne installations, then you must Register that object for future upgrades to properly upgrade that object. If not Registered, those objects will cause the upgrades to fail.

Changing an Administrative Object Name

You can rename administrative objects in the framework as you would any administrative object. As described in [How the System Identifies Administrative Objects: Symbolic Name Properties](#), renaming objects does not affect the system's ability to identify the object. In Business Modeler, find the object. Right-click on it, select Open>Edit and change the name. In MQL, use the modify ITEM NAME statement to make the change.

Keep these points in mind when changing admin object names:

- The framework uses string resource properties to manage internationalization of schema names. The string resource IDs for admin objects are defined using the actual admin object name. Therefore, if you change an admin object name and want the name to be displayed on internationalized versions of applications, you need to update the string resource ID in addition to the value for the ID. For instructions, see [Translating Schema Names](#) in Chapter 6.
- Do not use underscore _ in admin object names because that is used in symbolic names and could affect parsing of the string.
- Labels and other text for UI components—such as My Desk and Actions menus, page headings, and tree category labels—that contain “words” that are the same as admin object names are NOT updated automatically when schema is renamed. Onscreen text for UI components are defined, like other onscreen text, using string resource properties. Therefore, the string resource value must be changed to change the UI component text. Note that macros and expressions are not used for UI component text because they cannot be internationalized.

For example, suppose you rename the Part type to Item type. Onscreen text that is defined using code that retrieves data from the database, such as a select expression that displays the type name in a table, will show the updated name, Item. If you updated the internationalized string resource ID and value as described in [Translating Schema Names](#) in Chapter 6, the international versions will also display the updated type name. However, a menu with the label “Create Part” will not be updated because the label is defined using a string resource and not using a macro or select expression. To update the menu label, you need to change the value for the string resource from “Create Part” to “Create Item”. For instructions on changing onscreen text using string resource properties, see [Changing Onscreen Text](#).

The only administrative objects whose names you shouldn't change are the names of program objects installed with the framework and with the MatrixOne applications. The names for all MatrixOne application programs are prefixed with “eService[name of install directory]”. Programs in the Schema directory are prefixed with “eService” or the application name only. Do not change this prefix and don't create custom programs using

the eService or emx prefixes. If you need to modify the name of a MatrixOne application program, clone the program and change the name of the cloned program.

If you rename objects that are referenced in policy expression accesses, you will need to update any expressions that reference the object names. See [Administrative Objects Used in Expression \(Filter\) Accesses](#) in Chapter 3.

Merging Schema

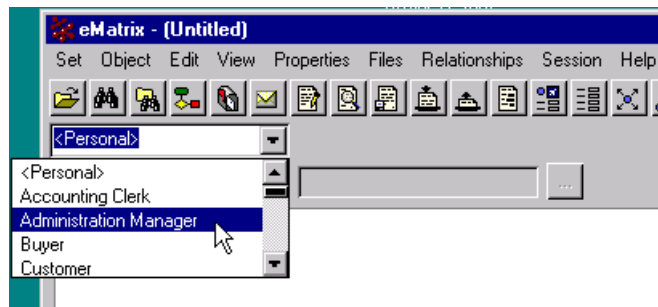
If you decide to use an existing administrative object in place of a framework object, you must transfer the administrative properties for the object to your object instead of the framework object. These properties include the symbolic name assigned to the framework object and the five properties that facilitate upgrades. As described in [Resolving Schema Similarities Between the Framework and Custom Schema](#), it is usually more efficient to configure the framework object to suit your needs than to replace it with another object.

Symbolic names for administrative objects are stored as properties of the program object eServiceSchemaVariableMapping.tcl. Each property assigns a symbolic name to an administrative object. When reassigning symbolic names, it is important that you:

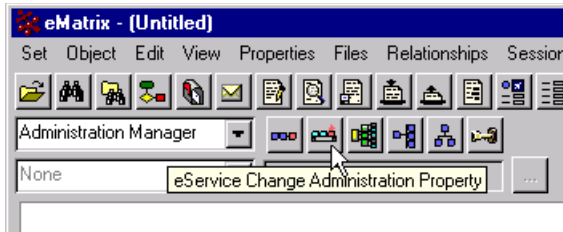
- Do not change symbolic names. For example, the symbolic name for the Part type is type_Part. If you want to use a type called Subpart instead of the framework Part type, you can point the symbolic name to the existing Subpart type but you should not change the symbolic name to type_Subpart. Changing a symbolic name could prevent the system from calling the object from within programs and from correctly updating the object when new software versions are installed.
- Make sure the symbolic name is assigned to the object you want the schema to use and that it is assigned to only one object.

To transfer administrative properties for an administrative object

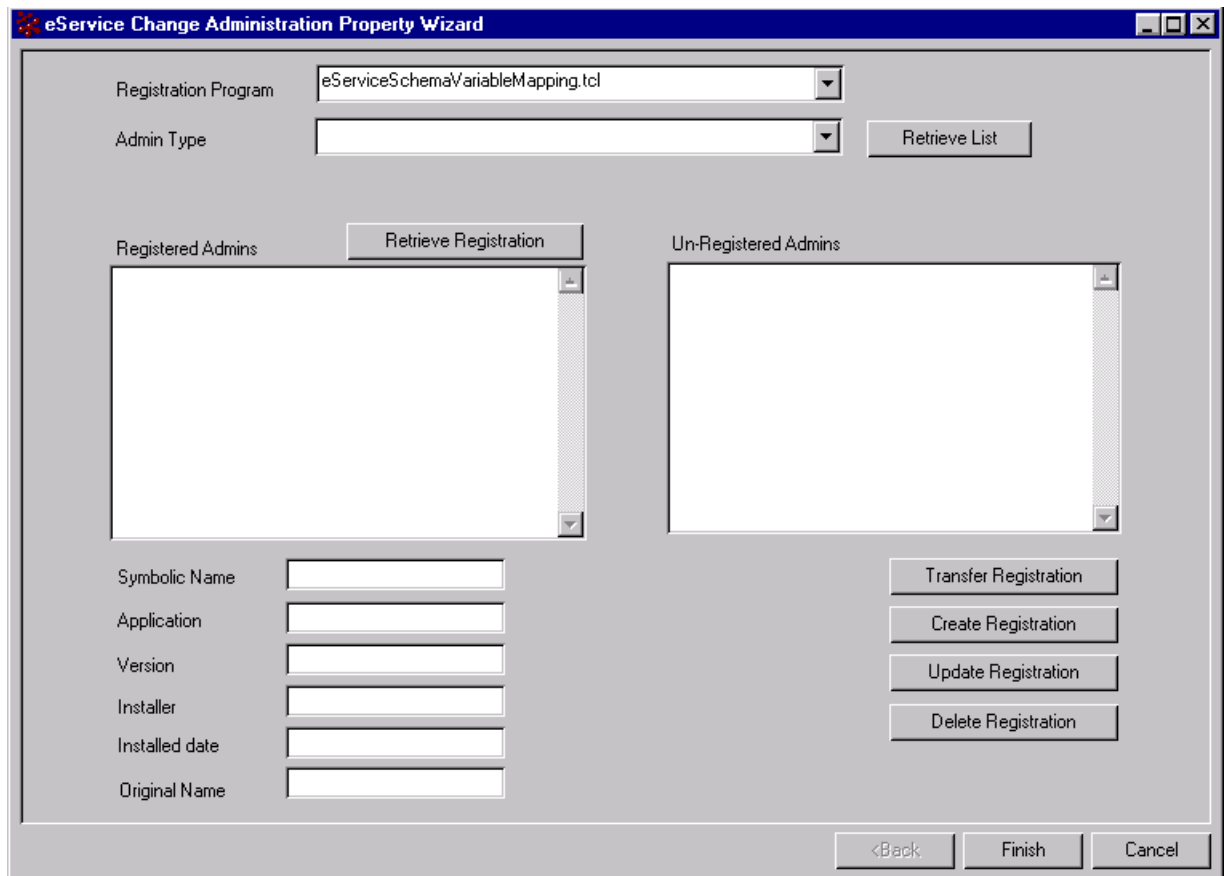
1. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
2. Choose **Administration Manager** from the role bar.



3. From the Administration Utilities toolset, click **eService Change Administration Property**.



4. Make sure **eServiceSchemaVariableMapping.tcl** is selected from the Registration Program list.



5. From the **Admin Type** list, choose the type of administrative object that you are reassigning.

For example, suppose you want to use your own type called Subpart instead of the framework's Part type. You would choose type from the Admin Type list.

6. Click **Retrieve List**.

The wizard lists all administrative objects of that type. The objects that are assigned to a symbolic name property are listed in the Registered Admins list. The objects that are not assigned to a symbolic name property are listed in the Un-Registered Admins list.

Using the example described above, Part would be in the registered list and SubPart would be in the unregistered list.

7. From the **Registered Admins** list, choose the object you want to reassign. This is the object that you want to replace with the unregistered object.
Using the example described above, you would choose Part.
8. Click **Retrieve Registration**.
All the properties for the object are listed in the lower left corner.

The screenshot shows the 'eService Change Administration Property Wizard' window. At the top, there are two dropdown menus: 'Registration Program' set to 'eServiceSchemaVariableMapping.tcl' and 'Admin Type' set to 'type'. To the right of these is a 'Retrieve List' button. Below these are two list boxes. The 'Registered Admins' list on the left contains: Media, Option, Option Rule, Organization, **Part** (highlighted), Part Rule, Person, and Process. Above this list is a 'Retrieve Registration' button. The 'Un-Registered Admins' list on the right contains: Subpart. Below the 'Registered Admins' list are seven text input fields: 'Symbolic Name' (type_Part), 'Application' (Framework), 'Version' (8-0-0-0), 'Installer' (MatrixOneEngineering), 'Installed date' (07-27-2000), and 'Original Name' (Part). To the right of these fields are four buttons: 'Transfer Registration', 'Create Registration', 'Update Registration', and 'Delete Registration'. At the bottom right of the window are three buttons: '<Back', 'Finish', and 'Cancel'.

9. From the **Un-Registered Admins** list, choose the object that you want to use instead of the registered object. Also make sure the registered object is still selected.
Using the example described above, you would choose SubPart and Part.
10. Click **Transfer Registration**.
The wizard switches the two objects so the symbolic name property now points to the previously unregistered object. The reassigned object also inherits the values for the other five properties.
Using the example described above, SubPart would now be registered and would be assigned to the type_Part property.

Registering Your Own Administrative Objects

You can register your own administrative objects (those that are not part of the framework) using the same properties used for registering framework objects. Remember to register the symbolic name for any custom administrative objects that you refer to in a program. If you use an unregistered symbolic name in a program, the program will not work.

The eService Change Administration Property wizard lets you add, delete, and modify the properties for administrative objects, including the symbolic name property and the five properties that facilitate upgrades for each object. These properties are described in [How the System Identifies Administrative Objects: Symbolic Name Properties](#) and [Properties of Administrative Objects](#). As with all administrative properties, you can add, modify, and delete administrative properties using MQL. The administration wizard is a tool to help make working with the properties easier.

Using symbolic names for your own administrative objects lets you use the same lookup methodology used in the framework. This lookup methodology ensures that programs call the correct objects even if objects are renamed. Alternatively, you can use your own lookup methodology.

When entering symbolic names for custom admin objects, do not use the exact same naming convention as the framework uses unless you are working closely with MatrixOne Engineering to add schema. You can use a slight variation of the naming convention, such as adding an abbreviation for your company's name. For example, instead of type_Part, a company with the abbreviation ACM could add type_ACMPart.

Do not register schema that you anticipate to be in an upcoming release of the framework without explicit permission from MatrixOne Framework Engineering. They must provide you with the registration information for the application and version properties. If you have not received instructions from Framework Engineering, do not enter “MatrixOneEngineering” for the installer property in any schema that you register.

IMPORTANT: *Do not change the properties for framework objects. The installation program maintains and updates these properties. Changing the administrative properties for framework objects can prevent the system from calling the objects within programs and from updating objects correctly when new versions are installed.*

To register an administrative object that’s not part of the framework

1. Using Business Modeler or MQL, make sure the administrative object is added to the database.
2. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
3. Choose **Administration Manager** from the role bar.
4. From the Administration Utilities toolset, click **eService Change Administration Property**.
5. Choose the **Registration Program** you are using to register the object.

The screenshot shows the 'eService Change Administration Property Wizard' dialog box. It has a title bar with the text 'eService Change Administration Property Wizard' and standard window controls. The main area contains several fields and buttons. At the top, there is a 'Registration Program' dropdown menu with 'customregistration.tcl' selected, and an 'Admin Type' dropdown menu. To the right of the 'Admin Type' dropdown is a 'Retrieve List' button. Below these are two large list boxes: 'Registered Admins' on the left and 'Un-Registered Admins' on the right. Above the 'Registered Admins' list is a 'Retrieve Registration' button. Below the list boxes are several input fields: 'Symbolic Name', 'Application', 'Version', 'Installer', 'Installed date', and 'Original Name'. To the right of these fields are four buttons: 'Transfer Registration', 'Create Registration', 'Update Registration', and 'Delete Registration'. At the bottom of the dialog are three buttons: '<Back', 'Finish', and 'Cancel'.

6. From the **Admin Type** list, choose the type of administrative object that you are adding.

For example, suppose you are adding a type called SubType. You would choose type from the Admin Type list.

7. Click **Retrieve List**.

The wizard lists all administrative objects of that type that are registered with the program you selected. The objects that are assigned to a symbolic name property are listed in the Registered Admins list. The objects that are not assigned to a symbolic name property are listed in the Un-Registered Admins list.

8. From the **Un-Registered Admins** list, choose the object you want to add properties for.

9. Enter values for the properties as needed. If you are using the framework's convention for symbolic names, make sure you use the format [administrative type]_[object name with no spaces].

The screenshot shows the 'eService Change Administration Property Wizard' window. At the top, there are two dropdown menus: 'Registration Program' with 'customregistration.tcl' selected and 'Admin Type' with 'attribute' selected. To the right of these is a 'Retrieve List' button. Below these are two list boxes. The 'Registered Admins' list on the left contains 'Responsible Assembly Line' and 'Target Weight'. Above it is a 'Retrieve Registration' button. The 'Un-Registered Admins' list on the right contains 'SKU number'. Below the lists are several text input fields: 'Symbolic Name' (containing 'attribute_SKUnumber'), 'Application' (containing 'internal'), 'Version' (containing '1.0.1'), 'Installer' (containing 'customer'), 'Installed date' (containing '07-27-2000'), and 'Original Name' (containing 'SKU number'). To the right of these fields are four buttons: 'Transfer Registration', 'Create Registration', 'Update Registration', and 'Delete Registration'. At the bottom right are three buttons: '<Back', 'Finish', and 'Cancel'.

10. Click **Create Registration**.

The properties are added to the registration program and the wizard lists the object in the Registered Admins list.

To remove the registration for an administrative object

1. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).

2. Choose **Administration Manager** from the role bar.
3. From the Administration Utilities toolset, click **eService Change Administration Property**.
4. Choose the **Registration Program** the object is registered with.
5. From the **Admin Type** list, choose the type of administrative object that you are working with.
6. Click **Retrieve List**.
The wizard lists all administrative objects of that type. The objects that are assigned to a symbolic name property are listed in the Registered Admins list. The objects that are not assigned to a symbolic name property are listed in the Un-Registered Admins list.
7. From the **Registered Admins** list, choose the object whose properties you want to delete.
8. Click **Delete Registration**.
The wizard adds the object to the Un-Registered Admins list and all properties for the object are deleted from the selected registration program.

To update properties for an administrative object

1. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all privileges (as defined in the person definition).
2. Choose **Administration Manager** from the role bar.
3. From the Administration Utilities toolset, click **eService Change Administration Property**.
4. Choose the **Registration Program** you are using to register the object.

Do not update properties for objects that are part of the framework (properties attached to the registration program `eServiceSchemaVariableMapping.tcl`).

5. From the **Admin Type** list, choose the type of administrative object that you want to work with.
6. Click **Retrieve List**.
The wizard lists all administrative objects of that type. The objects that are assigned to a symbolic name property are listed in the Registered Admins list. The objects that are not assigned to a symbolic name property are listed in the Un-Registered Admins list.
7. From the **Registered Admins** list, choose the object you want to update properties for.
8. Click **Retrieve Registration**.
9. Enter values for the properties as needed.
10. Click **Update Registration**.

Configuring State Names

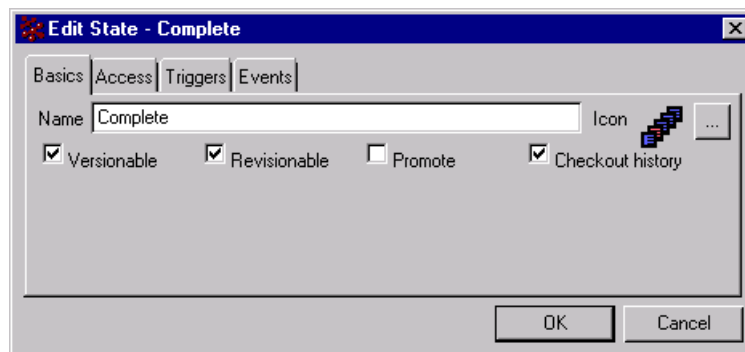
As described in [How the System Identifies State Names](#), the system tracks the names of states using symbolic name properties on policy objects. When you change the name of a policy state, you also need to change the state name that the symbolic name property points to. Similarly, when you add or remove a state from the policy, you need to add/remove the symbolic name property for the state. This section contains instructions for changing the name of a state, adding a state to a policy, and deleting a state.

This procedure describes how to change the property values for state names using a wizard in Matrix Navigator. Alternatively, you can use MQL to change the state name property using the `modify property` command.

To change the name of a policy state

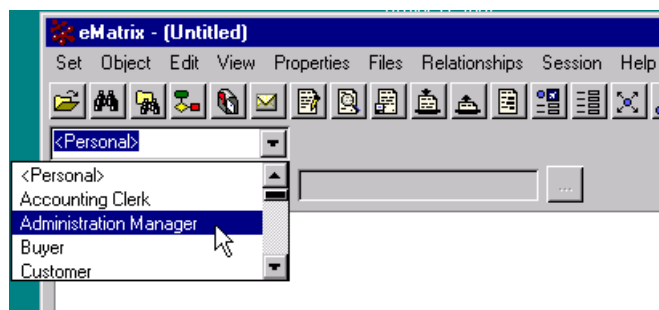
1. Change the name by editing the state as you normally would in Business Modeler or MQL.

For example, in Business Modeler, edit the policy. On the State tab of the Edit Policy dialog box, double-click the state. On the Basics tab of the Edit State dialog box, change the state name. In the sample graphic shown below, we've changed the state name from Close to Complete.

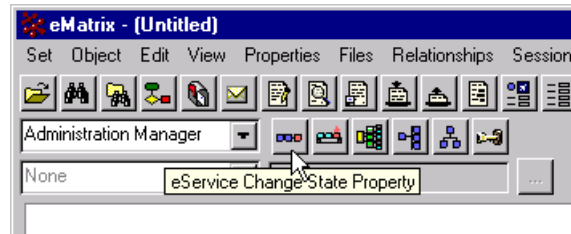


Now you need to update the symbolic name property for the state so it points to the renamed state. The remaining steps in the procedure describe how to do this.

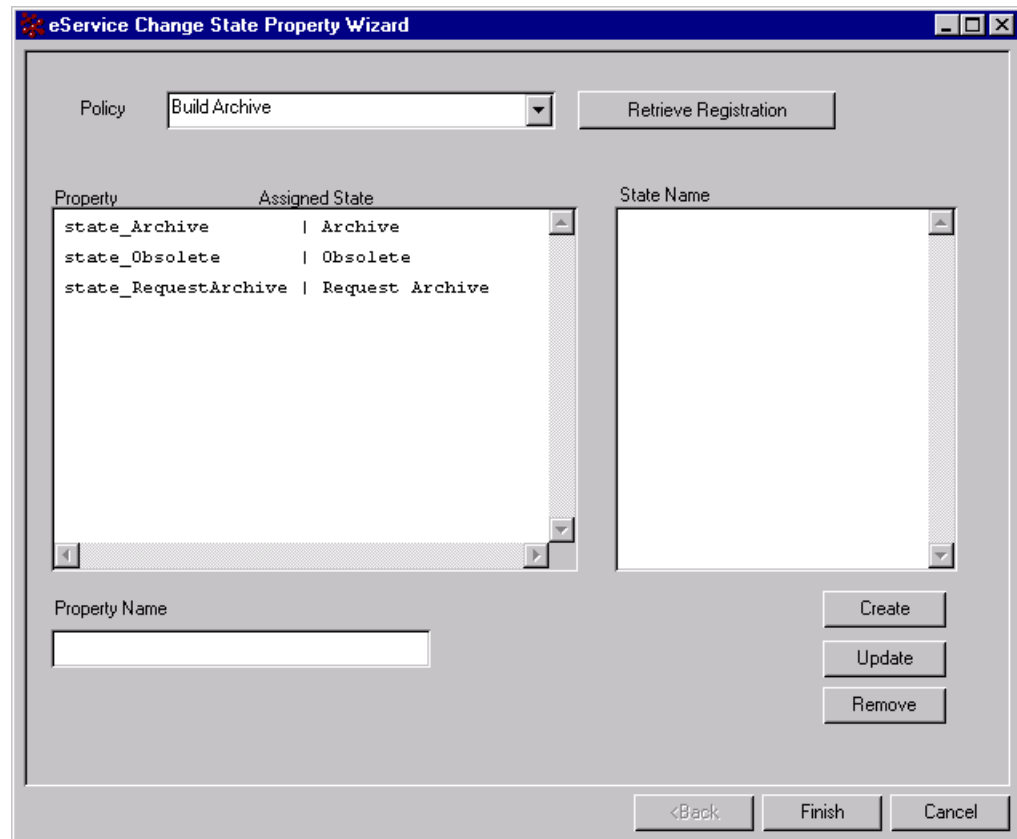
2. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all privileges (as defined in the person definition).
3. Choose **Administration Manager** from the role bar.



4. From the Administration Utilities toolset, click **eService Change State Property**.

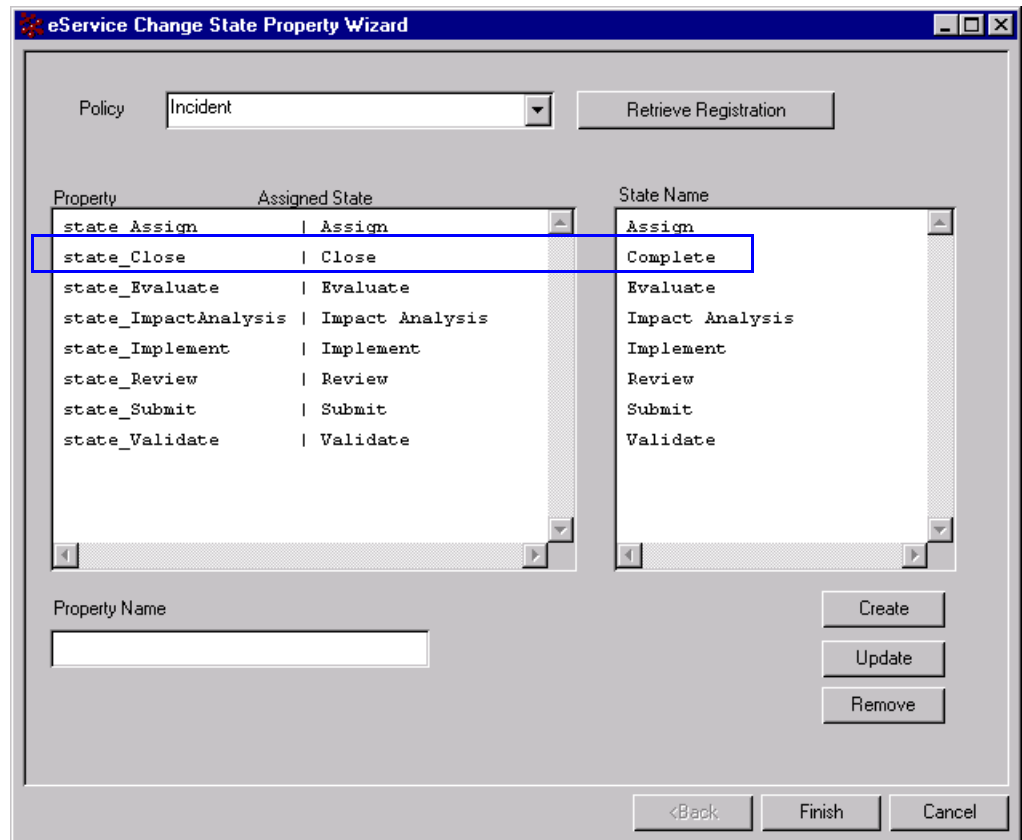


5. In the Change State Property wizard, choose the policy that contains the state you want to rename using the **Policy** drop-down list.



6. Click **Retrieve Registration**.

The wizard lists all the symbolic name state properties (Property column) added to the policy, including the state to which the property points (the Assigned State). The State Name column shows the state names currently in the policy lifecycle. The state you renamed in Step 1 is included. The goal is to make the Assigned State match the State Name. The Property does not change.



7. In the **Property/Assigned State** list, choose the state Property whose Assigned State you want to rename.

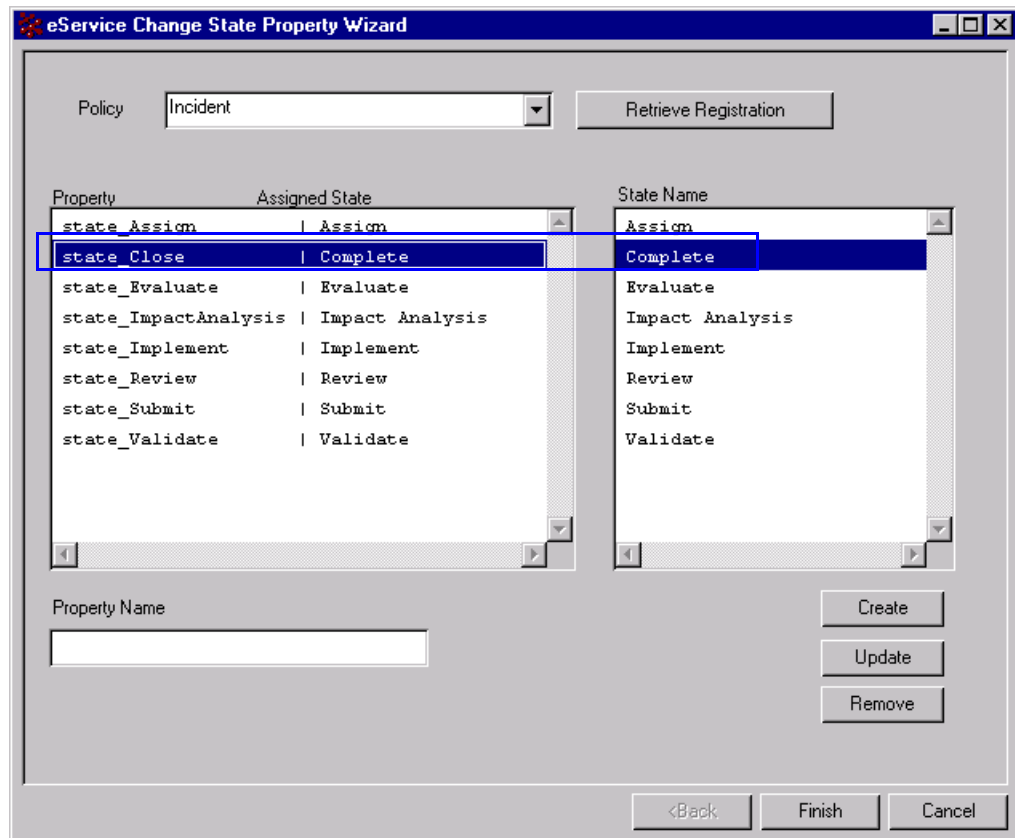
In the example shown in the graphic above, we would select state_Close Close.

8. In the **State Name** list, choose the state that you renamed.

In the example used above, we would select Complete.

9. Click **Update**.

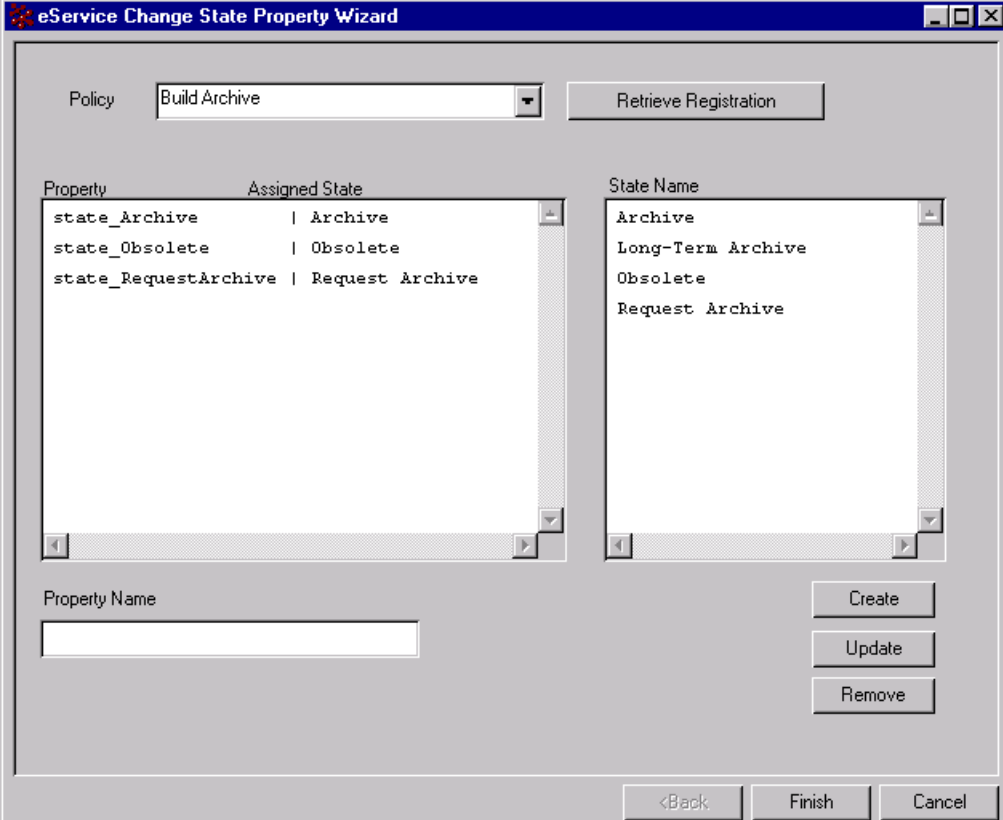
The wizard changes the Assigned State to the selected State Name.



To add a state to a policy

1. Add the state to the policy as you normally would in Business Modeler or MQL.
Now you need to add a symbolic name property for the new state. The remaining steps in the procedure describe how to do this.
2. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
3. Choose **Administration Manager** from the role bar.
4. From the Administration Utilities toolset, click **eService Change State Property**.
5. In the Change State Property wizard, choose the policy that contains the state you want to add using the **Policy** drop-down list.
6. Click **Retrieve Registration**.

The wizard lists all the symbolic name state properties (Property column) added to the policy, including the state to which the property points (the Assigned State). The State Name column shows the state names currently in the policy lifecycle. There should be one more state name than property because there is no property for the state you added in Step 1.



The dialog box is titled "eService Change State Property Wizard". It features a "Policy" dropdown menu set to "Build Archive" and a "Retrieve Registration" button. Below these are two list boxes: "Property" and "Assigned State". The "Property" list contains three items: "state_Archive", "state_Obsolete", and "state_RequestArchive". The "Assigned State" list contains three items: "Archive", "Obsolete", and "Request Archive". To the right of these is a "State Name" list box containing four items: "Archive", "Long-Term Archive", "Obsolete", and "Request Archive". At the bottom left is a "Property Name" text box. At the bottom right are three buttons: "Create", "Update", and "Remove". At the very bottom are three buttons: "<Back", "Finish", and "Cancel".

Property	Assigned State
state_Archive	Archive
state_Obsolete	Obsolete
state_RequestArchive	Request Archive

State Name

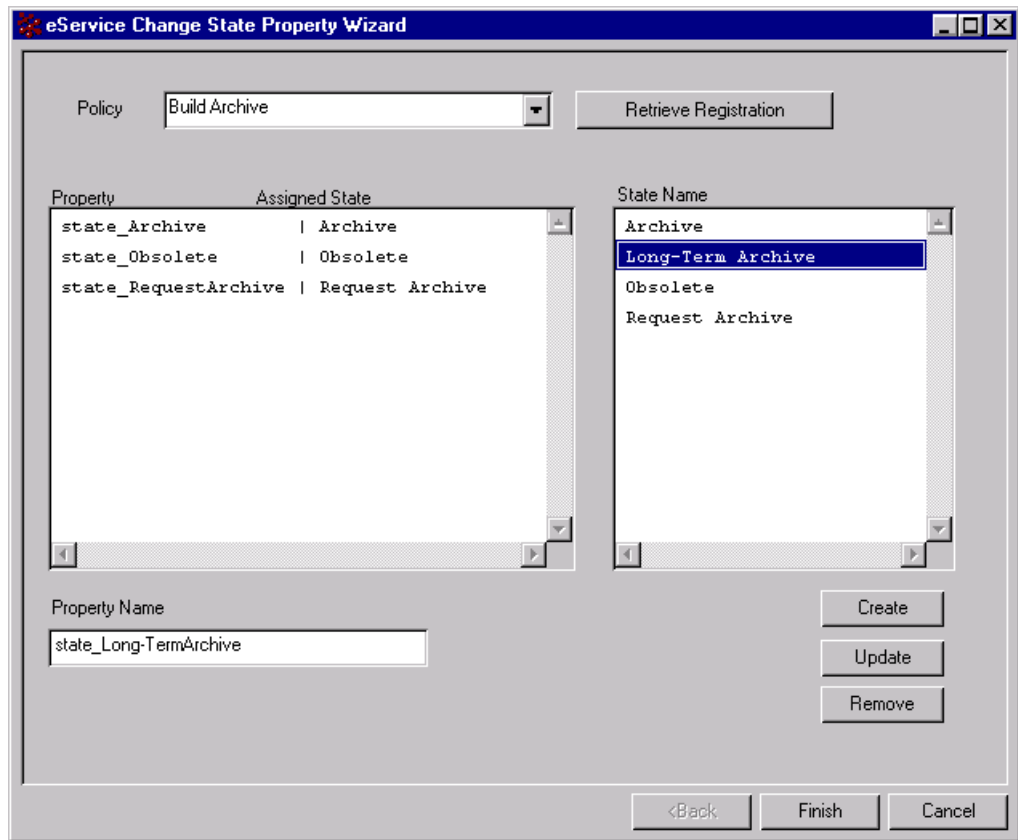
- Archive
- Long-Term Archive
- Obsolete
- Request Archive

Property Name

Create
Update
Remove

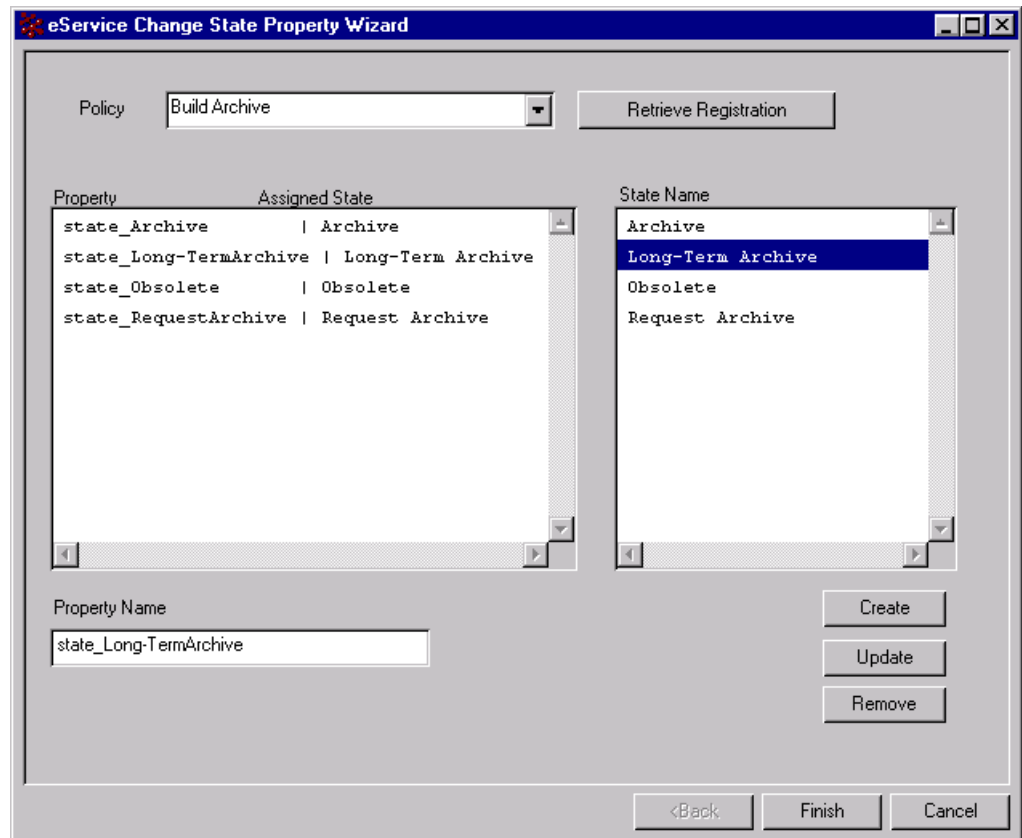
<Back Finish Cancel

7. In the **Property Name** box, enter the symbolic name for the new state using the format "state_[state name with no spaces]". For example, if you added a state called "Final Review", the symbolic name property would be "state_FinalReview".
8. In the **State Name** list, choose the state that you added.



9. Click Create.

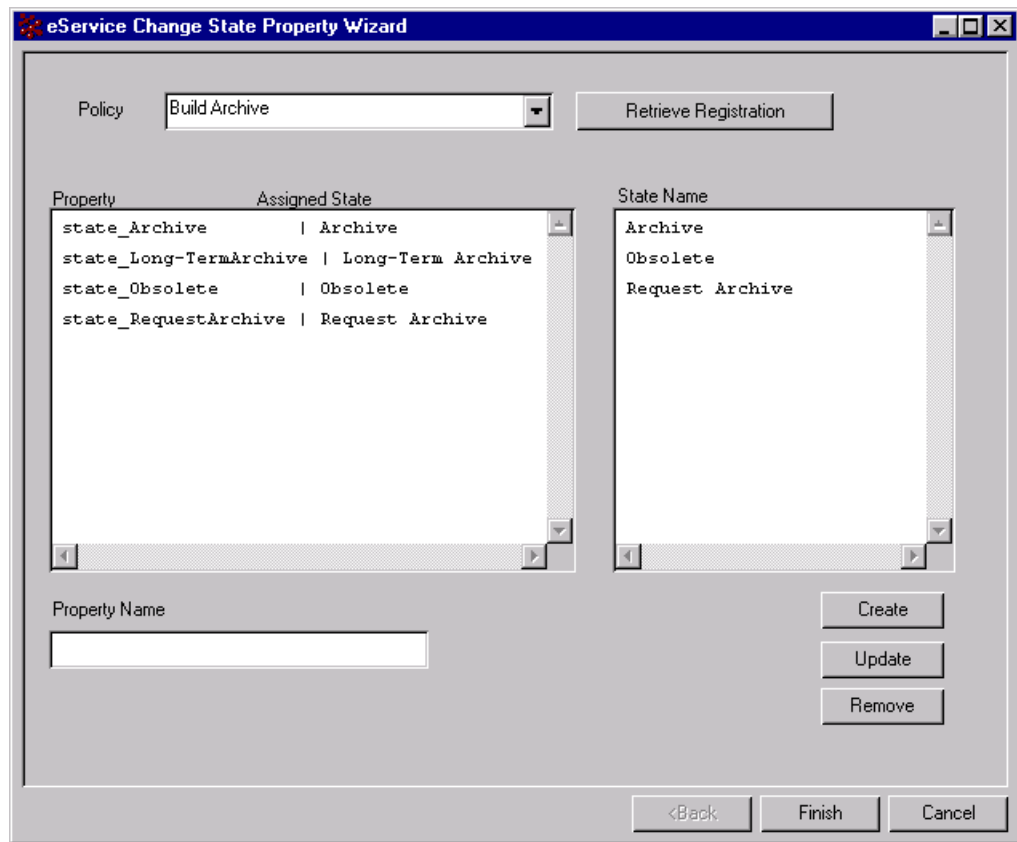
The wizard adds the property and the assigned state.



To delete a state from a policy

1. Delete the state from the policy as you normally would in Business Modeler or MQL.
Now you need to delete the symbolic name property for the state. The remaining steps in the procedure describe how to do this.
2. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
3. Choose **Administration Manager** from the role bar.
4. From the Administration Utilities toolset, click **eService Change State Property**.
5. In the Change State Property wizard, choose the policy that contains the state you want to delete using the **Policy** drop-down list.
6. Click **Retrieve Registration**.

The wizard lists all the symbolic name state properties (Property column) added to the policy, including the state to which the property points (the Assigned State). The State Name column shows the state names currently in the policy lifecycle. There should be one more property than state name because you deleted a state in Step 1.



7. Choose the property for the state that you deleted from the **Property/Assigned State** list.
8. Click **Remove**.

The wizard deletes the property and the assigned state.

Configuring Range Values for Attributes

Attributes can have range values from which users can select a specific value. These ranges can be specified programmatically or by adding them to the attribute object.

Another way to let users choose from a list of values for an attribute is to provide a chooser page, accessed using a Browse (...) button next to the field, from which the user can select or search for items. For instructions on adding popup range helper fields to a configurable form page (emxForm.jsp), see [Field with Popup Range Helper](#) in Chapter 8.

Range Values Specified by Program

Prior to framework 10.0.1.0, the range values for the following attributes were populated using a program. As of version 10.0.1.0, the range values for these attributes are empty. For better performance, the MatrixOne applications that use these attributes let users select values for the attributes using choosers.

- The **Change Board** attribute on the Product Line type. This attribute lists all the child groups of the Change Board group.

- The **Designated User** attribute on document types, such as Data Document, CAD Model, CAD Drawing, Drawing Print, Document, etc. This attribute lists all users who are allowed to access the document.
- The **ECR Evaluator** attribute on the ECR type. This attribute lists all the persons who are assigned to the ECR Evaluator role.
- The **Release Distribution Group** attribute on the ECO type. This attribute lists all the child groups of Distribution Groups.
- The **Responsible Design Engineer** attribute on the Configurable Product, ECR, and ECO types. This attribute lists all the persons who are assigned to the Senior Design Engineer role.
- The **Responsible Manufacturing Engineer** attribute on the ECO type. This attribute lists all the persons who are assigned to the Senior Manufacturing Engineer role.
- The **Responsible Product Manager** attribute on the Configurable Product type. This attribute lists all the persons who are assigned to the Product Manager role.

If you want to use a program to populate the range values for an attribute, you can use the program *eServicecommonWzrdListFromCommand.tcl*. The program accepts any MQL command that generates a delimited list and prepares that list to load either a wizard widget or a program attribute.

Arguments

- Any *MQL command* that produces a pipe (|) delimited list.
- *List* of items to be added in output list.
- The *delimiter*; the default is end-of-line (“\n”).
- A *selection* from the list to be used for default (optional). If this is not specified, the first item in the list will be selected.

Example Configurations

Note the use of {} as argument delimiters, thereby allowing a more MQL-like use of double quotes for named items.

To get a list of types:

```
input = {list type}
```

To get a list of subgroups of the Change Board group and make Unassigned as default:

```
program eServicecommonWzrdListFromCommand.tcl input {print
group $group_ChangeBoards select child dump |} {Unassigned} {}}
```

To get a list of members of the group Change Board 1:

```
input = {print group "group_ChangeBoard1" select person dump |}
{}}
```

To get a list of subtypes:

```
input = {print type "type_ParentType" select derivative dump |}
{}}
```

To get a list of ECO's with current state and an attribute value:

```
input = {temp query bus ECO * * select current
attribute[attribute_CategoryofChange] dump " "}
```

Returns

RPE settings for the list of names and the selected name.

Range Values Specified Manually

Many attributes have pre-defined ranges, which are suggestions to represent the type of information to be stored in them. You will probably need to add and delete some range values to suit your business processes. To see the pre-defined range values, refer to [Attributes](#) in Chapter 3.

To change the range values for an attribute, edit the attribute in Business Modeler. Click the Ranges tab, and add and delete ranges as needed.

Changing Onscreen Text

All onscreen text for the MatrixOne applications is stored in properties files named `emxAPPLICATIONNAMEStringResource.properties`. This includes text called from JSPs, text called from Tcl programs and triggers, and text displayed in email notifications. All properties files are located in `Apps/Framework/VERSION/UserInterface`. The `StringResource.properties` file for the framework is called `emxFrameworkStringResource.properties`. This file contains the text for the pages and features installed with the framework, such as the login and home pages, and IconMail feature.

The `StringResource.properties` files are also used to internationalize applications. Translated versions of the `StringResource` files are also located in the properties directory. The translated versions have the two-letter language code appended to the end of the file name. For example, the French version of the `StringResource` file for the framework is named `emxFrameworkStringResource_FR.properties`. The `emxFrameworkStringResource.properties_*` files contain names for administrative objects and other schema, such as type names, policy state names, and attribute range values. The following instructions apply to making onscreen text changes for the English version of the applications. For information about changing text for internationalized versions, see [Internationalizing the Applications](#) in Chapter 6.

To change onscreen text

1. Open the appropriate string resource properties file in any text editor. For example, to change onscreen English text for Engineering Central, open `emxEngineeringCentralStringResource.properties`. To change framework text, open `emxFrameworkStringResource.properties`. All properties files are located in `Apps/Framework/VERSION/UserInterface`.
2. Using the text search feature of the text editor, search for the text string that you want to change.
3. Edit the text string as needed. Make sure you only edit the value, which is the text that appears after the equals sign (=). Do not edit the ID itself.

For example, to change the column heading labelled “Login Name” within IconMail to “User Name”, you change this line in the `emxFrameworkStringResource.properties` file:

```
emxFramework.IconMail.Common.LoginName = Login Name
```

to:

```
emxFramework.IconMail.Common.LoginName = User Name
```
4. Save the properties file as ASCII.
5. If there is an `_en` version of the properties file, save the file as the `_en` version. The `en` version is used to show English text on a non-English operating system.

Always make backups of properties files that you configure. When you install a new version of the framework or applications, the properties files will be overwritten with updated files.

Configuring the Content of System Notifications and Defining Company-Specific Messages

Many of the applications regularly send notifications to users to inform them of important assignments and changes to objects that they own. Users receive these notifications in their IconMail inbox and, if e-mail is configured for the user, at the e-mail address specified for the user's profile.

The content for these notifications is stored in the `emxFrameworkStringResource.properties` file (and the equivalent file for each language). Each message has a "tag" for its subject and text, which is followed by the content to use when constructing the message. For example, the tags for the "new owner" messages in English are:

```
emxFramework.Beans.DomainObject.NewOwnerSubject=New owner of
'<type>' '<name>'.
emxFramework.Beans.DomainObject.NewOwnerMessage=You are the new
owner of '<type>' '<name>'
```

If some companies need custom messages, you can define these messages for specific companies. Each standard mail message tag can have additional tags created that append the company name, with no whitespace. For example, a "new owner" message subject tag for Acme Company. would be named:

```
emxFramework.Beans.DomainObject.NewOwnerSubject.AcmeCompany
```

When sending the notifications, the system always looks for company-specific messages defined for the current user's company. (The system uses the company connected to the person's business object.) If the system doesn't find company-specific tags for the messages, it uses the standard message.

To configure the content of system notifications and define company-specific notifications

1. Open the file called `Apps/Framework/VERSION/UserInterface/emxFrameworkStringResource_<LANGUAGE>.properties`, where `<LANGUAGE>` is the standard abbreviation for the language. The English versions have no language abbreviation.

If you are revising message text for English, you can open the file in any text editor.

If you are revising message text for any non-English language:

- a) Open the properties file using Microsoft Word. For example, if the text is on a page within Engineering Central and you want to change the German text, open `emxEngineeringCentralStringResource_deNative.properties`.

Word will prompt you to choose the file encoding necessary to load the file.

- b) Accept the default by clicking **OK**.

2. Find the message you want to configure.
3. To configure the content of the standard message, edit the text after the equal sign (=). Do not edit the tag that identifies the message.
4. To define a company-specific version of the message:
 - a) Copy the entire line for the standard message (tag and content) and paste it under the line.
 - b) To the end of the tag, add a period (.) and the company's name, with no white space between words and using the same capitalization and spelling as used in the company's database definition.
 - c) Edit the content by editing the text after the = sign.

5. If you are working with an English file, save the file normally.

If you are working with a non-English file:

- a) Save the file as encoded text.
- b) Click **Yes** to save it in text format, and then choose the same encoding that was used to load the file, which should be the default.
- c) Click **OK**.
- d) Convert the file to ASCII by running the following command:

```
native2ascii -encoding CODE INPUTFILE OUTPUTFILE
```

CODE is the language encoding (shown in the files header).

INPUTFILE is the name of the file you edited.

OUTPUTFILE is the name of the file to be created. The output file will be in Unicode, which means it will contain letters followed by escape characters in place of the native characters.

For example:

```
native2ascii -encoding JIS
emxFrameworkStringResource_ja_Native.properties
emxFrameworkStringResource_ja.properties
```

6. Make sure the saved file is in Apps/Framework/VERSION/UserInterface and make a backup copy of the property files.
7. If you are running a J2EE implementation, remember to run the warutil and deploy the archive file.
8. Restart the app server.

Configuring Automatic Business Object Naming

When a MatrixOne application user performs an action that creates a business object, it's often convenient and sometimes necessary for the system to create a unique name for the object. For example, when a customer places a sales order using the Product Configurator application, the system creates a Sales Order object and generates a name automatically. This name becomes a unique ID that distinguishes the sales order from all other orders.

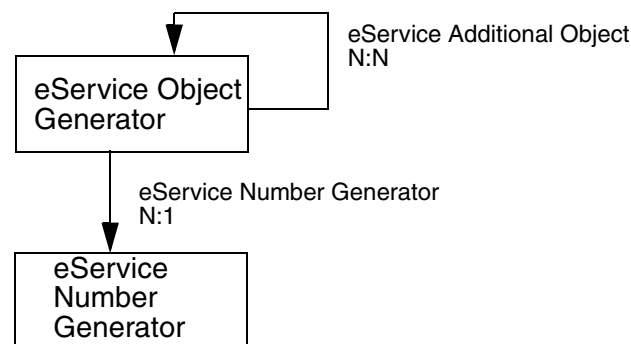
Automatic naming is established and can be configured per object type. This means you can use one naming convention for sales orders, and another convention for product configurations. To find out which object types are named automatically, refer to the user guide that accompanies the MatrixOne Application(s) you are using. The automatic naming conventions that have been established for each application are suggestions only. You will probably want to configure the naming as appropriate for your business processes.

All automatic names contain a number that increments by 1 for each object that is created. For example, the number for the first object created might be "0000100", the number for the second object created would then be "0000101", and so on. By default, each object name contains a prefix that is an abbreviation of the object type name. For example, the prefix for sales orders is "SO", so the first sales order object is named "SO0000100". You can change this prefix and you can also add a suffix to the name.

Another way you can set up automatic naming is so that whenever an object of a particular type (which we'll call the primary type) is created, additional objects of another type are also created. For example, you can make it so that whenever a sales order is created, an order confirmation is also created and linked to the sales order by a specific relationship. You can also set up the numbering sequence for the object names so the numbers are the same for the two objects that are created.

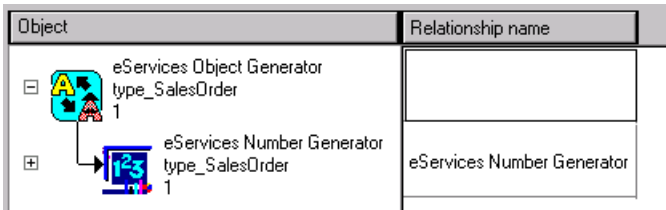
Data Model for the Automatic Naming Process

This diagram shows the data model for the automatic naming process. For help reading the model, see [How to Read Data Models](#) in Chapter 2.

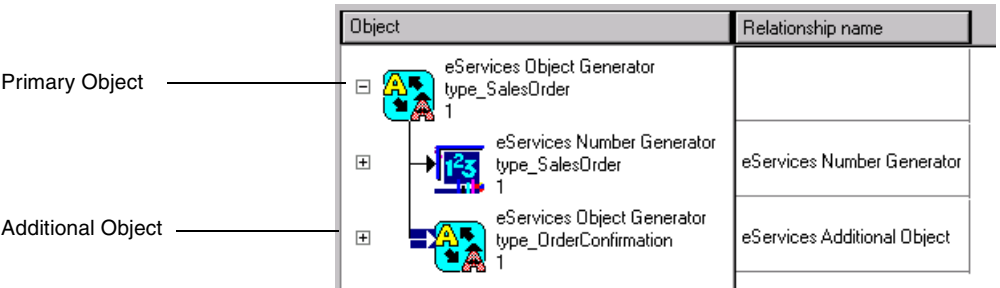


Every object type that needs automatic naming must have an eService Object Generator object. Every type that needs a unique numbering sequence requires an eService Number Generator object that is connected to the eService Object Generator. The Object and Number Generator business object types are governed by the eService Object Generator policy. The business objects are created in the eServiceAdministration vault (symbolic

name vault_eServiceAdministration). For example, the automatic naming business objects for sales orders looks like this:



Suppose an Order Confirmation object should be created every time a Sales Order object is created. In such a case, the eService Object Generator for order confirmations must be connected to the eService Object Generator for sales orders using the eService Additional Object relationship. Since the Object Generator objects are connected to the same Number Generator object, the numbering sequence is the same. The first sales order created would be called “SO0000100” and the first order confirmation would be called “OC0000100”.



We'll refer to the type whose Object Generator is connected directly to the Number Generator and on the From end of the Additional Object relationship, Sales Order in the case described above, as the primary type. Types on the To end of the Additional Object relationship are additional types—Order Confirmation in the above example.

The eService Object Generator uses these attributes:

Specifies the policy that governs the type of object being created. The object generator program uses this policy only when the value for the ObjectPolicy parameter that is passed to the object generator program is null and more than one policy can govern the object type. The policy name must be specified using the symbolic name.

Number of seconds to wait before the eService Object Generator object should be rechecked when it is locked.

The vault in which the business object is created if the current vault is ADMINISTRATION.

Number of times the Object Generator program has to create a new object. If the limit is reached and the object is not created, the system presents an error message.

Time period in milliseconds before the eService Object Generator should be accessed if it is locked.

Adds a suffix or prefix to the name that is generated automatically for objects. The characters you specify for the prefix are prepended to the unique number that is generated for the name.

The eService Number Generator uses these attributes:

The number assigned for the name of the next object that is created using an eService Object Generator. Every time a new object is created automatically, the system increments the Next Number value by adding one to the current value.

The eService Connect Relation relationship uses the following attribute:

The relationship the Object Generator program should use when connecting the newly-created primary object to the newly-created additional object. Use the symbolic name for the relationship.

Object Generator Program

The program that generates the automatic names is called eServicecommonObjectGenerator.tcl.

Parameters

The program accepts the following parameters. These parameters are passed by the program that creates the object, which may be a Tcl program or a JavaServer Page (JSP). See [Calling the Object Generator from JSP or Tcl](#).

ObjectGeneratorName—Name of the eService Object Generator that contains the attributes for the object type. For example, if the object generator is for Sales Orders, the object generator name would be “type_SalesOrder”.

ObjectGeneratorRevision—Revision of the eService Object Generator.

ObjectPolicy—Specifies the policy for the object being created. This is an optional parameter. If no policy is specified and more than one policy can govern the object type, the policy specified for the eService Safety Policy attribute on the eService Object Generator is used.

CreateAdditional—Specifies whether to create additional objects. This is a flag with the values Null and Additional. Null means no additional objects are to be created. Default is Null.

Returns

A list of lists. Each item in the list holds the ID, type, name, and revision of the object created. If there is an error, it returns a list with the first item as null and the second item contains the error message.

How the Automatic Naming Process Works

When a user performs a task that calls for an object to be created using the automatic naming process, the eServicecommonObjectGenerator.tcl program performs these steps:

1. Makes sure the eService Object Generator object for the object type exists. If the object doesn't exist, the program returns with a null and an error message.
2. Gets the attribute values for the eService Object Generator for the type.
3. To avoid creating objects with the same name when more than one user attempts to create a object at the same time, locks the Number Generator object for the object type. If the object is already locked, the program waits for a specific time period, which is the value of the attribute eService Retry Delay. The program retries for eService Retry Count times. If the program is unable to access the Number Generator, it returns with a null and an error message.
4. After locking the Number Generator object, gets the next number to be generated from the eService Next Number attribute on the eService Number Generator object for the object type.
5. Updates the Next Number attribute with the next value to be assigned, which is the current value of the attribute + 1.
6. Unlocks the Number Generator so it's now accessible when another user needs to create an object.
7. Generates the name of the object to be created as PrefixNextNumberSuffix.
8. Makes sure the name is unique. If the object is unique, the program creates the object. If the name is not unique, the program tries to get a unique number until it gets one or until the eService Processing Time Limit (an attribute on the Object Generator) gets

exhausted, whichever comes first. If the eService Processing Time Limit gets exhausted, then the program returns with a null and an error message saying that the Number Generator object is busy.

9. Determines if there are additional objects to be created using the value of the input parameter CreateAdditional. If no additional objects are to be created, the process is complete.
10. If additional objects are to be created, the program expands on the primary object for the relationship eService Additional Object and gets all the additional objects to be created. For each of these additional objects, the program makes sure the name is unique. If a name already exists, the program tries to get a new number. This ensures that all names are unique.

The program then connects the newly-created primary object with the newly-created additional object(s). The relationships by which these objects are to be connected are the values of the attribute eService Connect Relation. The program assumes the From type of the relationship is always the primary object.

Configuring Automatic Naming

This section contains instructions for making the most common changes for automatic naming. For help making other changes, such as changing the number of times the system tries to get the next number before failing, see the list of attributes and program parameters that are listed in the sections [Data Model for the Automatic Naming Process](#) and [Object Generator Program](#).

To change the name that is automatically generated for an object type

1. In Matrix Navigator, find the eService Object Generator for the type.
2. Show attributes for the object.

Attributes - eService Object Generator type_SalesOrder

Description: Autonomer for Sales Orders

eService Safety Policy: policy_SalesOrder

eService Processing Time Limit: 60

eService Safety Vault: vault_eServiceProduction

eService Retry Count: 5

eService Retry Delay: 1000

eService Name Suffix:

eService Name Prefix: SO

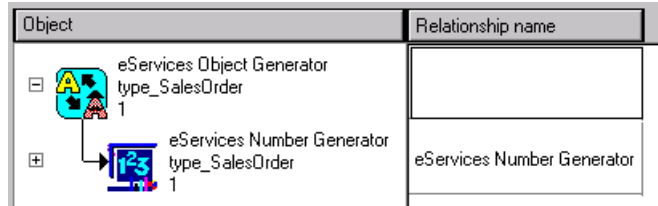
Modify Cancel

3. To add characters after the automatically-generated number, enter the characters in the **eService Name Suffix** box. To add characters before the automatically-generated number, enter the characters in the **eService Name Prefix** box.
4. Click **Modify**.

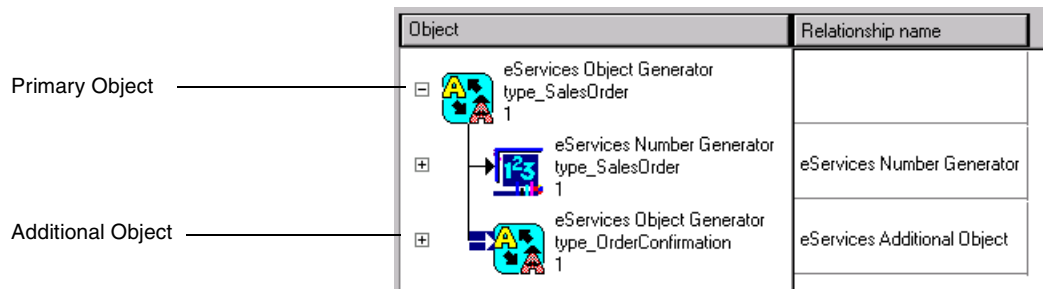
Use the following procedure to have the system create an object (additional object) whenever another object type (primary object) is created and have the two objects use the same numbering sequence.

To create an additional object when the primary object is created

1. In Matrix Navigator, make sure there is an eService Object Generator for the primary object and that it is connected to an eService Number Generator.



2. Make sure there is an eService Object Generator for the additional object. Create the object if needed.
3. Connect the Object Generator for the primary object to the Object Generator for the additional object using the eService Additional Object relationship.



When you make the connection, an Attribute dialog box appears with a box for the **eService Connect Relation** attribute.

4. Enter the name of the relationship that should connect the primary objects to the additional objects. Use the symbolic name for the relationship.



Remember to register the symbolic name for any custom administrative objects (objects that are not supplied with the framework). If you use an unregistered symbolic name as an input for a program, the program will not work. See [Registering Your Own Administrative Objects](#).

5. Click **Modify**.
6. Modify the Tcl or JSP program that creates the primary object type, as described below. It should pass to the eServicecommonObjectGenerator.tcl program the value "Additional" for the CreateAdditional parameter.

JSP

To use the object generator in a JSP program, you can use the `autoName` function that is defined in `eServiceUtil.jsp`. This file is in the “ematrix” directory. To use the `autoName` function, make sure you include the `eServiceUtil.jsp` in your JSP using an include directive: `<@include file="eServiceUtil.jsp">`.

The `eServiceUtil.jsp` defines two function signatures for overloaded `autoName` function. The first accepts a revision for the created object and the second does not accept a revision.

- `static public String autoName(HttpSession session, String type, String revision, String policy)` throws `MatrixException`

To use this `autoName` function to create an object of type `XYZ` and revision 1, you would include the below in a JSP page. The example assumes type `XYZ` is governed by policy `ABC`.

```
String XYZObjId = autoName(context, session, "type_XYZ", "1",  
"policy_ABC");
```

- `static public String autoName(HttpSession session, String type, String policy)` throws `MatrixException`

This function signature does not accept a revision. To use it to create an object of type `XYZ` without specifying a revision, you would include the below in a JSP page. The example assumes type `XYZ` is governed by policy `ABC`.

```
String XYZObjId = autoName(context, session, "type_XYZ",  
"policy_ABC");
```

Remember to register the symbolic name for any custom administrative objects (objects that are not supplied with the framework). If you use an unregistered symbolic name as an input for a program, the program will not work. See [Registering Your Own Administrative Objects](#).

If you want to create an additional object along with the primary object, you’ll need to pass the value “Additional” to the ObjectGenerator for the `CreateAdditional` parameter. To do this, you would define a third function signature for `autoName` and add the `Additional` parameter as the last parameter.

Tcl

To use the object generator within a Tcl program, call `eServicecommonObjectGenerator.tcl` using `utLoad`. Pass the `ObjectGeneratorName`, `ObjectGeneratorRevision`, `ObjectPolicy`, and `CreateAdditional` (optional) parameters to the `eServiceNumberGenerator` procedure.

Remember to register the symbolic name for any custom administrative objects (objects that are not supplied with the framework). If you use an unregistered symbolic name as an input for a program, the program will not work. See [Registering Your Own Administrative Objects](#).

The following example shows how a Tcl program would use the object generator to create objects of type `XYZ`, revision 1, and policy `ABC`.

```
eval [utLoad eServicecommonObjectGenerator.tcl]  
set sNewInfo [eServiceNumberGenerator "type_XYZ" "1"  
"policy_ABC" ""]
```

```

set sNewInfoErrorFlag [lindex $sNewInfo 0]
if {[string compare $sNewInfoErrorFlag "" ] == 0 } {
set mqlret 1
set outstr [lindex $sNewInfo 1]
} else {
set sNewObjectId [string trim [lindex [lindex $sNewInfo 0] 0]]
set sNewType [string trim [lindex [lindex $sNewInfo 0] 1]]
set sNewName [string trim [lindex [lindex $sNewInfo 0] 2]]
set sNewRev [string trim [lindex [lindex $sNewInfo 0] 3]]
}

```

Configuring Automatic Business Rules (Triggers)

Many business rules within an application are automated, especially rules within a business object's lifecycle. For example, when a part is promoted from Preliminary to Review, the system ensures that all the attached specifications are in the Review state and that a Find number other than the default has been specified. You can configure these automated processes by:

- Changing the argument inputs for existing rules. For example, instead of checking on the Find number, you could have the system check on a different attribute for parts, such as the classification or effectivity date.
- Adding new rules to check. For example, you could have the system also route the part to a specific role when a part is promoted.

This section describes how the framework and applications automate rules and how you can configure that automation.

How Rules are Automated

The MatrixOne applications automate business rules using trigger programs that run when a particular trigger event occurs. All trigger programs must be represented by program objects in Matrix and the programs can be either Tcl or Java Program Objects (JPOs). There are three kinds of trigger programs: Check, Override, and Action. For a description of each kind of trigger program and general information about triggers, see the Working with Triggers chapter of the *Matrix Business Modeler Guide* or the *MQL Guide*.

The MatrixOne applications handle triggers in a different way than in standard Matrix. This different approach lets you specify multiple rules for a specific trigger event. You can run multiple trigger programs of the same kind for an event (for example, run three check programs when a part is created). It also makes it easier to enter input for the trigger program arguments.

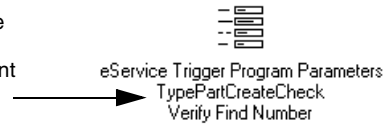
Make sure the `MX_TRIGGER_RECURSION_DETECTION` parameter is set to Signature in the initialization files (`bos.ini`, `ematrix.ini`).

Trigger Management Components

The trigger management system contains two main components. The first component, which is installed with the framework, is a program called Trigger Manager (`emxTriggerManager`). The Trigger Manager accepts input for one argument only. This argument is the name of *business objects* of a type called eService Trigger Program Parameters, the second component of the trigger management system. Trigger Program Parameters objects represent specific trigger programs and are installed with the MatrixOne applications.

Input for the Trigger Manager is . . .

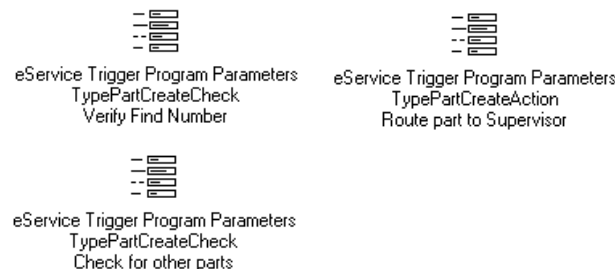
. . . the names of the eService Trigger Program Parameters business objects that represent trigger programs to run.



eService Trigger Program Parameters objects have attributes that specify the:

- trigger program to run
- inputs to be passed for the trigger program's arguments
- order in which the trigger program should be executed, if other trigger programs of the same kind run for the trigger event
- if the trigger program is a Java program, the name of the method inside the JPO to run

One eService Trigger Program Parameters object must exist for each trigger program that needs to be run for a trigger event. For example, if a trigger event has two check programs and one action program, there must be three Trigger Program Parameters objects.

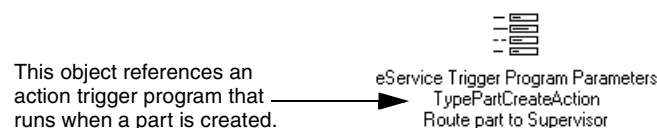


Trigger Program Parameters Name and Revision

To ensure the name of each Trigger Program Parameters object is unique and to help you easily identify the different objects, we recommend using this naming convention:

[ADMIN OBJECT TYPE][ADMIN OBJECT NAME][TRIGGER EVENT][TYPE OF TRIGGER]

Using this convention, the name of a Trigger Program Parameters object indicates the trigger the object is associated with. For example, a Trigger Program Parameters object called "TypePartCreateAction" indicates that the object references an action trigger program that runs when a part is created.



Policy triggers require one more piece of information for the Trigger Program Parameter object name:

[admin object type][admin object name][state name][trigger event][type of trigger]

For example, this Trigger Program Parameter is associated with an action trigger program that runs when an object is demoted from the Active state of a Person policy.



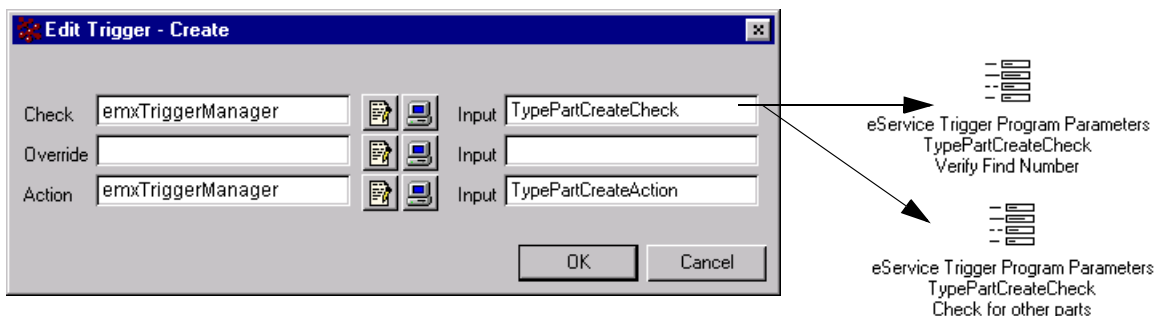
Here are some examples of names for Trigger Program Parameters for each administrative object type:

- attributes
AttributeAddressModifyCheck
- type
TypeCompanyCheckinAction
TypeECRChangeOwnerCheck
- relationship
RelationshipDocumentsFreezeAction
RelationshipCapabilityModifyAttributeAction
- policy
PolicyProductReleaseApproveCheck
Policy ProductReleaseOverrideAction

When a trigger event occurs, the Trigger Manager queries for all Trigger Program Parameter objects that have the same name as the input. The query includes all revisions. If more than one trigger program should execute for the same trigger event and the same kind of trigger (Check, Override, Action), the names of the Trigger Program Parameter objects must be the same and the revision must be different. The revision can be a standard number or letter, or it can be a description of the trigger program.

Consider the following example. If two check programs should run when a part is created, there must be two Trigger Program Parameters objects with the name TypePartCreateCheck but with different revisions. Each object has attributes that reference a specific trigger program and set of argument inputs. The objects also have an attribute that specifies which program should run first.

The Trigger Manager finds all Trigger Program Parameters objects whose name matches the input, regardless of revision.



Trigger Manager Processing

When a trigger event calls the emxTriggerManager program, the program performs these steps:

1. Make a copy of all of the Matrix environment variables.
2. Set a new environment variable called ORIGINAL_INVOCATION with the value from the original INVOCATION variable.

The reason for this is when a Tcl trigger program is executed through the JPO Trigger Manager, this value gets set to program and we do not want to lose the original INVOCATION value. This means that triggers that rely on the INVOCATION environment variable for program execution should look for the ORIGINAL_INVOCATION environment variable.
3. Query for business objects that meet this criteria:

Name—Input specified for the emxTriggerManager program. This is the name of the eService Trigger Program Parameters object that represents the trigger program to be run.

Type—eService Trigger Program Parameters

Revision—* (all)

Vault—eService Administration if there is such a vault or * (all) if eService Administration doesn't exist.
4. Discard all returned objects that are in the Inactive state.
5. Sort the objects in ascending order according to the eService Sequence Number attribute.
6. Loop through each of the objects found and determine if they are Tcl or Java programs.
7. Execute the programs in the specified sequence and include the inputs. Trigger Manager waits for each program to complete successfully before calling the next program. If trigger is a Java program, get the name of the method inside the JPO and invoke it. The method is specified in the eService Method Name attribute for the eService Trigger Program Parameters object. If no method is specified, default to the main (mxMain).

Or:

If the trigger is a Tcl program, execute the program using the MQL command class.
8. The program returns zero for success and a nonzero for a failure.

If, after invoking a JPO trigger, the Trigger Manager gets a return value of 1 and then checks the INVOCATION type. If the INVOCATION is an action (Action Trigger), it throws an exception. For any other type of INVOCATION, the Trigger Manager JPO returns 1 to the Matrix invoking environment to indicate failure. Otherwise, a zero is returned to indicate success.

For each trigger that is completed without error, Trigger Manager resets all of the Matrix environment variables back to their original values, until all trigger(s) are executed successfully.

If any program returns failure, the Trigger Manager stops execution and returns failure. In the case of a check trigger, failure prevents the event from occurring. In the case of an action trigger, the programs that are called do not return success or failure because actions do not wait for a return code. If a program that is called as an action

fails as a result of an error in an MQL command, the event that invoked them will, as expected, be aborted. However, if you want to trap an error in an action program by means other than MQL errors, you can use something like the following:

```
If {sString == ""} {  
    mql abort transaction  
    return  
}
```

Reasons for Running Triggers in a Specific Sequence

The reasons for having one trigger run before or after another trigger are as follows:

1. Triggers that are expected to fail more often should be run earlier than triggers that are not expected to fail as often. This reduces the amount of time a user waits for triggers that will usually pass because they will only be run after all of the other more likely-to-fail triggers have passed.
2. Triggers that are fast should be run before triggers that are slow. It's better to get through fast triggers early so that if the event needs to be attempted more than once, the user will not spend as much time waiting for a slow trigger to run more times than necessary.
3. Triggers that perform checks which other triggers (within the same event) depend on must be run first so that those preconditions are satisfied for the later "dependant" triggers. This enables subsequent triggers to reduce the amount of redundant checking that occurs since they know their preconditions have been met.

Items #1 and #2 above are somewhat subjective and require decisions based upon domain knowledge of the process and data to make the proper trade-offs.

Item #3 indicates that triggers installed with the framework and applications should not have their order changed relative to each other unless you know there are no interdependency checks. To determine if there are interdependencies, see the Administrator's Guide for the application.

Sources of Trigger Programs

Trigger programs can come from four main sources:

- The trigger programs used by many applications, called utility trigger programs, are installed with the framework and are described in [Utility Trigger Programs](#) in Appendix A.
- Programs specific to a MatrixOne Application are installed with the application. For information about trigger programs that are specific to an application, refer to the user guide that accompanies the MatrixOne Application.
- The Subscription Services Application Library contains many trigger programs.
- You can create your own custom trigger programs.

Do not make changes to the programming for trigger programs that are installed with the framework or MatrixOne applications. You should handle any adjustments using the argument inputs passed by the eService Trigger Program Parameters object and described in [Modifying Inputs and Execution Sequence for Trigger Programs](#).

Also, don't change the file names of the trigger programs. The file names for all MatrixOne applications programs are prefixed with "eService[name of installed directory]". Do not change this prefix and don't create custom programs using the

eService prefix. If you need to modify the name of a MatrixOne applications program, clone the program and change the name of the cloned program.

Adding a Trigger for a Trigger Event

This section describes how to have a trigger program run when a particular trigger event occurs. The instructions assume the trigger program has already been installed (or created if you have a custom program). Make sure you follow the instructions carefully and avoid making these common mistakes:

- Creating the eService Trigger Program Parameters object in a vault other than eService Administration. The objects must be stored in eService Administration.
- Failing to promote the eService Trigger Program Parameters object to Active after creating it. The Trigger Manager program only looks for objects in the Active state.
- Not matching the name of the eService Trigger Program Parameters object to the name entered in the Input box for the trigger in Business Modeler. The names must match exactly.

These procedures describe how to add triggers using Matrix Navigator and Business Modeler, but you can use MQL instead.

To add a trigger for a trigger event

1. In Matrix Navigator, create an eService Trigger Program Parameters object that represents the trigger program.
 - a) Select **Object>New>Original**.
 - b) Complete the Original dialog box.

The screenshot shows a dialog box titled "Original" with a close button (X) in the top right corner. It contains five input fields, each with a dropdown arrow on the right: "Type" (set to "eService Trigger Program Parameters"), "Name" (set to "PolicyConfigurableProductStateReviewPromoteCheck"), "Revision" (set to "Check for Parts in Release State"), "Vault" (set to "eService Administration"), and "Policy" (set to "eService Trigger Program Policy"). At the bottom right are "Create" and "Cancel" buttons.

Type—eService Trigger Program Parameters.

Name—The name should indicate the trigger the object is associated with by including the administrative type (policy, type, attribute), the administrative object name (Part, Originator, etc.), the trigger event (Create, Promote, etc.), and the kind of trigger program (Check, Override, Action). For policies, include the state name also. For more information, see [Trigger Program Parameters Name and Revision](#).

Revision—Any string, such as 1 or A or a. You may want to include a description of the trigger program associated with the object.

Vault—The vault must be eService Administration.

Policy—eService Trigger Program Policy.

- c) Click **Create**.

2. Enter attributes for the new object.

Attribute	Value
eService Target States	
eService Constructor Arguments	
eService Method Name	syncPersonAdminObject
eService Program Argument 10	
eService Program Argument 11	
eService Program Argument 12	
eService Program Argument 13	
eService Program Argument 14	
eService Program Argument 15	
eService Program Argument Desc 10	
eService Program Argument Desc 11	
eService Program Argument Desc 12	
eService Program Argument Desc 13	
eService Program Argument Desc 14	
eService Program Argument Desc 15	
eService Program Name	emxPerson
eService Sequence Number	1
eService Program Argument 1	\$(EVENT)

eService Program Name—Enter the name of the program object that should be executed when the trigger event occurs.

eService Sequence Number—Enter a number that represents the order in which the program should run. For example, if three check programs execute when the trigger event occurs and this program should run last, enter 3. If it doesn't matter which order the programs run in, you must still enter a numeric value other than 0. For more information, see [Reasons for Running Triggers in a Specific Sequence](#).

eService Program Argument n—Enter a value for each argument the object should pass to the trigger program, making sure the argument inputs are in the correct order. Also make sure you use the symbolic name for any administrative objects that you specify as an input. For example, if you need to specify a relationship for an argument input, use the symbolic name for the relationship and not the standard name (relationship_GBOM instead of GBOM). For a list of accepted argument inputs for a trigger program and the correct order for the arguments, refer to [Utility Trigger Programs](#) in Appendix A and to the guide that accompanies your eServices application.

Remember to register the symbolic name for any custom administrative objects (objects that are not supplied with the framework). If you use an unregistered symbolic name as an input for a program, the program will not work. See [Registering Your Own Administrative Objects](#) and [Configuring State Names](#).

eService Program Argument n Description—Enter a description for what each argument does. This description can also include other details, such as the default value for the argument.

eService Method Name—The name of the method to invoke in the JPO. Specifying a method is important if you need to use the same JPO with multiple methods in many different triggers. If no value is entered, the Trigger Manager uses the main (mxMain) method of the JPO.

eService Target States—Only used for promote and demote triggers. Stores a comma separated list of the symbolic names of states. The Trigger Manager fires the trigger program only when promoting or demoting to one of the listed states. If the target state is not in the list, the trigger does not fire. If the attribute is not populated, the Trigger Manager fires the program regardless of the target state.

This is useful for policies that allow branching to multiple states but the trigger program shouldn't run for all states. For example, suppose a policy allows an object to be promoted from State1 to State2A or State2B but the trigger should only run when the object is promoted to State2A. This attribute should contain state_State2A.

3. If this is the first trigger program of this kind for the event (the Trigger Manager program and input hasn't been specified for this kind of trigger for this event), proceed to step 4.

Or:

If there is already a trigger program of this kind for the event:

–Make sure the sequence numbers for the existing trigger programs are correct. If the program you added should run before the existing programs, the sequences for the existing programs will probably have to change. For instructions on changing the sequence number for a trigger program, see [Modifying Inputs and Execution Sequence for Trigger Programs](#).

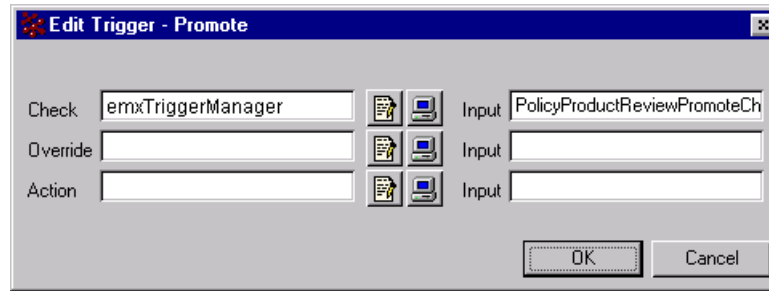
–The name of the Trigger Program Parameter should already have been specified as the input for the Trigger Manager. Therefore, the procedure is complete and you can skip the remaining steps.

4. Promote the object to the Active state.
5. In Matrix Business Modeler, find the administrative object for which you want to add the trigger.

For example, to have a program run when a part is promoted to the Release state, find the policy that governs the part. To have a program run when a file is checked into an object, find that object type.

6. Double-click the administrative object to edit it.
7. Select the **Trigger** tab. (For policies, double-click the appropriate state and then select the Trigger tab.)
8. In the box for the type of trigger program that you want to add (Check, Override, Action), enter **emxTriggerManager**.
9. In the **Input** box to the right of the program name, enter the name of the eService Trigger Program Parameters object that represents the trigger program you want to run. Make sure the name is *exactly* the same as the Trigger Program Parameters object.

If there are eService Trigger Program Parameters objects with different names that need to be called—for example, TypeAllCreateAction and TypePartCreateAction—you can enter both names separated by a space.



Modifying Inputs and Execution Sequence for Trigger Programs

You can change the argument inputs that are passed to a trigger program and the sequence in which the program is executed by modifying the attributes for the program's Trigger Program Parameter object.

These procedures describe how to add triggers using Matrix and Business Modeler, but you can use MQL instead.

To change the argument inputs or execution sequence for a trigger program

1. In Matrix Navigator, find the eService Trigger Program Parameters object that represents the trigger program.
2. Access the object's attributes.
3. Modify the eService Program Argument attribute that contains the input you want to change.

Make sure you use the symbolic name for any administrative objects that you specify as an input. For example, if you need to specify a relationship for an argument input, use the symbolic name for the relationship and not the standard name (relationship_GBOM instead of GBOM). For a list of accepted argument inputs for a trigger program and the correct order for the arguments, refer to [Utility Trigger Programs](#) in Appendix A and to the guide that accompanies your MatrixOne Application.

4. Change the number for the **eService Sequence Number** attribute so it represents the order in which the program should run. For example, if three action programs execute when the trigger event occurs and this program should run last, enter 3. If it doesn't matter which order the programs run in, you must still enter a numeric value other than 0.

Activating and Deactivating Triggers

Since the Trigger Manager program only uses eService Trigger Program Parameters objects that are in the Active state (see [Trigger Manager Processing](#)), you can deactivate a trigger by demoting it to the Inactive state. To re-activate the trigger, promote it back to Active.

Setting the Preferred File Store for a Company

Each top-level company can have a preferred file store set for it and all its business units using the File Store Symbolic Name attribute on the company's business object. This attribute should only be set by host company administrators.

The File Store Symbolic Name attribute is specified for an entire Company. When users log into the system, the system finds the Organization connected via the Employee relationship. If this Organization is not a Company (it's a Business Unit), the system navigates up the Organization relationship structure until it finds the top-level Company business object. If the File Store Symbolic Name attribute has a value, it overrides the value determined by the business object's governing policy.

Use System or MQL to create a store for a company. Register the store and assign it a symbolic name using the eService Change Administration Property wizard as described in [Registering Your Own Administrative Objects](#). For more information on file stores, see the *Matrix System Manager Guide*.

Setting Up 2D and 3D Viewers

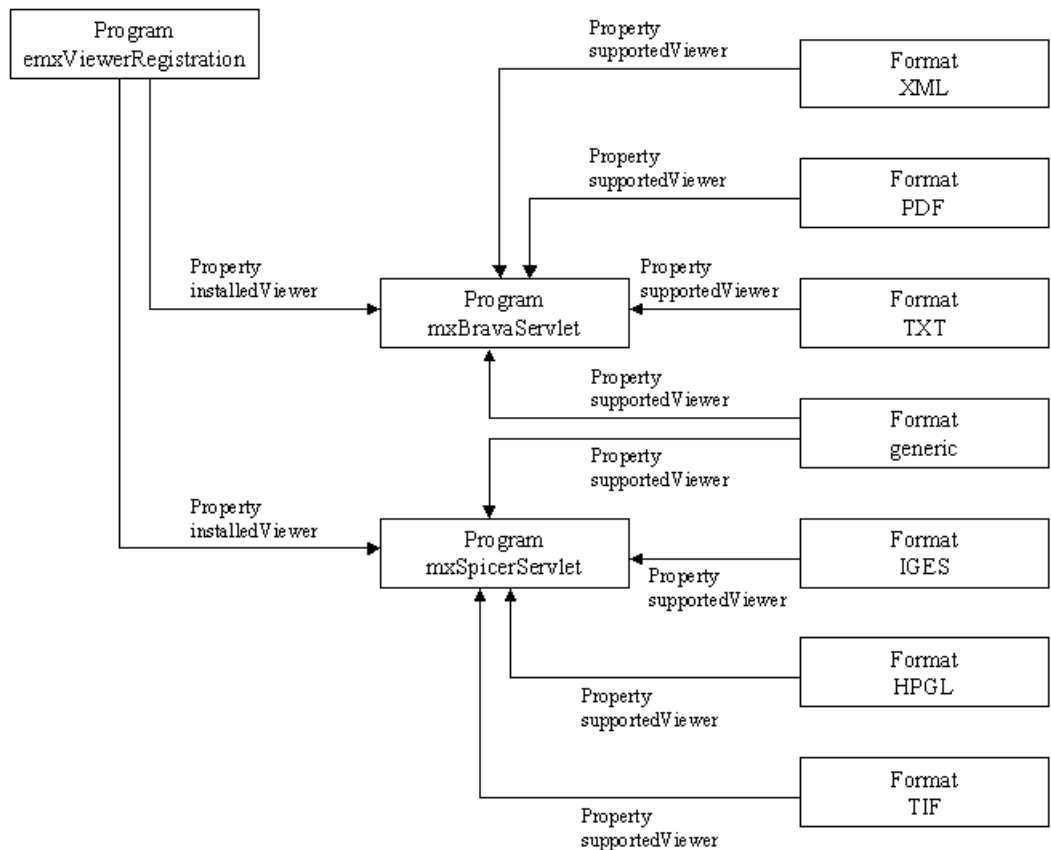
The framework gives you the tools needed to let users launch 2D/3D viewer applications, such as Brava, Spicer, and eVis, from any application. To integrate with a viewer, follow these steps:

1. After installing the framework, install the server and client software for the viewer and the MatrixOne servlet that provides the integration. This may include configuring the `framework.properties` file. For instructions, see the installation instructions that accompany the integration package.
2. Make sure all formats that you want to associate with the viewer are added to Matrix using Business Modeler or MQL.
3. Register the viewer with the framework as described in [Registering Viewers](#).
4. Assign the formats for the viewer as described in [Assigning Formats to Viewers](#).
5. Specify the file extensions that should be opened in viewers as described in [Specify File Extensions for Viewers](#).
6. Define the formats for which users can choose a preferred viewer. For example, if you have more than one viewer registered for a format, you may want to let users choose the viewer they prefer. See [Specify Formats for which a Preferred Viewer Must Be Chosen](#).
7. Make sure everyone who needs to use the viewer has the client software installed locally. Some applications provide the ability to add a Download option to user's Profile page so they can download the e-Vis viewer application and install it. For instructions on configuring this for a particular application, see the application's Administrator's Guide.

How the System Registers Viewers

To launch a viewer from within an application, the viewer must be registered and the formats that the viewer can open must be assigned to the viewer. This section describes the connections and objects that must exist in the Matrix database to register a viewer and assign formats. The framework comes with a wizard that creates these objects and connections for you. For information, see [Registering Viewers](#).

The following graphic shows the database objects and connections needed to register a Brava viewer and a Spicer viewer and assign four formats to each. The graphic is described below.



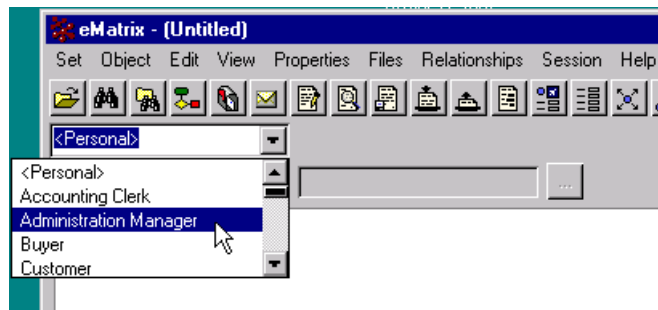
The program object `emxViewerRegistration` is a placeholder for registering viewers. To register a viewer, there must be program object with the same name as the servlet used to invoke the viewer and this program must be connected to `emxViewerRegistration` with the `installedViewer` property. Formats that are compatible with the viewer must be defined as format objects and connected to the viewer program object with the `supportedViewer` property. The program objects that represent the viewers also have a property called `viewerTip` that stores the text for a ToolTip for the viewer.

Registering Viewers

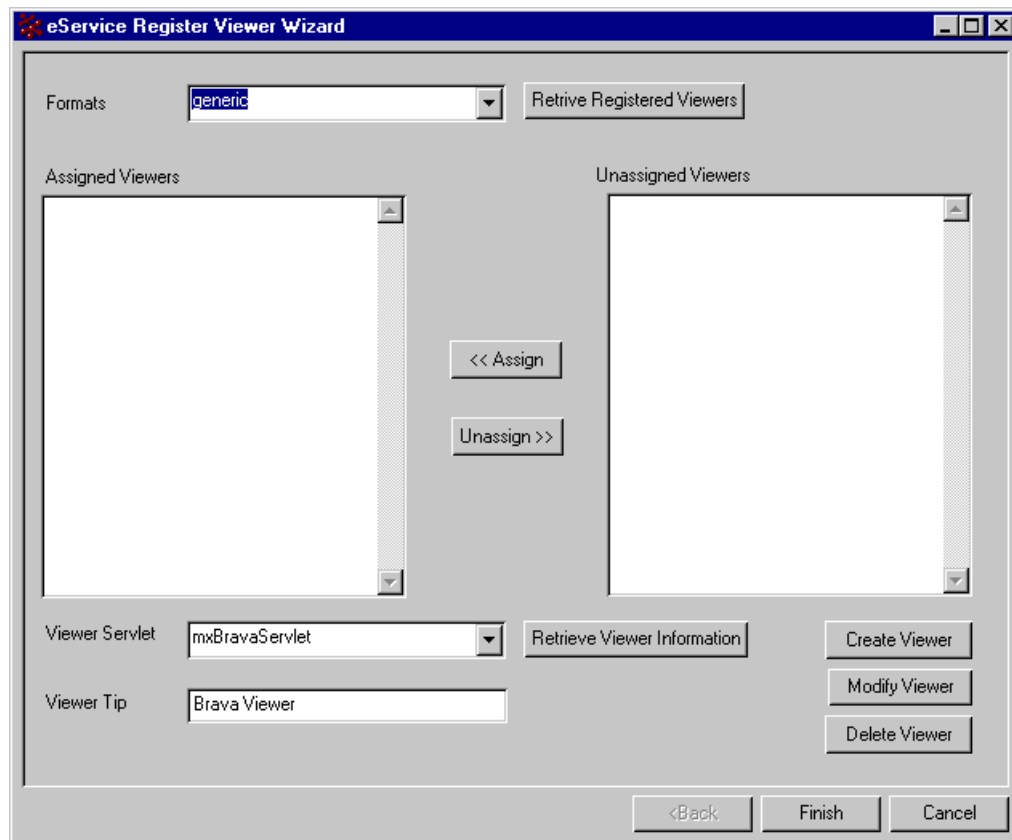
Before users can use a viewer from a MatrixOne Application, you need to register the viewer and assign the formats the viewer is compatible with.

To register a viewer

1. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
2. Choose **Administration Manager** from the role bar.



3. From the Administration Utilities toolset, click **eService Register Viewer**.
The eService Register Viewer wizard opens.



4. In the **Viewer Servlet** box, enter the name of the servlet that invokes the viewer.
5. In **Viewer Tip**, enter the text you want for the viewer's ToolTip. This tip will appear in the MatrixOne applications when a user moves the mouse pointer over the icon that represents the viewer. The system also uses this tip to populate the preference list on the person profile pages for formats that are set up for preference selection (see [Specify Formats for which a Preferred Viewer Must Be Chosen](#)).
6. Click **Create Viewer**.

The viewer is listed in the Unassigned Viewers list. A program object is created with the name you entered for the viewer servlet. The tip text is assigned as a value for the property viewerTip. The program object is connected to the emxViewerRegistration program with the property installedViewer. Now you need to assign formats for the viewer, as described in [Assigning Formats to Viewers](#).

To modify the tip for a viewer or delete the registration for a viewer

1. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
2. Choose **Administration Manager** from the role bar.
3. From the Administration Utilities toolset, click **eService Register Viewer**.
The eService Register Viewer wizard opens.
4. Choose the viewer you want to work with from the **Viewer Servlet** list.
5. Click **Retrieve Viewer Information**.

The tip for the viewer is shown. This tip will appear in the MatrixOne applications when a user moves the mouse pointer over the icon that represents the viewer.

6. To modify the tip for the viewer, edit it as needed in the **Viewer Tip** box. Then click **Modify Viewer**.

Or:

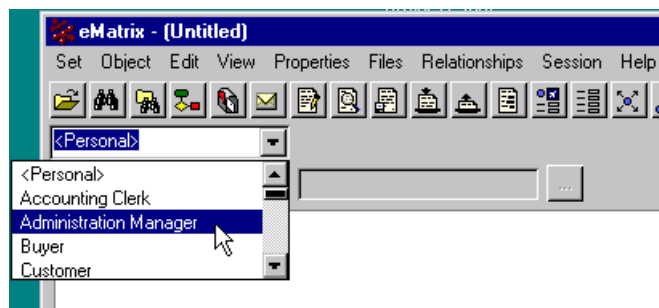
To delete the registration, click **Delete Viewer**.

Assigning Formats to Viewers

Assigning formats defines which viewers a format is compatible with. If you already have the Register Viewer wizard open, skip to step 5.

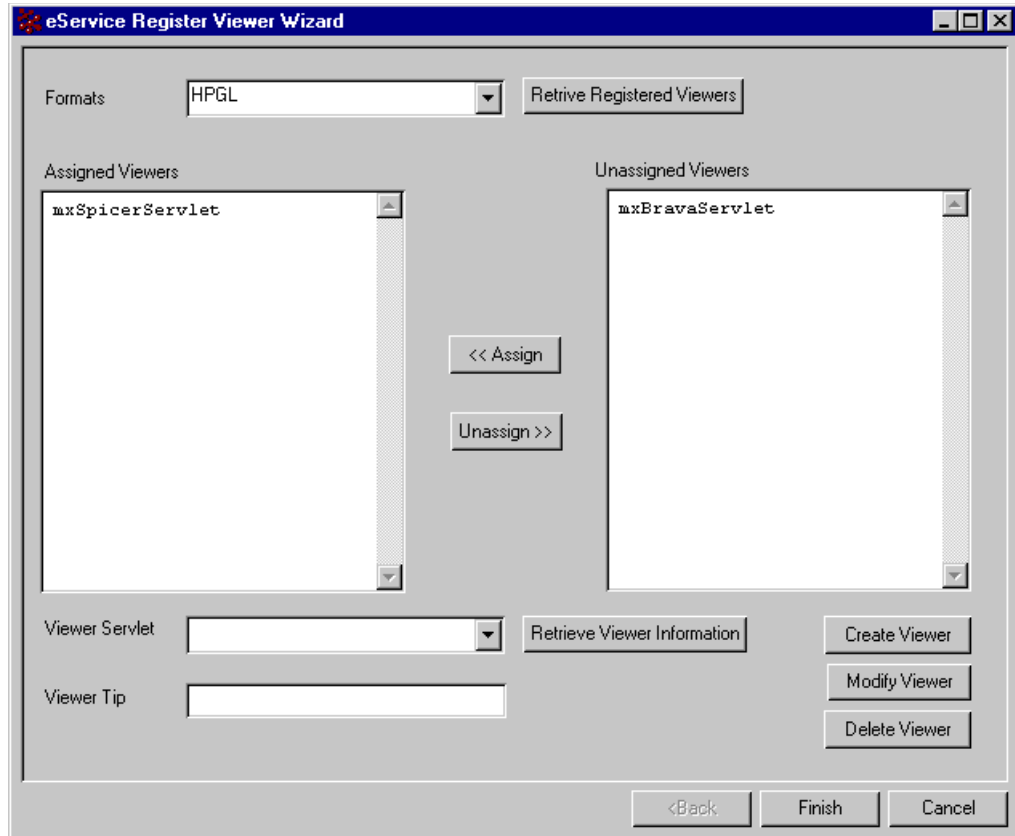
To assign and unassign formats for a viewer

1. Make sure the formats you want to assign as being compatible with the viewer are defined in Business Modeler or MQL.
2. Log into Matrix Navigator as a user who is assigned to the Administration Manager role and who is a Business Administrator-type user with all access privileges (as defined in the person definition).
3. Choose **Administration Manager** from the role bar.



4. From the Administration Utilities toolset, click **eService Register Viewer**.
The eService Register Viewer wizard opens.

5. From the **Formats** list, choose a format that you want to assign viewers to. If the format you want isn't listed, you need to define the format in Business Modeler or MQL.
6. Click **Retrieve Registered Viewers**.
Any viewers already assigned to the format are listed in the Assigned Viewers list. All other registered viewers are listed in the Unassigned Viewers list.
7. Choose viewer that is compatible with the format from the **Unassigned Viewers** list and click **Assign** to move the viewer to the Assigned Viewers list.



8. Repeat the previous step for all viewers that are compatible with the selected format.

Specify File Extensions for Viewers

Tell the system which file extensions it should open in a registered viewer using the following property in `emxSystem.properties`. If a user attempts to view a file whose extension is not in this list, the system does not attempt to open the file in a viewer, even if a viewer has been registered and installed. Instead, the system handles the file as it does any standard file for the application you are in. The standard view behavior for applications is to attempt to open the file in the browser and if the browser doesn't recognize the extension then it prompts the user to save the file. Separate each extension in the list with a comma and no space.

```
emxFramework.ViewerExtensions=jt,dwg
```

Specify Formats for which a Preferred Viewer Must Be Chosen

If a format has several registered viewers, you can require that users choose the format they prefer. When a user chooses to view a file with a format assigned to a 2D/3D viewer, the system first looks for a preferred viewer specified in the person's profile. Each user can specify his/her preferred viewer for a format or the administrator can specify it. If the system finds a preferred viewer for the format, it opens the file using that viewer. If it doesn't find one, the system does not launch any viewer.

You specify the formats for which users must choose a preferred viewer using the following property in `emxSystem.properties` (located in `Apps/Framework/VERSION/UserInterface`).

```
eServiceFramework.FormatsWithViewerPreference  
=Format1SymbolicName,Format2SymbolicName,...
```

For example, if you want users to choose preferred viewers for JT and ACAD files in viewers, set this property to `format_JT,format_ACAD`.

Based on this property, the applications dynamically add fields to the Person profile pages. The system constructs the labels for the fields by appending "Viewer Preference" (from the `ViewerPreferenceLabel` resource tag) to the format name (resolved from the symbolic name).

```
<FormatSymbolicName> <ViewerPreferenceLabelTag>
```

For example, when the `FormatsWithViewerPreference` property is set to `format_JT`, the system adds a JT Viewer Preference field to the Person profile pages.

The screenshot shows a web form titled "Edit Person Details". At the top, a message states "Fields in red italics are required" next to a question mark icon. The form contains several input fields: "Location" (a dropdown menu), "Office Phone Number" (text input with "(123)322-2004"), "Home Phone Number" (text input with "(123)456-7890"), "Web Site" (text input with "http://www.CompanyNam"), "Email Address" (text input with "TestBuyer@CompanyNe"), "Fax Number" (text input with "Unknown"), "Pager Number" (text input with "Unknown"), and "JT Viewer Preference" (a dropdown menu with options "BravaServlet", "BravaServlet", and "eVisServlet"). At the bottom of the form are "Done" and "Cancel" buttons.

The system populates the preference fields by listing every viewer assigned to the format (which is done using the Viewer Registration wizard, described in a previous section). If a tip has been defined for the viewer, the tip text is used to represent the viewer in the drop-down list. If there is no tip text, the name of the program object for the viewer servlet is used.

Choosing a Preferred Viewer

This section describes how a user chooses a preferred viewer and how the preference is stored in the database.

A user (or administrator) chooses a preferred viewer using the Person profile page of the MatrixOne Application that contains the viewer feature (such as Sourcing Central or Team Central). A user can choose his/her own preferred viewer or the administrator can choose it. The steps for editing a user's profile varies with each application so consult the User Guide for the application for details. Here is an example of a page for editing a person's profile.

Edit Person Details

Fields in red italics are required ?

Location	
Office Phone Number	(123)322-2004
Home Phone Number	(123)456-7890
Web Site	http://www.CompanyNam
Email Address	TestBuyer@CompanyNe
Fax Number	Unknown
Pager Number	Unknown
JT Viewer Preference	BravaServlet BravaServlet eVisServlet

Done Cancel

For each preferred viewer that a user chooses, the system creates an administrative property on the person's admin object that represents the servlet to use for the viewer. The property is named **FORMAT-ViewerPreference**, where **FORMAT** is the name of the file format defined in Matrix.

When the user clicks the View and Markup icon for a file with a format that has a preferred viewer selected, the system opens the file in the preferred viewer. If the user hasn't chosen a preferred viewer, the system does not open a viewer.

Some applications let you restrict the list of viewers in the preferred viewer list to a specific role. For information on how to implement this for a particular application, refer to the Administrator's Guide for the application.

Modifying Supplied Source Code

If you decide to modify one of the supplied programs, clone the original program and change the name of the cloned program. The names for all MatrixOne application programs are prefixed with “eService[name of install directory]” or “emx”. Do not change this prefix and don’t create custom programs using the eService prefix.

Customizing User Documentation

This section describes how to make changes to the user documentation that accompanies MatrixOne applications. The applications are designed to be configured to suit the business processes for each customer. The documentation should be updated to reflect changes made to the applications.

If changes are made to an application or to the underlying framework, MatrixOne cannot ensure the accuracy of the documentation. These changes include but are not limited to: changing onscreen text, adding or removing fields on a page, making changes to the administrative objects in the schema, adding new JSPs or changing existing JSPs, changing trigger programs, changing the installation or login process, or changing the values in any properties file.

Overview of User Documentation

Each MatrixOne Application has two forms of user documentation:

- A user guide that is in Acrobat PDF format
The PDF user guide is created from the FrameMaker source files by printing the entire book to a postscript file and then generating the PDF in Acrobat. Users must have Acrobat Reader to view the file. The file is accessible from the online help and can be distributed separately as needed.
- An online help system that is html based
The online help is context-sensitive so when a user clicks help from a page (visual page, not JSP), the help system displays information about that page. The help system uses the same FrameMaker source files used for the PDF user guide. These source files are converted to a JavaScript-based help system using WebWorks Publisher Professional Edition.

The applications also have documentation for Matrix business and system administrators, such as an administrator's guide and a readme that describes installation procedures. Although these documents are also created in FrameMaker and are produced in a similar way, this document applies only to documentation that is aimed at standard end users.

Two Strategies for Making Changes

There are two ways to make changes to the documentation. You can change the “output” files (the PDF file for the user guide, the html and other files for the help) or you can change the source files and then recreate the output files based on the new source files. Unless your changes are minor, such as adding a note or replacing a few words or graphics, it's best to change the source files even though doing so may initially take more time and resources. If you choose to change only the output files:

- You must make the changes to both the pdf and help.
- Maintaining the documentation may be more difficult when you receive a new version of the software with its documentation from MatrixOne because you'll have to make all the changes again to the new output files. If you don't track the changes you make, this could be very difficult.

- The new or changed information will not be included in the search and index features. For example, if you add a new term to the help system, the index and search features will not recognize the term so users will not be able to find the information you added except by browsing through the help system.

The best way to determine which strategy is best for you is to consider how extensive your changes are. If you can make a short list (maybe 6 or 7 items) of exactly what changes need to be made to each set of output files, then changing the output might be acceptable. You can keep this list and when you receive a new set of documentation files, you can implement these few changes again. Also remember to evaluate how serious it is not to have new or changed terms in the index or search.

Changing the Documentation Output

You can make minor content changes—such as changing a term, adding a step to a procedure, or replacing an image—to the output files for both types of documentation. For example, suppose you configure the application by changing a term (replace the word “originator” with “creator”), changing the labels on buttons or links, changing the title of a page, or adding an additional field on a page. You can edit the online help and/or the user guide to reflect these kinds of changes in an application.

Before making changes to the output, read *Two Strategies for Making Changes* to make sure you are aware of the consequences.

Changing the Online Help Output

The files for each online help system are located in `ematrix/doc/APPLICATION_NAME`. Each online help system consists of many different files, including HTML, GIF, XML, and JS files. The written content for the help system is in the HTML files.

To edit text in the help system, edit the HTML file that contains the text using any text editor.

To edit text in a help system

1. Before making changes, you should make a backup copy of the original output files, which are located in `ematrix/doc/APPLICATION_NAME`.
2. Run the help system directly from the Web server directory structure as a standalone system and not from the application. Use Internet Explorer (IE). To run the help system, open `ematrix/doc/APPLICATION_NAME/wwhelp/js/html/frames.htm` in IE.
3. Go to the topic you want to edit.
4. In Internet Explorer, right-click on the right frame and choose **View Source**.
Another way to find the HTML file that contains the text you want to edit is to use the Find feature of Windows Explorer (or the equivalent for UNIX) to search for the file that contains a particular text string.
5. From the text editor, edit the content as needed.

Only people familiar with HTML should edit the files. Make sure you edit only the help text and do not change HTML tags, especially anchor and paragraph IDs (such as `<a name="146224">` or `<p id="145937" class="Body">`). Changing HTML tags may create errors in the context-sensitive help or cross-references.

6. Save the file using the same name as the original file. Make sure the extension is html and not txt. The file name must be exactly the same as the original or the help system will not find the file.
7. If you changed the heading for the topic, make the same change to the Contents tab by changing the heading in ematrix/doc/APPLICATION_NAME/wwhdata/js/toc.js and files.js and ematrix/doc/APPLICATION_NAME/APPLICATIONNAMEtoc.xml.
8. Make a list of the changes you made so you can make the same changes when you receive an updated version of the help system.

To change an image in the help system

1. Open the help system.
2. Right-click on the image.
3. In Internet Explorer, choose **Properties** and find the file name for the image at the end of the URL field.
Or:
In Netscape, the file name is shown next to the View Image option of the right-click menu.
4. Find the file in ematrix/doc/APPLICATION_NAME/images. Open it in an image editor, make changes to it, and save it, or replace the file with an image of the same name.
5. Launch the help system again or refresh the current page to display the changed graphic.

Changing the PDF User Guide

The user guide for each application is located in ematrix/doc/pdf. To edit text in a PDF file, you must have Adobe Acrobat. Open the PDF file in Acrobat and make text changes as needed. Make a list of the changes you made so you can make the same changes when you receive an updated version of the user guide.

Changing the Documentation Source

Contact MatrixOne to receive the source files and instructions for changing the source.

Configuring the File Collaboration Server for use with JSP Applications

For instructions on setting up a File Collaboration Server for use with JSP applications and the MatrixOne Application, see the *Matrix System Guide*.

Changing Shadow Agent Passwords

The password is the same for all shadow agent persons, including all of the grant agents. The password is stored in the code section of the `emxcommonSessionInfo` program object and is “shadowsecret” by default.

It is important that if you change this value in the program object, you also change the password for all of the agents. It is also important that if you are upgrading and have already changed the password, that you change it again in this new location and change all of the new agents before using the system. You may want to change the password back to shadowsecret for the original User Agent person initially until you know everything is working and then change the password for all of the agents at once.

Currently, these are the shadow agent persons that are installed with the AEF.

Shadow Agent Person	Assigned to Group
User Agent	Shadow Agent
Common Access Grantor	Access Grantor
Project Space Access Grantor	Access Grantor
Route Delegation Grantor	Access Grantor
Unmanaged Document Grantor	Access Grantor
Workspace Access Grantor	Access Grantor
Workspace Lead Grantor	Access Grantor
Workspace Member Grantor	Access Grantor

Configuring Approval Verification with External Authentication

This section describes how to require users to enter their username and password when entering approvals for systems with external authentication (Single Signon or SSO). There are two types of approvals: approvals for lifecycle promotions and demotions and indicating an approval option (approve, reject, or abstain) for route tasks. These instructions apply to external authentications that use SiteMinder or ClearTrust.

Note that when using approval verification with SSO, the systems redisplay the username and password page if the person enters incorrect information. The approval will not be accepted until a correct username/password pair is entered.

For information on additional verification configurations that can be implemented in non-SSO environments, see *Miscellaneous Properties* in Chapter 7.

1. Enable lifecycle signature approvals (only when using the AEF's lifecycle page, `emxLifecycle.jsp`):
 - a) For SiteMinder, copy the file `emxLifecycleSSOLogout.fcc` from the `MATRIXHOME/framework/CommonUI` directory to the location used for SiteMinder forms.
 - b) In the `emxSystem.properties` file, set the following properties:

```
emxFramework.LifeCycle.ApprovalPasswordConfirmation=TRUE
emxFramework.LifeCycle.ExternalAuthentication=TRUE
emxFramework.LifeCycle.ExternalAuthenticationURL=URL
```

For SiteMinder, the URL should be the URL to `emxLifecycleSSOLogout.fcc`, which you copied in Step 1a. For example, `http://HOST:PORT/siteminderagent/forms/emxLifecycleSSOLogout.fcc`.
For ClearTrust, the URL should be:

```
emxFramework.LifeCycle.ExternalAuthenticationURL=/cleartrust/
ct_logon.jsp?ct_orig_uri=/ematrix/common/
emxLifecycleSSOAuthentication.jsp?callbackFunctionName=passwordCallback
()
```
 - c) If you are using Document Central, update the `emxDocumentCentral.properties` file with the following setting:

```
eServiceDocumentCentral.General.FDACompliantSignatures = TRUE
```
2. Enable route task approvals (including common component routes and Team Central routes):
 - a) For SiteMinder, copy the `emxTaskSSOLogout.fcc` from `MATRIXHOME/framework/AppsCommon` to the location used for SiteMinder forms.
 - b) In the `emxSystem.properties` file, set the following:

```
emxFramework.Routes.EnableFDA=true
```
 - c) In the `emxComponents.properties` file, set the following:

```
emxComponentsRoutes.InboxTask.ExternalAuthentication= TRUE
emxComponentsRoutes.InboxTask.ExternalAuthenticationURL=URL
```

For SiteMinder, the URL should be the URL to emxTaskSSOLogout.fcc, which you copied in Step 2a. For example, `http://HOST:7001/siteminderagent/forms/emxTaskSSOLogout.fcc`.

For ClearTrust, the URL should be
`emxComponentsRoutes.InboxTask.ExternalAuthenticationURL=/cleartrust/
ct_logon.jsp?ct_orig_uri=/ematrix/components/
emxTaskSSOAuthentication.jsp?callbackFunctionName=passwordCallback()`

- 3.** Run the warutil and deploy the archive file.

Internationalizing the Framework and Applications

Internationalizing the Applications

This section describes guidelines for internationalizing a MatrixOne application, including the login and home pages. These guidelines assume the applications are written in JavaServer Pages (JSPs). These guidelines include:

- the utilities/tag extension used for internationalization and naming conventions for internationalization coding
- the Resource Bundle naming convention, and string key naming and translation

For more information, see Java 2 (JDK 1.2+) documentation.

For information on internationalizing Dynamic UI component text, see [Internationalizing Dynamic UI Components](#) in Chapter 8. For information on how Matrix desktop applications and Matrix Navigator for the Web are internationalized, see “Localizing Matrix” in the *Matrix PLM Platform Installation Guide*.

Resource Bundle File Naming

The naming convention for the base resource file, which is always in English, is: `emxAPPLICATION_NAMEStringResource.properties`. For example, the string resource file for Team Central is `emxTeamCentralStringResource.properties`. The string resource

file for pages that can be available for any application, such as the login and home page, is `emxFrameworkStringResource.properties`.

The naming convention for all internationalized versions of the `StringResource` files is `emxAPPLICATION_NAMEStringResource_LANGUAGE.properties`, where `LANGUAGE` is the standard abbreviation for the language, such as `fr` for French or `de` for German, in lowercase.

All string resource files must be located in `Apps/Framework/VERSION/UserInterface`.

In order for the applications to read the `StringResource` properties files, they must be encoded in ASCII. This is true for all languages, both double-byte and latin-based languages. To edit text for languages that include non-ASCII characters, work with the native file that is included in the properties directory (such as `emxFrameworkStringResource_jaNative.properties`) and then convert the modified file to ASCII. These native files are encoded using the native format for the language. For the English files, you need only make sure you save the properties file as ASCII.

Introducing New Languages

You can introduce new settings that target a language, a country, and a variant. To do this, copy the original string resource properties file and rename it using the following convention:

`ORIGINAL_FILENAME_LANGUAGE_COUNTRY_VARIANT.properties`

`LANGUAGE` is the language code are defined by the ISO-3166 standards. For example, “en” is English and “es” is Spanish. `COUNTRY` is the country code defined by the ISO-639 standard. For example, `US` is for the United States and `UK` is for the United Kingdom. `VARIANT` is up to you.

For example:

`emxTeamCentralStringResource_en_us_win.properties`

Depending on the request received from the user's browser, the program will look for:
`emxTeamCentralStringResource_en_us_win.properties`

then for:

`emxTeamCentralStringResource_en_us.properties`

then for:

`emxTeamCentralStringResource_en.properties`

then for:

`emxTeamCentralStringResource.properties`

When introducing new languages, it's preferable to use the language extension only. Unless there is a need for translations for different countries, the translation should be done at the language level only.

The string resource property files must be located in the application server directory structure, for example, `Apps/Framework/VERSION/UserInterface`.

Resource String Naming

Each string in the resource file should be formatted as follows to prevent name collisions:

`emxAPPLICATION_NAME.PAGE.KEY=STRING_VALUE`

For example, for a page named `emxTeamI18nTest.jsp` in the Team Central application, the strings for a greeting and farewell would be entered as follows in the string resource file for English:

```
emxTeamCentral.TeamI18nTest.greeting=Hello
```

```
emxTeamCentral.TeamI18nTest.farewell=Goodbye
```

Sample of an Internationalized JSP

```
<%@ taglib uri="/emxUtil.tld" prefix="emxUtil" %>
<%@ page import="matrix.db.*, matrix.util.*,
com.matrixone.servlet.*, java.util.* , java.net.URLEncoder,
com.matrixone.apps.teamcentral.i18n.*" %>
<%include file = "emxTeamI18n.jsp"%>

<emxUtil:localize id="i18nId"
bundle="emxTeamCentralStringResource" locale='<%=
request.getHeader("Accept-Language") %>' />
<html>
<head>
<title>i18n Sample page</title>
</head>
<body >
<emxUtil:i18n>emxTeamCentral.i18nTest.greeting</mxUtil:i18n>
</body>
</html>
```

`emxTeamI18n.jsp` defines two methods that can be used for internationalization.

```
<%@page import="matrix.db.*, matrix.util.*, java.util.* ,
com.matrixone.apps.teamcentral.i18n.*" %>
<%
    String DateFrm = (new
Integer(java.text.DateFormat.MEDIUM)).toString();
%>

<%!
    // Call this method to internationalize variables in java.
    // i18nStringNow("stringid",
request.getHeader("Accept-Language"));
    static public String i18nStringNow(String text, String
languageStr) {
        i18nNow loc = new i18nNow();
        return
(String)loc.GetString("emxTeamCentralStringResource",
languageStr, text);
    }
%>
```

Localizing the Date and Time

Here is a sample of a table cell that displays a Due Date in the user's language and in the appropriate time zone.

```
<td><emxUtil:lzDate displaydate="true" localize="i18nId"
tz='<%=timeZone%>' %sDueDate%></emxUtil:lzDate>&nbsp;</td>
```

When the displaydate parameter is true, only the date is displayed. To display only the time, use displaytime=true. To show both the date and time, don't specify either parameter. The timezone must be stored as a variable which is offset in hours from Greenwich Mean Time.

Debugging i18n

A back-door solution is provided for debugging internationalized pages. When the locale is set to debug, the i18n tag returns the string ID instead of the actual string. The following is the same code presented previously, but translated:

```
<%@ taglib uri="/emxUtil.tld" prefix="emxUtil" %>
<%@ page import="matrix.db.*, matrix.util.*,
com.matrixone.servlet.*, java.util.* , java.net.URLEncoder,
com.matrixone.apps.teamcentral.i18n.*" %>
<%@include file = "emxTeamI18n.jsp"%>

<emxUtil:localize id="i18nId"
bundle="emxTeamCentralStringResource" locale='debug' />
<html>
<head>
<title>i18n Sample page</title>
</head>
<body >
<emxUtil:i18n>emxTeamCentral.i18nTest.greeting</mxUtil:i18n>
</body>
</html>
```

Translating Schema Names

This section describes how to translate names of schema objects, including attribute names and ranges, type names, and policy states. The section includes methods and code snippets for calling these translated names in a JSP.

The aliasing method for translating schema, which uses the Language Alias wizard, has been deprecated and should not be used.

Adding and Editing Translations for Schema

Translations for schema names are stored using strings in the `emxFrameworkStringResource.properties` files, which are installed as part of the framework. Each schema object has a unique string ID, which includes the actual name of the object in the schema. Therefore, if you change the name of a schema object, you must rename the string ID for the object in the English `emxFrameworkStringResource.properties` file, as well as in each of the translated versions of the file (such as `emxFrameworkStringResource_ja.properties`). Similarly, if you add new schema that you want to be translated, you must add a string for it in each property file.

In order for the applications to read the properties files, they must be encoded in ASCII. This is true for all languages, both double-byte and latin-based languages. To edit text for languages that include non-ASCII characters, work with the native file that is included in the properties directory (such as `emxFrameworkStringResource_jaNative.properties`) and then convert the modified file to ASCII, as described below. These native files are encoded using the native format for the language. For the English files, you need only make sure you save the properties file as ASCII.

To add and edit translations for schema

1. If you are changing names of schema in English, open the `emxFrameworkStringResource.properties` file in any text editor. All properties files are located in `Apps/Framework/VERSION/UserInterface`.
Or:
If you are revising text for any non-English language:
 - a) Open the `emxFrameworkStringResource_LANG.properties` file using Microsoft Word.
Word will prompt you to choose the file encoding necessary to load the file.
 - b) Accept the default by clicking **OK**.
2. Add string IDs for new schema and edit the IDs for renamed schema as needed. Follow the formats described in [Formats for Schema String Resource IDs](#).
3. If you are working with an English file, save the file normally.
If you are working with a non-English file:
 - a) Save the file as encoded text.
 - b) Click **Yes** to save it in text format, and then choose the same encoding that was used to load the file, which should be the default.
 - c) Click **OK**.

d) Convert the file to ASCII by running the following command:

```
native2ascii -encoding CODE INPUTFILE OUTPUTFILE
```

CODE is the language encoding (shown in the files header).

INPUTFILE is the name of the file you edited.

OUTPUTFILE is the name of the file to be created. The output file will be in Unicode, which means it will contain letters followed by escape characters in place of the native characters.

For example:

```
native2ascii -encoding JIS
emxFrameworkStringResource_ja_Native.properties
emxFrameworkStringResource_ja.properties
```

4. Make sure the saved file is in Apps/Framework/VERSION/UserInterface and make a backup copy of the property files.
5. If you are running a J2EE implementation, remember to run the warutil and deploy the archive file.
6. Restart the app server.

Always make backups of properties files that you configure. When you install a new version of the framework or applications, the properties files will be overwritten with updated files.

Formats for Schema String Resource IDs

When adding and changing resource file IDs, you must follow the defined formats. There is a standard format for all administrative objects and a slightly different format for attribute ranges and policy states.

Use only English characters in string resource IDs. Of course, the translated values can have non-English characters.

Administrative Objects

emxFramework.<Admin Object Type>.<Admin Object Name>=<Translated Value>

Where:

- emxFramework is the keyword to group all these resource IDs as part of the framework.
- Admin Object Type is the admin object type, such as Type, Role, Attribute, etc.
- Admin Object Name is the name of the admin object that needs to be translated. All spaces should be replaced with “_” for this ID.
- Translated Value is the translated value for the admin object.

Examples (for French translation of schema objects with English names):

- emxFramework.Type.Manufacturing_Process=Processus de fabrication
- emxFramework.Role.Buyer=Acheteur
- emxFramework.Attribute.Count=Total

Attribute Ranges

emxFramework.Range.<Attribute Name>.<Range Name>= Range Name Translated

Where:

- `emxFramework.Range` is the key that groups all the range names within the `FrameworkStringResource` file.
- `Attribute Name` is the actual name of the attribute to which the range value is associated. The blank spaces are replaced with “_”.
- `Range Name` is the actual value of the attribute range for which translation is required. The blank spaces are replaced with “_”.
- `Attribute Name Translated` is the translated value for the specified type name.

Examples (for French translation of schema objects with English names):

- `emxFramework.Range.Approval_Status.Abstain=S'abstenir`
- `emxFramework.Range.Approval_Status.Approve=Approuver`
- `emxFramework.Range.Create_Folder.No=Non`
- `emxFramework.Range.Create_Folder.Yes=Oui`

Policy States

`emxFramework.State.<Policy Name>.<State Name>= <State Name Translated>`

Where:

- `emxFramework.State` is the key that groups all the state names with in the `FrameworkStringResource` file.
- `Policy Name` is the actual name of the policy to which the state is associated. The blank spaces are replaced with “_”.
- `State Name` is the actual name of the state for which translation is required. The blank spaces are replaced with “_”.
- `State Name Translated` is the translated value for the specified state name.

Examples (for French translation of schema objects with English names):

- `emxFramework.State.Build_Archive.Archive=Archiver`
- `emxFramework.State.Buyer_Desk.Exists=Existe`
- `emxFramework.State.CAD_Model.Release=Distribuer`

Implementing Schema Translations within a JSP

To use the translated schema names within a JSP, you need to:

1. *Look up the actual schema object name.*
2. *Include `emxI18NMethods.jsp`.*
3. *Get the browser language settings.*
4. *Get the translated schema names in the JSP.*

Look up the actual schema object name

If the actual admin object name is not available dynamically to the JSP, use the symbolic name to look up the actual name of the object. Always use the object’s symbolic name to look up the actual name. To get the actual name of any admin object, use the `getPropertyValue` method and enter the corresponding symbolic name and the session object as input parameters.

Examples:

- `String typeName = Framework.getPropertyValue(session, "type_Part");`
Returns the actual admin object name - "Part"
- `String attributeName = Framework.getPropertyValue(session, "attribute_LeadTime");`
Returns the actual admin object name - "Lead Time"
- `String roleName = Framework.getPropertyValue(session, "role_DesignEngineer");`
Returns the actual admin object name - "Design Engineer"
- `String stateName = Framework.getPropertyValue(session, "policy_ECPart");`
Returns the actual admin object name - "EC Part"

Include emxI18NMethods.jsp

The utility methods that get the translated schema names from the StringResource files are defined in the `emxI18NMethods.jsp`, which needs to be included in any JSP that uses the methods. These methods look up to the resource bundle to get the translated value. If not found, it returns the same value back.

Get the browser language settings

A language string that defines the current language setting in the browser is required as one of the input parameters for all the utility methods. The following statement gets the language settings.

```
String languageStr = request.getHeader("Accept-Language");
```

Get the translated schema names in the JSP

Method signature for admin objects

```
String displayAdminName = getAdminI18NString (<Admin Type>, <Admin Name>, languageStr);
```

Where:

- Admin Type is the type of admin object, such as Type, Attribute, Role.
- Admin Name is the name of the admin object for which translation is required. If the actual name is not available dynamically in the page, the name may be obtained from the symbolic name and using the lookup method described in [Look up the actual schema object name](#).
- LanguageStr is the language setting obtained from the current request object.

Examples:

- `String displayAdminName = getAdminI18NString ("Type", "Part", languageStr);`
Resource bundle property: `emxFramework.Type.Part= Part`
Returns: "Part"
- `String displayAdminName = getAdminI18NString ("Type", "Mechanical Part", languageStr);`
Resource bundle property: `emxFramework.Type.Mechanical_Part=Mechanical Part`
Returns: "Mechanical Part"

- `String displayAdminName = getAdminI18NString (“Policy”, “EC Part”, languageStr);`
Resource bundle property: `emxFramework.Policy.EC_Part= EC Part`
Returns: “EC Part”
- `String displayAdminName = getAdminI18NString (“Relationship”, “EBOM”, languageStr);`
Resource bundle property: `emxFramework.Relationship.EBOM= EBOM`
Returns: “EBOM”
- `String displayAdminName = getAdminI18NString (“Attribute”, “Lead Time”, languageStr);`
Resource bundle property: `emxFramework.Attribute.Lead_Time= Lead Time`
Returns: “Lead Time”
- `String displayAdminName = getAdminI18NString (“Basic”, “Revision”, languageStr);`
Resource bundle property: `emxFramework.Basic.Revision= Revision`
Returns: “Revision”

Method signature for policy state names

```
String displayStateName = getStateI18NString (<Policy Name>,
<State Name>, languageStr);
```

Where:

- Policy Name is the actual name of the policy for which the states are associated. If the actual name is not available dynamically in the page, the name may be obtained from the symbolic name and using the lookup method described [Look up the actual schema object name](#).
- State Name is the actual name of the state for which translation is required. If the actual name is not available dynamically in the page, the name may be obtained from the symbolic name and using the lookup method.
- LanguageStr is the language setting obtained from the current request object.

Examples:

- `String displayStateName = getStateI18NString (“EC Part”, “Preliminary”, languageStr);`
Resource bundle property: `emxFramework.State.EC_Part.Preliminary = Preliminary`
Returns: “Preliminary”
- `String displayStateName = getStateI18NString (“EC Part”, “Release”, languageStr);`
Resource bundle property: `emxFramework.State.EC_Part. Release = Release`
Returns: “Release”

Method signature for attribute ranges

```
String dispRangeName = getRangeI18NString (<Attribute Name>,
<Range Value>, languageStr)
```

Where:

- Attribute Name is the actual name of the attribute that has this range value associated with it. If the actual name is not available dynamically in the page, the name may be obtained from the symbolic name and using the lookup method described in [Look up the actual schema object name](#).

- Range Value is one of the range values associated with this attribute and that needs to be translated. These range values may be obtained from the database for the given attribute.
- LanguageStr is the language setting obtained from the current request object.

Examples:

- String dispRangeName = getRangeI18NString ("Lead Time", "8 Weeks", languageStr)
Resource bundle property: emxFramework.Range.Lead_Time.8_Weeks = 8 Weeks
Returns: "8 Weeks"
- String dispRangeName = getRangeI18NString ("Lead Time", "6 Weeks", languageStr)
Resource bundle property: emxFramework.Range.Lead_Time.6_Weeks = 6 Weeks
Returns: "6 Weeks"
- String dispRangeName = getRangeI18NString ("Lead Time", "4 Weeks", languageStr)
Resource bundle property: emxFramework.Range.Lead_Time.4_Weeks = 4 Weeks
Returns: "4 Weeks"

Sample Code for Getting Admin Object Translations

Sample JSP code for getting the admin objects translation is given below. Typically, you'd get the current name of the admin object using the symbolic name. For instructions on how to get the current admin object name, see [Getting Admin Object Names from Symbolic Names](#) in Appendix A.

```
//*****
<%@include file = " emxI18NMethods.jsp"%>

<%
String languageStr = request.getHeader("Accept-Language");

// Get the Part Type name
String sPartType = getAdminI18NString("Type", "Part",
languageStr );

// Get the Role name
String sRoleName = getAdminI18NString("Role", "Design
Engineer", languageStr);

// Get the Attribute name
String sAttributeName = getAdminI18NString("Attribute", "Lead
Time", languageStr);

// Get the Policy name
String sPolicyName = getAdminI18NString("Policy", "EC Part",
languageStr);

// Get the Relationship name
String sRelName = getAdminI18NString("Relationship", "EBOM",
languageStr);
```


%>

// *****

Installing Internationalized User Documentation

The installations for the framework and MatrixOne applications does not include internationalized versions of the online help and user guides. When a user's browser is set to a language other than English, the application appears in that language but the user documentation remains in English. You can download internationalized versions of the user documentation from MatrixOne's CustomerLink Web site. Generally, the documentation is available in French, German, Italian, and Japanese. Check CustomerLink for the specific languages available for your application and to see what version of the application the documentation reflects.

To install versions of user documentation in languages other than English

1. Log onto MatrixOne CustomerLink at https://customerlink.matrixone.com/custlink/cust_welcome.jsp.
2. Go to the Documentation tab and click "Translation In Other Languages" at top of page.
3. Click on the language and version you need, such as "French Documentation 10.x".
4. Checkout the files checked into the PDF and HTML formats for the documents you need. Include the Common and AEF documentation and whatever applications you require.
5. Place the pdf file(s) in this directory on the server:
ematrix/doc/pdf
6. In the ematrix/doc/APP_NAME directory of the server, create a directory whose name is the standard, 2-letter abbreviation for the language.

APP_NAME is the directory for the application you are working with, such as sourcingcentral. The en directory, which contains the English help files, will already be in the APP_NAME directory.

For example, if you are installing German documentation for Matrixone Engineering Central, create a de directory as follows: ematrix/doc/engineeringcentral/de.
7. Unzip the files for the application's help system into the directory you created, making sure the sub-folder structure of the extracted files is preserved.
8. In the ematrix/doc/common directory, create a directory whose name is the standard, 2-letter abbreviation for the language.
9. Unzip the files for the help system into the directory you created.
10. In the emxFrameworkStringResource.properties file for the language you are installing, change the value of the emxFramework.HelpDirectory=en property to the 2-letter abbreviation for the language. This file is located in Apps/Framework/VERSION/UserInterface.

For example, if you are installing the French documentation for Sourcing Central, open the emxFrameworkStringResource_fr.properties file and change emxFramework.HelpDirectory=en to emxFramework.HelpDirectory=fr.
11. Run the warutil and deploy the archive file.
12. Restart the app server.

Internationalizing System-Generated Notifications

This section describes how to internationalize notifications generated by the MatrixOne applications. For example, Team Central sends notifications to a person when an event occurs that the person has subscribed to, such as when a user adds a document to a folder. All applications with a routing feature notify task assignees when their tasks are created. These system-generated emails can be sent in the language chosen by the user or they can be sent with the message text repeated in multiple languages.

Note that only messages whose text is translated within the appropriate `emxFrameworkStringResource.properties` file can be internationalized. Therefore, ad hoc messages created by a user cannot be internationalized. When a user creates a new message, all recipients receive the message in the language used by the message creator.

Specifying Default Languages to Include in System Notifications

Using the Default Language option of the Preferences window, users can choose a language preference for system notifications. If you are using an application that does not have this option (it was removed from the Preferences page) or if a particular user does not choose a preferred language, then you can have the system send notifications in multiple languages. To specify the languages you want to include, use the following property in `emxSystem.properties`.

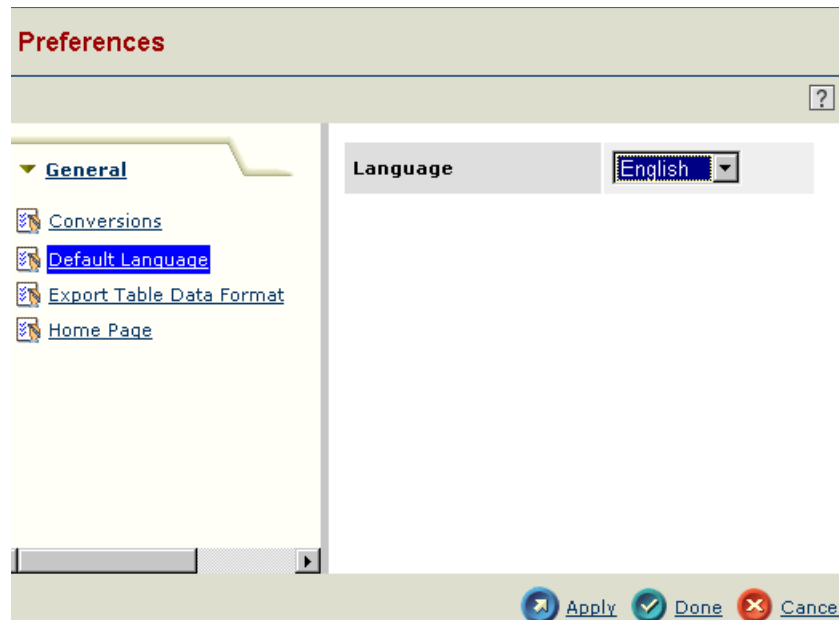
```
emxFramework.NotificationLanguages = en
```

The message text of each system-generated notification will be repeated in all of the languages you specify. The subject will be in the first language listed. Use the standard abbreviation for each language and separate languages by a space. For example, if you enter the value “en fr de”, the subject of all system emails will be in English. Each message will be repeated in English, French, and German.

You can specify any language that has a corresponding `emxFrameworkStringResource.properties` file. By default, these languages are English (en), French (fr), German (de), Italian (it), and Japanese (ja).

Defining the Languages Users Can Choose for System Notifications

Using the Default Language option of the Preferences window, users can choose the language in which they want to receive system notifications.



How the Language Drop-Down is Defined

The languages available in the Language drop-down list are defined using these properties in `emxSystem.properties`.

```
emxFramework.IconMailLanguagePreference=English German French
Italian Japanese
emxFramework.IconMailLanguageLocaleEnglish=en
emxFramework.IconMailLanguageLocaleGerman=de
emxFramework.IconMailLanguageLocaleFrench=fr
emxFramework.IconMailLanguageLocaleItalian=it
emxFramework.IconMailLanguageLocaleJapanese=ja
```

When a language preference is chosen for a user, the system creates an administrative property called `IconMailLanguagePreference` and sets the value for the property equal to the corresponding `IconMailLanguageLocal` property. For example, if a person chooses French for the preferred language, the system creates an `IconMailLanguagePreference` property with the value `fr`.

The text in the drop-down list is defined in the `emxFrameworkStringResource.properties` files. Here are the properties in the English version of the file. The other versions of the `StringResource` files have an equivalent set of properties with values appropriate for each language.

```
emxFramework.IconMailLanguageEnglish=English
emxFramework.IconMailLanguageGerman=German
emxFramework.IconMailLanguageFrench=French
emxFramework.IconMailLanguageItalian=Italian
emxFramework.IconMailLanguageJapanese=Japanese
```

How the User-Specific System Notifications Work

When an application sends a system notification, it looks for an `IconMailLanguagePreference` property attached to each recipient's Person admin object. If it finds one, it sends the notification in the language that corresponds to the property's value (it en fr, etc.). If it doesn't find a property (for example, the person hasn't selected a preference), it sends the notification in the languages defined in the `emxFramework.NotificationLanguages` property (as described in [Specifying Default Languages to Include in System Notifications](#)).

When sending a notification to more than one recipient, the applications send one notification for every unique language. For example, suppose there are three recipients: two have Italian as the preferred language and one has French. The application will send one Italian notification to the two Italian speakers and one French notification to the French speaker. The French speaker will only see his name in the recipient fields and the Italian speakers will only see their two names in the recipient fields.

The user-specific language preference only works when the recipients are persons and does not work when roles or groups are specified. Currently, no application sends system notifications directly to roles or groups. Sourcing Central does send some notifications to roles but first obtains a list of all persons in the company with the role and then fills in the recipient fields with those persons.

Removing a Language from the Language Preference List

To remove a language from the Language list

1. Open `emxSystem.properties`, which is located in `Apps/Framework/VERSION/UserInterface`.
2. Delete the language and the space following it from the `IconMailLanguagePreference` property.

For example, to remove Italian, you would change this line:

```
emxFramework.IconMailLanguagePreference=English German French  
Italian Japanese
```

To this:

```
emxFramework.IconMailLanguagePreference=English German French  
Japanese
```

3. Comment out the `IconMailLanguageLocale` property that corresponds to it by adding a `#` to the beginning of the line. For example:

```
emxFramework.IconMailLanguageLocaleEnglish=en  
emxFramework.IconMailLanguageLocaleGerman=de  
emxFramework.IconMailLanguageLocaleFrench=fr  
#emxFramework.IconMailLanguageLocaleItalian=it  
emxFramework.IconMailLanguageLocaleJapanese=ja
```

4. Save the file.
5. Run the `warutil` and deploy the archive file as always when making changes to files under the staging directory.
6. Restart the application server.

Adding a Language

To add a language to the Language list

1. Open `emxSystem.properties`, which is located in `Apps/Framework/VERSION/UserInterface`.
2. Add it to the `IconMailLanguagePreference` property, separated from the other languages with a space.
3. Add an `IconMailLanguageLocale` property for the language and set the value to the standard abbreviation for the language.
4. Save the file.
5. Make sure you run the `warutil` and deploy the archive file as always when making changes to files under the staging directory.
6. Restart the application server.
7. Make sure the `emxFrameworkStringResource.properties` file for the language has a set of properties for the drop-down list text.

Login and System Properties

Configuring the Login Process

Applications based on the dynamic user interface use `emxLogin.jsp` and the properties files. When a user accesses an application based on the dynamic user interface, the system goes to a general JSP for all applications, `emxNavigator.jsp`. The application submenus and commands are represented by administrative objects in the database, as described in [Overview of Dynamic User Interface](#) in Chapter 8.

The `StringResource.properties` files are different from the properties files discussed in this section. The `StringResource.properties` files define all text used in an application for internationalization. There are `StringResource.properties` files for each application and for the framework; one for each language. For more information about these files, see [Internationalizing the Applications](#) in Chapter 6.

Login Page

Users enter a username and password using `emxLogin.jsp`. If the username and password is valid, the page forwards to `emxNavigator.jsp`.

If your system uses Single Signon (SSO) for authenticating users, then users will use the SSO tool's login page and so the system bypasses `emxLogin.jsp`.


 A screenshot of the MatrixOne login interface. It features a blue header bar with the 'MatrixOne' logo on the right. The main content area is white and contains two text input fields: the first is labeled 'Username' and the second is labeled 'Password'. Below these fields is a blue circular button with a white right-pointing arrow, followed by the text 'Login'.

A Matrix administrator can require a user to change his/her password by setting Password Change Required for the person's admin object. When the person attempts to log in, a message appears that says the person must change the password. When the user clicks OK, the Login page displays fields for changing the password.

The change password fields also appear on the Login page when the administrator sets system-wide password expirations for every *x* days. In Business Modeler, system-wide password settings are in the Settings>Password menu. Every *x* number of days, all users who have a password will be prompted with the page shown above. They must change their password before they can log in.

Requirements for Successful Login

To log into a dynamic user interface application, a person must enter the username and password for a person who has:

- a valid person admin object defined with that username and password
- a matching Person business object that is in the Active state and connected to an Active Company business object with the Employee relationship

If the first bullet is met but at least one of the conditions in the second bullet are not met, the system presents an error message that asks the person to contact their administrator. When the person chooses a dynamic user interface application, they'll receive the error message. (The additional check for the items in the second bullet does not occur if the user chooses Info Central or a pre-dynamic UI application. For information on the login requirements for a particular pre-dynamic UI application, see the Administrator's guide for that application.)

Logging into a dynamic UI application gains the person access to `emxNavigator.jsp`. But to access menu items and other application components, the person must belong to roles that have access to those components, as described in *Controlling User Access to User Interface Components* in Chapter 8.

To log in as a different user, you must log out as the current user first. If you try to access the login page without first logging out, the system displays your default home page within `emxNavigator.jsp`.

Login Properties File Description

You can use the properties file called `emxLogin.properties` to configure the look and content of the login page. This file is located in the `Apps/Framework/VERSION/UserInterface` directory.

As with all properties files, after making changes, save the file and restart the Web server. If you are running a J2EE implementation, run the warutil and deploy the archive file. The properties file lets you control:

- The action the page should perform when the user clicks the submit button. By default, this action is to call the login servlet.

```
# default: emxLogin.FormAction= /servlet/login
emxLogin.FormAction= /servlet/login
```
- The main image to display on the login page. This image should be located in the ematrix directory.

```
#define an image as the default for the login page
emxLogin.DefaultLoginImage= defaultLoginImage.gif
```
- The target page to load. A target page is required when using the login servlet. By default the target page is emxNavigator.jsp, which loads the emxSystem.properties. For more information, see [Configuring System Properties](#).

```
# framework target Page
# if form action is /servlet/login this is required
# comment out if not needed
emxLogin.FrameworkTarget= emxNavigator.jsp
```
- Whether an additional page should be included on the login page and if so, the name of the page. The page must be a JSP, HTML, or JS file and will be displayed under the Login button. For example, you might want to include a page that contains branding information, information about the software version, or a link to a Contact page.

```
#if true, code will be included from the defined page
emxLogin.UseLoginInclude= false

#login include page
#here you can include a .jsp, .html or .js file
emxLogin.LoginIncludePage= emxLoginInclude.jsp
```
- The background color and image. The image must be located in the ematrix directory.

```
# background color of page
# default is: bgColor=#ffffff
emxLogin.BodyColor= bgColor=#006699

# background image for page
# default is: none
emxLogin.BodyImage= defaultLoginBackground.gif
```
- The color for error text, and the top and bottom bars on the login page.

```
# color of top and bottom bars, default is transparent
# enter colors as hex/html colors
emxLogin.AccentColor= #000000

# color to display error code
# default is #ff0000
emxLogin.ErrorColor= #ffffff
```
- The image to use on the contact us page (if a link is provided on the login page).

```
# The image to be used on the Contact Us page
```

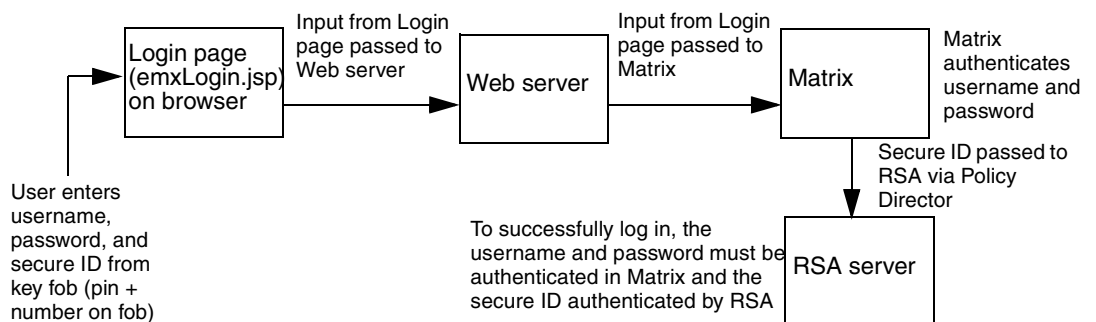
```
emxLogin.contactImageUrl = images/logoDefault.gif
```

- Whether a membership agreement appears for new users and users who have not logged in for 14 days.
View the Membership Dialog
emxLogin.viewAgreement=false
- Whether the login page includes a Register Here link. If this property is set to True, users can click this link and go to a page that lets them fill in registration information (name, company, address, etc.). The system adds the person to the database and emails the person's Company Representative to request that the person be activated. A person who registers cannot log in until they are activated.
View the Register Here Link
emxLogin.RegisterHere=false
- Whether a Secure ID field is displayed on the login page, just below the Password field. The Secure ID field is for integrating with the RSA SecurID® user authentication tool, which is currently only supported through IBM/Tivoli Policy Director and WebSEAL. These components must be installed and configured in order for the Secure ID field to work.
#Use the secure Id field
emxLogin.UseSecureId= false

Secure ID and RSA Token Authentication

If the *Login Type* attribute for the user's Person business object is set to Secure ID and the emxLogin.UseSecureId property is set to true, the login page requires the user to fill in the Secure ID field. If the user's Login Type is set to Standard, the login page ignores any entry in the field.

The diagram below shows how the system authenticates a user whose Login Type is set to Secure ID and when the UseSecureID property is set to show the Secure ID field on the login page. To implement this kind of authentication, you need tokenAuthen.jar and supporting native libraries, which are all supplied by IBM/Tivoli, on your Web server.



Single Signon

To have the MatrixOne applications work in a Single Signon (SSO) environment, a Single Signon server must be installed and configured. See the *Matrix PLM Platform Installation Guide* for details.

When configuring the SSO server, make sure the `emxLogin.jsp` and `emxLogout.jsp` pages are by-passed. Use this property in `framework.properties` to specify the page to go to instead:

```
ematrix.sso.logout.url=URL
```

When running in an SSO environment that has been set up to by pass the `emxLogin.jsp` and `emxLogout.jsp` pages, the system:

- forwards to `emxNavigator.jsp` after a user logs in through the SSO login page. Note that if a user accesses the `emxLogin.jsp` directly (by entering the URL in the browser or by clicking a shortcut to the URL), the system goes to the SSO login page instead.
- forwards to the page set up on the SSO server for bypassing `emxLogout.jsp`, such as a portal page, when a person clicks the Logout button on `emxNavigator.jsp`. The person is not logged out until the browser is closed and cannot log in as a new user until a new browser window is opened.

If the SSO server is not configured to bypass `emxLogout.jsp`, the system returns to `emxNavigator.jsp` when a user attempts to log out.

The framework installs with a command object called `AEFLogoutToolbar`, which creates the Logout tool on the global toolbar. The Logout tool calls `emxLogout.jsp`. For more information on the Logout tool and command object, see [Logout Command](#) in Chapter 8.

To by pass the standard logout behavior, remove the framework's Logout tool and add a custom logout tool that points to a custom JSP. The custom JSP should perform the SSO logout and forward users to another main page from which they can log back in. You configure the page a command object calls using the `href` parameter. The most efficient way to create a new logout command is to clone `AEFLogoutToolbar` and then change the `href` parameter to point to your custom JSP. For more information on configuring global toolbar commands, see [Building Menus and the Global Toolbar](#) in Chapter 8.

Configuring System Properties

Before `emxSystem.properties` is loaded, the system checks the locale setting of the browser using the HTTP request to determine the user's language preference. Based on this, the appropriate properties file is loaded (Java `ResourceBundle` logic is applied to locate the property file). This ensures that properties for a specific language are used on the Application Menu page.

The `emxSystem.properties` file contains properties that define the properties files to call for installed applications, default Home page for `emxNavigator.jsp`, and other properties that apply to all applications.

As with all properties files, after making changes, save the file and restart the application server. If you are running a J2EE implementation, run the `warutil` and deploy the files.

- [*Search Properties*](#)
- [*Miscellaneous Properties*](#)
- [*File Management Properties*](#)
- [*Special Character Properties*](#)
- [*Email Notification Properties*](#)
- [*IconMail Properties*](#)
- [*Initialization Properties*](#)
- [*Engineering Central Properties*](#)
- [*Date/Time Format Properties*](#)
- [*Framework Directories and Resource File Properties*](#)
- [*Start Page, Home Page, and Banner Properties*](#)
- [*Configurable Tree \(Context Navigator/Structure Navigator\) Properties*](#)
- [*Configurable Table Properties*](#)
- [*Application Properties*](#)

The `emxSystem.properties` file also contains properties for configuring preferences that are installed with the framework. For descriptions of these properties, see [*Configuring General Framework Preferences*](#) in Chapter 8.

Search Properties

- **Set default query limit**—The search page allows limiting the number of items returned in the search. By default, the limit is 100 items. You can change the default using the following property:
`emxFramework.Search.QueryLimit = 100`
- **Specify search results maximum**—To prevent destructive searches by users, the Business Administrator can set a maximum limit on search results. On the search dialog, the user can specify a value to limit the number of items returned. If the user types a number higher than what is set in this property, an error message is shown and the user must type a number lower than the maximum. The default is 1000. Allowable values are 1 to 32767.
`emxFramework.Search.UpperQueryLimit=1000`

- **Define vaults for searching**—The vault field is a system preference for vaults to be used in the search. The list of available vaults to search will depend on the person's preference setting and the following property setting:
`emxFramework.DefaultSearchVaults = ALL_VAULTS`
 This property can be assigned to any one of the following values:
 - ALL_VAULTS – all main and secondary vaults for the company to which the person belongs. This list will include both local and remote vaults.
 - LOCAL_VAULTS – all main and secondary vaults for the company to which the person belongs. This list will include only local vaults.
 - DEFAULT_VAULT (default vault of the logged in person)
 - String of vaults [separted by commas].
- **Specify the default types for the generic Search page type chooser**—When a user with the Administration Manager role clicks the Search button on the global toolbar, the generic Search page opens. The Type field contains an ellipsis button to open the type chooser. Use this property to specify the types listed in the type chooser. The list should contain top-level types, since you can drill down for the child types. The following is the default:
`emxFramework.GenericSearch.Types = type_Part,type_DOCUMENTS,type_ECR,type_ECO,type_ProjectSpace,type_Project`
 The type list can also be overridden by passing appropriate URL parameters. For details, see [URL Parameters for Generic Search](#) in Chapter 8.
- **Specify additional basic search attributes**. Clicking on the “More” link on the Search page displays the list of few additional basic attributes and all available attributes on any selected type. These attributes can be used to define more specific search criteria. The additional basic attributes that will be displayed can be configured using the following property setting in `emxSystem.properties`:
`emxFramework.Search.AdditionalBasicAttributes=owner,description,policy,originated,modified`
 The setting can contain one or more comma separated values from the following table.

Additional Basic Attributes	Remarks
Owner	Free form text box for now. The user can type in the user's full name, which is matched to the "full name" attribute in the Person Admin object. Note: This will match the actual user name(id) of the logged in user.
Policy	There will be a range selection for the policy for any selected types.
Originated	Date field
Modified	Date field
Description	Description

- **Control what happens to the search results window when a user clicks the name of an object in the results window**—The search and search result pages display in a popup window on top of the main window. When a user clicks the name of an object in the results list, details about the object appear in the Content frame within the main

application window. Use this property to control what happens to the search popup window after the user clicks an object's name. The default value is **Show Content**, which means the search popup window remains open but moves behind the application window so the user can see the object details. To have the search popup remain open and on top of the main window, enter **Show Search** for the property's value. To close the search popup window when a user clicks an object name, enter **Close Window**.

```
emxFramework.Search.OnSelect.Action = Show Content
```

2D/3D Viewer Properties

- **Define formats used for 2D/3D viewers**—Lets you tell the system which formats users should be choose a preferred 2D/3D viewer for. You list these formats by setting the property to a comma-delimited list of Format symbolic names.

```
emxFramework.FormatsWithViewerPreference = format_JT
```

For example, if you want users to be able to open JT and ACAD files in viewers, set this property to `format_JT, format_ACAD`. For more information on viewers, see [Setting Up 2D and 3D Viewers](#) in Chapter 5.

- **Define the file extensions for viewers**—Use this property to define the file extensions the system should use for 2D and 3D viewers. If a user attempts to view a file whose extension is not in this list, the system will not attempt to open the file in a viewer, even if a viewer has been registered and installed. Separate each extension in the list with a comma and no space.

```
emxFramework.ViewerExtensions=jt,dwg
```

- **Define the viewer prefix**—Lets you define the prefix for all viewer URLs so all viewers can be registered the same.

```
emxFramework.Viewer.ServletPreFix= /WEBAPPNAME/servlet/
```

Miscellaneous Properties

- **Set the frequency of UI cache update**—All dynamic user interface components, such as menu, command, table, and inquiry objects, are loaded into the UI cache in application scope when the application pages/programs request the data. The cache is dynamic based on the ComponentAge property. The default is 3600 seconds, which is 1 hour.

```
emxNavigator.UICache.ComponentAge = 3600
```

The cached data for any given admin object or trigger information expires based on this pre-set amount of time (ComponentAge). Once expired, the next request for that specific data is dynamically obtained/cached fresh from the database and the cache life is reset for that object.

The Reload Cache tool is part of the Navigator Toolbar menu and is installed with the framework. This tool is configured using the admin command object

`AEFReloadCacheToolbar`, which is connected to the Toolbar admin menu object. The default access for this command is assigned to the Administration Manager role and therefore only appears when a user of that role logs in.

When the Reload Cache tool is clicked, the entire cache data is reset and all cached data expires. So, any further request to an admin object from UI cache obtains the data fresh from the database and cached again.

The cached data also includes trigger data, such as input arguments, program names, and changes to the state of the eService Trigger Program Parameters objects. When changes occur to this trigger data, the cache must be reloaded to reflect the change.

- **Define the page title if not defined**—Use this property to specify a page title if not defined by a JSP. Some application framesets use this property to fill in a title when not defined.

```
# default Page Title
emxFramework.Common.defaultPageTitle = MatrixOne
```

- **Define the number of pages to track in Page History**—This property defines the maximum number of pages stored on the Page History page, for all users. By default, this value is set to 50. This means that when a user visits the 51st page, that page replaces the first page visited in the list.

```
emxSystem.pageHistory.limit=50
```

- **Require password verification for promotion signatures**—When this property is TRUE, a Password Verification page appears when users apply a signature for lifecycle promotion or demotion. Users must enter the password they use to log in for the signature to be applied. This is only for signatures applied using the AEF Lifecycle and Approval pages. See [Lifecycle Page](#) in Chapter 8. For instructions on using this verification with external authentication (Single Signon), see [Configuring Approval Verification with External Authentication](#) in Chapter 5.

```
# Approval Password Confirmation
emxFramework.LifeCycle.ApprovalPasswordConfirmation=TRUE
```

- **Require password verification for task approvals**—To require route task assignees to enter their password when choosing Approve, Reject, or Abstain for a task, set the following property to true. These approval options are available only for tasks with Approve actions.

```
#FDA
emxFramework.Routes.EnableFDA=false
```

To require users to enter their username in addition to their password, set the following property to true. Set it to false to require the password only.

```
emxFramework.Routes.ShowUserNameForFDA=true
```

To specify the number of incorrect attempts to allow when entering the username/ password, use the following property. When a user reaches this limit, the system logs the user out and inactivates the user.

```
emxFramework.Routes.VerificationCount = 3
```

Use the following property to specify the username to notify when a user is inactivated due to incorrect verification attempts. This should be a person who can re-activate the user, if appropriate.

```
emxFramework.Routes.NotifyPeople = Test Everything
```

When using the route task verification with external environment, several additional setup steps are needed. See [Configuring Approval Verification with External Authentication](#) in Chapter 5. Additionally, the three properties listed above are ignored and only the EnableFDA property is used. When external authentication is used with the EnableFDA property set to true, the system requires both the user's username and password. If the user enters incorrect information, the system redisplay the username and password page. There is no verification count limit and the user is not inactivated.

- **Setting URL for SSO (external authentication) when using lifecycle approvals**—See [Configuring Approval Verification with External Authentication](#) in Chapter 5 for information about this property.

```
emxFramework.LifeCycle.ExternalAuthenticationURL
```

- **Define the format for displaying person names**—Use this property to define the format for displaying person names. The applications use the format specified here to populate the Full Name field for person admin objects when a person is added or when a name is changed.

```
emxFramework.FullName.Format = <Last Name>, <First Name>
```

The property accepts these three sub-keys, which are all fields that are filled in when a person is added in a MatrixOne Application: <First Name>, <Last Name>, and <User Name>. For example, you could change the format to <First Name> <Last Name> or <User Name>. The default format is <Last Name>, <First Name>.

The framework includes a conversion utility that lets you populate the Full Name field for person admin objects that don't have a Full Name defined (persons created in versions prior to 10 don't have the Full Name defined). You can also use the utility if you decide to change the full name format. See [Changing the Format to Display Person Names](#) in Appendix C readme for details. Additionally, the framework includes trigger programs that run when the person's name is changed to update the Full Name field. For more information, see [Automation for Persons](#) in Appendix A.

- **Specify language for documents**—Use this property to display a Language drop-down list on all document checkin pages, including the page for checking in a new revision. Users can use the list to specify the language the document is in. The file content language will be used by Verity during searches. This allows Verity to search documents written in multi-byte languages for multi-byte characters.

```
emxFramework.ShowLanguageForFileCheckin=false
```

- **Filter the parameters that are passed on requests for security**—Use this property to filter the parameters that are passed on the request to look for possible characters that a hacker would use to gain access to the database. If you want to perform such a filter, enter a comma separated list of characters. If you do not want to perform this sort of check, do not specify any value.

Use caution when using this property. If an incorrect character is specified, it will render applications unusable.

```
# FilterParameter Pattern
```

```
emxFramework.FilterParameterPattern=
```

- **Turn off export features in Program Central**—Set this property to true to turn off all export to Excel features. Currently, only Program Central uses the property.

```
#Set to true to turn off all exporting across apps that use  
property
```

```
ematrix.common.disableAllExports=false
```

- **Specify the icon to use for the Export to Excel feature in Program Central**

```
#Icon for Export in common header
```

```
ematrix.common.icon.spreadsheet=../common/images/  
buttonContextExcelExport.gif
```

- **History Action setting**—This property specifies special history actions used to do the history parsing for the common history component. Do not change the value for this property.

```
#Setting to pick up History Special Actions
```



```
emxFramework.History.SpecialActionType = change owner,change  
policy,change type,change name,change vault,modify form
```

- **Define the object types for which access grants should be made when the objects are added to folders**—This property applies to objects added to Team Central workspace folders and Program Central project folders. The schema name for folder object is Workspace Vault. Typically, in Team Central, when any object type is added to a folder, people with access to the folder should be granted access to the object. However in Program Central, only grants for document types and their sub-types are made. This property lets you specify the types for which grants should be made. The default value is shown below. To add types, enter the symbolic type names separated by commas. Note that for the accesses to be granted for a type, the type's policy must have the Access Grantor role with the appropriate permissions. The property also accepts All, which means grants will be made for all types added to folders (again, the policy must support granting with the Access Grantor role).

```
emxFramework.FolderContentTypesThatTypesRequireGrants=type_Docu  
ment,type_Part,type_RequestForQuote,type_Quotation,type_Package
```

This property works in conjunction with the above property. When set to true, the access grants for folders apply to any sub-type of a type defined in the above property. So any sub-type defined for the Document type will be included in the access grants. When set to false, the grants are made only for the specific types listed in the above property.

```
emxFramework.IncludeSubTypesForGrants=true
```

For more specific instructions on common configuration options for each application, see the Administrator's Guide for the application.

- **Specify calendar years before**—This property sets the number years before the current year to be included in the Year drop-down list that is displayed when a user clicks the calendar icon. This, used in conjunction with the `emxFramework.CalendarDisplay.YearsAfter` `emxFramework.CalendarDisplay.CustomYears` properties, defines the Year list for the calendar drop-down. The default is 2.

```
emxFramework.CalendarDisplay.YearsBefore = 2
```

- **Specify calendar years after**—Specify the number years after the current year to be included in the Year drop-down list that is displayed when a user clicks the calendar icon. This, used in conjunction with the `emxFramework.CalendarDisplay.YearsBefore` and `emxFramework.CalendarDisplay.CustomYears` properties, defines the Year list for the calendar drop-down. The default is 3.

```
emxFramework.CalendarDisplay.YearsAfter = 3
```

- **Specify custom calendar years**—Specify any years to include in the calendar drop-down not contiguous to the currently defined years. This, used in conjunction with the `emxFramework.CalendarDisplay.YearsBefore` and `emxFramework.CalendarDisplay.YearsAfter` properties, defines the Year list for the calendar drop-down.

```
emxFramework.CalendarDisplay.CustomYears = 1995
```

- **Specify bean mapping files**—List one or more mapping files that will be searched before falling back on the standard search algorithm, which looks at the mapping files associated with the bean packages shipped with the AEF and applications. For details, see the “New Features” section of the *Application Exchange Framework* version 10.5 Readme file.

```
emxFramework.BeanMapping.PreliminarySearchFiles =
customMappingFile.properties,emxIntegrationsBeanMappingFile.pro
perties
```

File Management Properties

- **Specify whether to use the Applet for checkin**—Enabling the Applet allows drag-and-drop file checkin. The default is false.

```
emxFramework.UseApplet = true
```

- **Limit the number of files to upload**—Set a limit to the number of files that can be uploaded using the Applet. If the user specifies a number of files to upload that exceeds this limit, the Applet issues a warning.

```
emxFramework.Applet.MaximumNoFilesLimit=200
```

- **Set the session timeout for checkin/checkout**—This property sets the file checkin/checkout timeout length in seconds. If files don’t complete the upload or download within the number of seconds set here, the process times out. The default, 86400 seconds, is 24 hours.

```
emxFramework.ServerTimeOutInSec = 86400
```

- **Download the Java Plug-in for the Applet** —The Java Plug-In version 1.4 or above is required to run the file upload Applet. This property defines the URL for downloading the Java Plug-in to run the file upload Applet, if the user machine does not have the required 1.4 or above. By default, for Windows it points to an auto-install cab file; for Unix it points to a Java-plug-in download page on the Sun website. For Windows, it is Auto-install, but for Unix(Solaris, HP, AIX) it points to download URL. You should download and install and manually register the plug-in with your browser.

```
# java plug-in to run the file upload Applet
emxFramework.Unix.JavaPlugInURL=http://java.sun.com/j2se/1.4.2/
download.html
emxFramework.Windows.JavaPlugInURL=http://java.sun.com/
products/plugin/autodl/jinstall-1_4-windows-i586.cab
```

Special Character Properties

These properties define the characters that should not be used in fields. Some applications accept some of the special characters listed by default in these properties. These characters can be removed from the properties to allow users to enter them. For a list of the special characters accepted by a particular application, see the Administrator’s Guide.

- **Define characters that cannot be used in description, note and other fields**

```
# Javascript Define Bad Characters for use in description, note
and other fields
```

```
emxFramework.Javascript.BadChars = @ * ? [ ] # $ { }
```

- **Define characters that cannot be used in name fields**

```
# Javascript Define Bad Characters for use in Name fields
```

```
emxFramework.Javascript.NameBadChars = @ , * ? [ ] # $ { } " < >
```

- **Use comma for decimal**—Use this property to have the system accept and use commas for decimals. This is a server-wide preference setting. The value for this property should be the same as defined for MX_DECIMAL_SYMBOL in the ematrix.ini file: MX_DECIMAL_SYMBOL=,
For more information on setting the decimal character for Matrix and Oracle, see the *Matrix PLM Platform Installation Guide*.

```
emxFramework.DecimalSymbol=,
```

- **Display thousandths separator for currency and quantity fields**—Set this property to true to display a thousandths separator for currency and quantity fields (for example, 1,000). If the decimal symbol is set to comma, then the thousandths separator is a period. Otherwise, the separator is a comma.
The JSP that displays the field must make use of this property to display the separator. To see if a particular application/page uses it, check the application's Administrator's Guide.

```
emxFramework.DigitSeparator = false
```

Email Notification Properties

The MatrixOne applications notify users by IconMail and email (if an email address is entered for the person and email is enabled in the person's definition) about important tasks and information. For example, Sourcing Central notifies suppliers when a buyer sends them a request for a quotation. Team Central notifies users when they have an active task and when an event occurs that the person has subscribed to. Users can read these notifications from within the MatrixOne Application by clicking IconMail or they can read the notifications in their email application, such as Microsoft Outlook. These properties define how the system notifications should appear.

- **Include URL links to applications in email notifications**—Use this property to include links in the email notifications so the recipients can click the link and go right to the relevant page of the application. If the person isn't logged in, the login page appears first. For more information on configuring external links to the MatrixOne Application, see [Accessing the Applications Externally](#) in Chapter 8.

```
emxFramework.EnableURLNotification = true
```

- **Define the user the notifications should appear to be from**—The value for this property should be the username of the user the system notifications should appear to be from. The system notification IconMail messages display the username of this user in the From field of the messages. The email notifications list the Full Name and Email address defined for the user. If recipients reply to the notifications, the replies are sent to this email address. Make sure the person definition specified for this agent has an appropriate Full Name and Email address.

```
emxFramework.NotificationAgent = VALID_eMATRIX_USERNAME
```

For example, suppose the properties value is set to a user called MGreene, whose Full Name is defined as Meg Greene and the email address is defined as mgreene@host.com. A system notification that a user receives in an email application would look like this:

From: Meg Greene [mailto:mgreene@host.com]
Sent: Thursday, April 11, 2002 4:09 PM
To: VCP User
Subject: 'Folder Created'

Business Object: Workspace Design Team 0000000004

'Folder Created' on 'Workspace' 'Design Team' '0000000004' at
04-11-2002 04:22 PM

[http://host:7001/ematrix/common/
emxNavigator.jsp?objectId=34400.35078.64661.42476](http://host:7001/ematrix/common/emxNavigator.jsp?objectId=34400.35078.64661.42476)

- **Specify default languages for system notifications**—Use this property to specify the languages system-generated notifications should be in. For details, see *Internationalizing System-Generated Notifications* in Chapter 6.

emxFramework.NotificationLanguages = en

- **Define languages for user-specific language preferences**—Use the following properties to control which languages are listed in the System Generated Mail Preference drop-down list on the Person profile page. This drop-down may not be available for all applications. For details, see *Internationalizing System-Generated Notifications* in Chapter 6.

#Property to set the Icon Mail Language Preference for a person
emxFramework.IconMailLanguagePreference=English,German,French,I
talian,Japanese,Korean

emxFramework.IconMailLanguageLocaleEnglish=en

emxFramework.IconMailLanguageLocaleGerman=de

emxFramework.IconMailLanguageLocaleFrench=fr

emxFramework.IconMailLanguageLocaleItalian=it

emxFramework.IconMailLanguageLocaleJapanese=ja

emxFramework.IconMailLanguageLocaleKorean=ko

IconMail Properties

- **Define search for recipients**—These two properties define the Search page that appears when a person clicks the Browse button to specify a recipient or a copied recipient (To or Cc fields). To have the page show Search In options for selecting companies to search in, set the first property to True. To define which options to include, use the second property.

iconmail defaults for recipient search

emxFramework.IconMail.ShowSelects = true

emxFramework.IconMail.FindSelection = My

Company,Customers,Suppliers,Collaboration Partners

Initialization Properties

When first loaded, emxNavigator.jsp loads an initialization file that contains common initialization programs for all applications. This file is specified in the property:

```
eServiceSuiteFramework.InitializationFile =  
emxCommonAppInitialization.jsp
```

This common initialization file, emxCommonAppInitialization.jsp, is located in ematrix/common and any common initialization programs should be included in this file.

Then emxNavigator.jsp loads the properties file for each installed application and looks for the following property:

```
SUITEKEY.InitializationFile = JSP_INITIALIZATION_FILE
```

For example, for Team Central, the Navigator page loads emxTeamCentral.properties and looks for the property eServiceSuiteTeamCentral.InitializationFile. If a valid JSP is specified, it loads that file.

All the property files for the installed applications will be loaded in the application scope with the alias name specified in emxSystem.properties file. The property used is:

```
SUITEKEY.PropertyFileAlias = NAME_OF_APPLICATION_LEVEL_ATTRIBUTE
```

For example, for Team Central, the property in emxSystem.properties is:

```
eServiceSuiteTeamCentral.PropertyFileAlias =  
emxTeamCentralProperties
```

The property name of the alias is also set in “application” for every application using the same property mentioned above. For Team, it will be as below.

```
application.setAttribute(“eServiceSuiteTeamCentral.PropertyFileAlias”,  
“emxTeamCentralProperties”);
```

All applications should use the alias name defined above and should use the following method to avoid duplicate instances of the same property object in “application” scope.

```
propertyAlias =  
(String)application.getAttribute("eServiceSuiteTeamCentral.Prop  
ertyFileAlias");
```

Engineering Central Properties

- **Create deleted user on login attempt**—To have the system create a new user when a deleted user attempts to log in, set this property to TRUE.

```
#If user has been deleted, should the user be create when  
attempting to logon.
```

```
eServiceEngineeringCentral.emxCreatePersonBusObject= FALSE
```

- **Specify vault to create deleted user in**—If the previous property is TRUE, use this property to specify the vault to create the deleted users in.

```
#the default vault to create a person object if a user is  
deleted
```

```
eServiceEngineeringCentral.emxSafetyVault= eService Production
```

Date/Time Format Properties

The below table lists the properties in `emxSystem.properties` that affect how date/times are sent to the database and displayed to users. To see how these properties and Matrix settings affect the way date/times are read, stored, and displayed in Matrix and framework applications, see [Time/date format entry, storage, and display in Matrix and AEF apps](#).

The most important points to keep in mind when configuring date/time format properties are:

- The only settings that affect the display of dates and times in framework-based applications, such as the MatrixOne applications, are the `emxFramework.DateTime.DisplayFormat` and `emxFramework.DateTime.DisplayTime` properties and the browser's language/region setting.
- The `eServiceSuites.eMatrixDateFormat` and `eServiceSuites.eMatrixInputDateFormat` system properties have no affect on how dates are displayed in framework-based applications. These properties must match Matrix date/time properties as described in [Matrix date formats and the matching framework properties](#).
- The **only reason** to change the Matrix NORMAL and TERSE settings is to change the date/time display and accepted entry formats in Matrix Navigator for the desktop or Web.

Property and Default Value	Description and Accepted Values
<code>emxFramework.DateTime.DisplayFormat = MEDIUM</code>	Use this property to determine the length of the time/date display for all users. • SHORT (e.g. 12.13.52 3:30 pm) The date fields for some MatrixOne Application pages contain logic that compares the date the user enters with another date. For example, the task definition pages compare the task due date with the current date. If the <code>DisplayFormat</code> is set to SHORT, then only dates between 1950 and 2049 can be entered and compared correctly on such pages. If you think users will need to enter dates outside of this range, choose the MEDIUM format. • MEDIUM (e.g. Dec 12, 1952) • LONG (e.g. December 12, 1952) • FULL (e.g. Tuesday, December 12 1952 AD 3:30:42 pm PST) Note that the actual display format is determined by the browser language setting. For example, if <code>emxFramework.DateTime.DisplayFormat = MEDIUM</code> on a browser set to en-us, the dates will display as "Jan 12, 1952". If the browser is set to a European language, the dates display as "12 Jan 1952".
<code>emxFramework.DateTime.DisplayTime = false</code>	When this property is set to false, the date display does not include the time and when set to true, the date display includes the time. Coding within the JSP can override this setting.

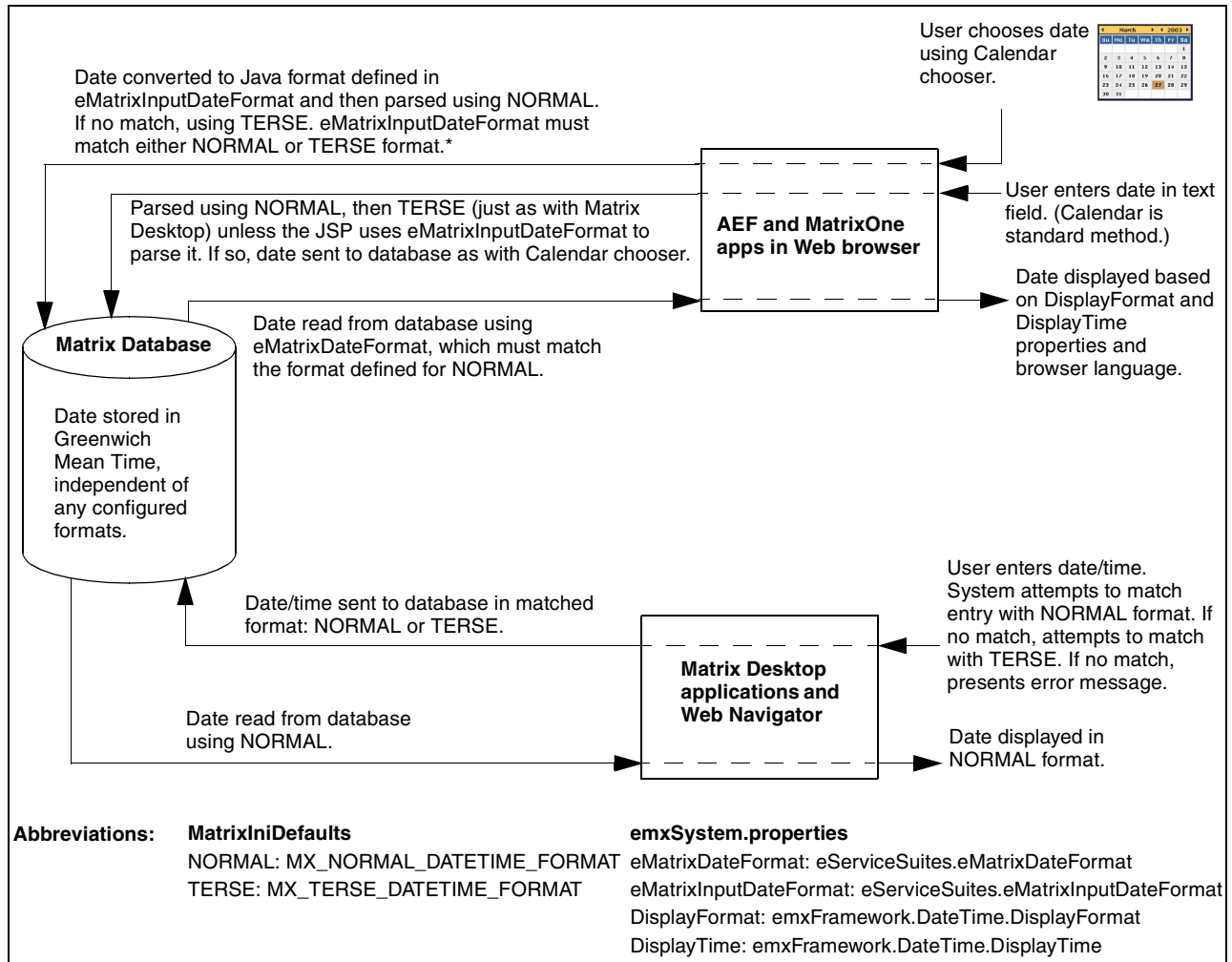
Property and Default Value	Description and Accepted Values
eServiceSuites.eMatrixDateFormat = MM/dd/yyyy hh:mm:ss a	Specifies the Java date format the system should use to read dates from the database. To display a date, the system converts this format to the display format defined by the DisplayFormat and DisplayTime properties and by the browser's language/region setting. This property does not affect the display of dates in framework-based applications. Must be the same format defined in MX_NORMAL_DATETIME_FORMAT. By default, it is the same format. For information on how and why to make changes, see Matrix date formats and the matching framework properties. <i>If you change the date/time format in the eMatrixDateFormat or eMatrixInputDateFormat properties, make sure there are no trailing spaces at the end of the format. For example, if you remove the "a" at the end of the default format, remove the space that precedes it also.</i>
eServiceSuites.eMatrixInputDateFormat = MM/dd/yyyy hh:mm:ss a	Specifies the Java date format the system should use when sending dates to the database. When the user chooses a date/time using the Calendar chooser (the standard UI mechanism for entering dates in MatrixOne applications), the system converts this localized date to the format defined in this property. The system then attempts to parse the date/time using the format defined in NORMAL. If the format doesn't match NORMAL, the system parses the format using TERSE. Must be the same format defined in either MX_NORMAL_DATETIME_FORMAT or MX_TERSE_DATETIME_FORMAT. By default, it is the same format as NORMAL. For information on how and why to make changes, see Matrix date formats and the matching framework properties.

To configure how date/times are displayed in framework-based applications

1. Define the length of the format and whether the time is included using these properties in emxSystem.properties:
emxFramework.DateTime.DisplayFormat
emxFramework.DateTime.DisplayTime
For descriptions of the properties, see [Date/Time Format Properties](#).
2. Make sure users' browsers are set to the correct language and region (for example, en-gb for English UK). The region determines the actual format used, such as whether the month or day is displayed first.

Time/date format entry, storage, and display in Matrix and AEF apps

This graphic shows which settings affect the input and display of date/times in framework-based applications and the relationship between these settings and Matrix settings. The default settings work and should only need to be changed under specific conditions listed in [Matrix date formats and the matching framework properties](#).



* Most date fields require the user to choose the date using the Calendar chooser and do not require or allow a time entry. If the user's input does not include the time but the Matrix date format that eMatrixInputDateFormat matches, either NORMAL or TERSE, does include the time, the framework enters midday (12:00 noon) for the time. If the matching Matrix date format does not include the time, the date passes to Matrix without a time and Matrix adds it. Typically, only a TERSE format would be defined without a time.

Matrix date formats and the matching framework properties

The eMatrixDateFormat and eMatrixInputDateFormat system properties have no affect on how dates are displayed in framework-based applications. As described in the above sections, these properties define the Java format to use to send dates to Matrix and to read

them from Matrix. Therefore, they must define the same formats that are defined for dates in Matrix, as described in this table.

emxSystem.properties property and default value	Must match this setting in the MatrixIniDefaults program	How default values match:
eServiceSuites.eMatrixDateFormat = MM/dd/yyyy hh:mm:ss a	MX_NORMAL_DATETIME_FORMAT	When the framework installs, the property matches the NORMAL format, which is: MX_NORMAL_DATETIME_FORMAT=moy/dom/yr4 h12:min:sec mer
eServiceSuites.eMatrixInputDateFormat = MM/dd/yyyy hh:mm:ss a	either MX_NORMAL_DATETIME_FORMAT or MX_TERSE_DATETIME_FORMAT	When the framework installs, the property matches NORMAL.

Reasons for changing the framework properties

When installed, the framework date/time properties and the Matrix date/time settings in the MatrixIniDefault program are in sync. You should only need to change the framework date/time properties if one of the following conditions apply.

- You change the Matrix NORMAL format.
You should only need to change the NORMAL format if you are using Matrix Navigator for the desktop or Web and want a different input and display format.
Another reason to change NORMAL is if you have a date text field in a framework-based application that does not use the eMatrixDateFormat class to parse the date *and* you want a format other than the default format to be accepted for input. In such a case, the system will parse the entered date using NORMAL and then TERSE just as it does in Matrix Navigator.
If you make a change to the NORMAL format, you must also change the eMatrixDateFormat and eMatrixInputDateFormat properties in emxSystem.properties so they are in sync with the format. (If eMatrixInputDateFormat has been changed to match TERSE, there is no need to change it when NORMAL changes.)
- You decide to match eMatrixInputDateFormat with TERSE instead of NORMAL.
One reason to map eMatrixInputDateFormat to TERSE is if your application has freeform date entry fields that use the eMatrixDateFormat class and you want it to accept the TERSE format instead of NORMAL. The MatrixOne applications typically use the Calendar chooser instead of freeform text fields.
Note that you may have reason to change the TERSE format, for example, to provide an alternate format that will be accepted as input. You can make this change without changing the eMatrixInputDateFormat as long as eMatrixInputDateFormat matches the NORMAL format. By default, the TERSE setting is moy/dom/yr4.

Changing the Matrix date/time settings and the framework properties

Make any changes to the Matrix NORMAL and TERSE settings in the MatrixIniDefaults program and not in the ini file or UNIX startup scripts. Since the MatrixIniDefaults program is in the database, it is shared by *all* users of the system, regardless of the ini file

they are referencing on their desktop or to which server location they connect. If the ini file contains the same setting as MatrixIniDefaults, the ini setting overrides the MatrixIniDefaults program setting.

If you install the framework and the database already contains a MatrixIniDefaults program, the installation copies the existing program to a program called MatrixIniDefaults_VERSION, for example MatrixIniDefaults_10-0-0-0, before it installs its own copy of the program. You should compare the copy with the new version and copy over any settings that you had in the existing program. If the MatrixIniDefaults_VERSION program has date/time format settings that are different from the newly installed MatrixIniDefaults program, make sure you follow the guidelines described before when copying over the changes to MatrixIniDefaults.

The eServiceSuites.eMatrixInputDateFormat and eServiceSuites.eMatrixDateFormat properties require that the date/time format be specified in Java syntax. For a list of the symbols and meanings, see [Java date/time syntax](#).

If you change the date/time format in the eMatrixDateFormat or eMatrixInputDateFormat properties, make sure there are no trailing spaces at the end of the format. For example, if you remove the “a” at the end of the default format, remove the space that precedes it also.

When defining the Matrix date formats, make sure they use numerals for the month and day and use the four-digit year format. Specific operating systems may also have constraints. For example, operating systems for multi-byte languages need the 24 hour time format. The following table shows examples acceptable Matrix date/time settings and the corresponding emxSystem.properties settings.

MatrixIniDefaults program, MX_NORMAL_DATETIME_FORMAT MX_TERSE_DATETIME_FORMAT	emxSystem.properties, eServiceSuites.eMatrixDateFormat eServiceSuites.eMatrixInputDateFormat
Moy/dom/yr4 h12:min:sec +mer	MM/dd/yyyy hh:mm:ss a
Moy/dom/yr4 h24:min:sec	MM/dd/yyyy HH:mm:ss
Dom/moy/yr4 h12:min:sec +mer	dd/MM/yyyy hh:mm:ss a
Dom/moy/yr4 h24:min:sec	dd/MM/yyyy HH:mm:ss
moy/dom/yr4	MM/dd/yyyy
dom/moy/yr4	dd/MM/yyyy

Java date/time syntax

The symbols and meanings for this syntax are listed below. The format specified for the property when the framework is installed is MM/dd/yyyy hh:mm:ss a, which results in a time/date format such as 02/15/2002 10:15:45 am.

To specify the time format, use a time pattern string. In this pattern, all ASCII letters are reserved as pattern letters, which are defined as the following:

Symbol	Meaning	Presentation	Example
G	era designator	(Text)	AD
y	year	(Number)	1996
M	month in year	(Text & Number)	July & 07

Symbol	Meaning	Presentation	Example
d	day in month	(Number)	10
h	hour in am/pm (1~12)	(Number)	12
H	hour in day (0~23)	(Number)	0
m	minute in hour	(Number)	30
s	second in minute	(Number)	55
S	millisecond	(Number)	978
E	day in week	(Text)	Tuesday
D	day in year	(Number)	189
F	day of week in month	(Number)	2 (2nd Wed in July)
w	week in year	(Number)	27
W	week in month	(Number)	2
a	am/pm marker	(Text)	PM
k	hour in day (1~24)	(Number)	24
K	hour in am/pm (0~11)	(Number)	0
z	time zone	(Text)	Pacific Standard Time
'	escape for text	(Delimiter)	
"	single quote	(Literal)	'

The count of pattern letters determine the format.

(Text): 4 or more pattern letters—use full form, < 4--use short or abbreviated form if one exists.

(Number): the minimum number of digits. Shorter numbers are zero-padded to this amount. Year is handled specially; that is, if the count of 'y' is 2, the Year will be truncated to 2 digits.

(Text & Number): 3 or over, use text, otherwise use number.

Any characters in the pattern that are not in the ranges of ['a'-'z'] and ['A'..'Z'] will be treated as quoted text. For instance, characters like ':', '.', ',', '#', and '@' will appear in the resulting time text even they are not embraced within single quotes.

A pattern containing any invalid pattern letter will result in a thrown exception during formatting or parsing.

Framework Directories and Resource File Properties

- **Define the framework directories**

```
eServiceSuiteFramework.Directory = common
eServiceSuiteFramework.RootDirectory = ..
eServiceSuiteFramework.CommonDirectory = common
eServiceSuiteFramework.ComponentDirectory = component
```

- **Define the framework string resource file**

```
eServiceSuiteFramework.StringResourceFileId =
emxFrameworkStringResource
```

Start Page, Home Page, and Banner Properties

- **Define toolbar labels**—This property lets you define the maximum number of characters to be displayed in toolbar labels.
`emxFramework.Toolbar.MaximumLength = 15`
- **Define channel tab labels**—This property lets you define the maximum number of characters to be displayed in a PowerView channel tab label.
`emxFramework.PowerView.Channel.Label.MaximumLength = 17`
- **Define the URL for Dynamic UI applications**—This property specifies the page to call when a user chooses an application that is based on the dynamic user interface. This should be `emxNavigator.jsp`, prefixed with the directory under `ematrix`. You can configure how the Navigator page looks when a user logs in by appending parameters to the JSP. For information, see [Configuring the Main Navigator Page](#) in Chapter 8. The `ApplicationStartPage` property is also being used by some programs to call the `emxNavigator.jsp` page.
`eServiceSuiteFramework.URL = common/emxNavigator.jsp`
`eServiceSuiteFramework.ApplicationStartPage = emxNavigator.jsp`
- **Define default home page**—These properties let you define the page that appears when a user first logs in. You can define the default home page per role. Below are a few examples. For more information on defining the default behavior of the Navigator page, see [Configuring the Main Navigator Page](#) in Chapter 8.
`# emxNavigator properties`
`emxNavigator.Home.ContentPage = emxBlank.jsp`
`emxNavigator.Home.ContentPage.BuyerAdministrator = ../`
`suppliercentralsourcing/`
`emxQuoteRTSTemplateSummaryFS.jsp?suiteKey=eServiceSuiteSupplier`
`CentralSourcing`
`emxNavigator.Home.ContentPage.Supplier = ../`
`suppliercentralsourcing/`
- **Define the images in the banner (header)**—Use these properties to configure the banner images.
`emxNavigator.UIHeader.LeftImage = utilNavigatorHeaderBG.gif`
`emxNavigator.UIHeader.RightImage = logoMatrixOneVCP.gif`
- **Define the URL for the PLM link on the default Home page**—The default Home page contains a link that lets users get more information about Product Lifecycle Management (PLM). By default, this link points to the MatrixOne Website. Use the following property to configure the URL the link points to.
`emxNavigator.Home.PLMLink = http://www.matrixone.com`

Configurable Tree (Context Navigator/Structure Navigator) Properties

- **Control whether the Structure Navigator opens automatically when structured object is selected**—When this property is true, the Structure Navigator automatically displays when a user accesses an object that has a hierarchical structure defined, such as a workspace in Team Central or a project in Program Central. When false, the user must click the Show/Hide Structure button to show the Structure Navigator.
`#determines whether or not the open structures by default or not`
`emxFramework.UIStructureTree.ShowUponOpen=true`

- **Define the icon image to use for a type**—Specify the image file to use in the root category of a type's navigation tree and in table column headings that display an icon. For example, this property defines the icon for a mechanical part:

```
emxFramework.smallIcon.type_MechanicalPart =
iconSmallMechanicalPart.gif
```

- **Define the default image file to use for root tree nodes and table column headings**—If there is no `emxFramework.smallIcon` property defined for the type's icon, the system looks for a property for the type's parent type icon, then grandparent's type and so on. If no property is defined for any parent type's icon, the system uses the default image, defined by this property:

```
# default icon name for the type
emxFramework.smallIcon.defaultType = iconSmallDefault.gif
```

One exception to the above is that if the system uses a parent's tree menu because no tree menu is defined for the subtype, then the icon associated with the parent type is used instead of the subtype.

- **Configure the default navigation tree**—Use these properties to specify the menu admin object that represents the default tree. For more information on the default tree, see [Using and Configuring the Default Tree](#) in Chapter 8.

```
#Setting to pick up default tree
emxNavigator.DefaultTree.TreeName = menu_DefaultTree
```

- **Make the version 10 and higher navigation tree work with version 955x applications**—If you are using a version 955x application with version 10 or higher framework, set the following property to true.

```
emxFramework.UITree.PreV10AppsSupport = false
```

Configurable Table Properties

- **Set pagination range**—Use this property to define the number of rows in a table per table page. For example, if you set the property to 15, the system creates a new page for the 16th row and users must use the pagination controls to see the 16th row.

```
#Pagination Range value
emxFramework.PaginationRange = 10
```

- **Define the style sheets to use for table pages**—These two properties define the style sheets to use for table pages. The List style is for table pages that display in the content frame of `emxNavigator.jsp`. The Dialog style is for table pages that display in popup windows. The Style parameter, which can be passed to `emxTable.jsp`, determines whether the table pages uses the List or Dialog style sheet. For information on the Style parameter, see [URL Parameters Accepted by emxTree.jsp](#) in Chapter 8. For information on the default style sheets, see [emxUISearch.css](#) and [emxUIList.css](#) in Chapter 8.

```
#Setting to define style sheet in table component
emxNavigator.UITable.Style.List = styles/emxUIList.css
emxNavigator.UITable.Style.Dialog = styles/emxUISearch.css
```

- **Specify the default for header repeat**—This property defines the default value for how often header rows repeat in configurable tables. Headers in configurable tables will repeat based on this value unless the `headerRepeat` parameter is passed with a different value to `emxTable.jsp`. To turn off header repeat as the default, set this property to 0.

```
emxNavigator.UITable.HeaderRepeat = 15
```

- **Remember table page selections across pages**—When this setting is true, the system remembers items selected across different pages in a paginated table (only table pages built using `emxTable.jsp`). If the user selects rows from several different pages within a table and then clicks a toolbar item to process the items, such as Remove Selected, the system processes selected items from all pages, not just the one that is currently displayed. Remembering selections across pages can be turned off global using an `emxSystem.properties` property and turned on and off per table page using the `rememberSelection` parameter for `emxTable.jsp`.
#Setting for the turning on / off the selection being remembered across pages
`emxNavigator.UITable.Pagination.RememberSelection = true`
- **Turn on the performance time logging**—This property can be assigned to any file. For example, `emxFramework.Timing.LogFile = /temp/emxFrameworkTiming.log`. If the key is assigned to empty there will be no logging. To enable logging, assign a file name. Currently the time logging is implemented only on the configurable table component. This is for internal testing purposes and should not be turned on in a production environment.
`emxFramework.Timing.LogFile =`
- **Control setting for in-cell editing mass updates.** Users can change multiple records with a single operation with the controls located in the header frame of the In-Cell Editing popup window. If this property is set to `true` then the `massUpdate` section in the TableEdit Header is shown in all cases. This is the default. It can be overridden by passing the parameter `massUpdate=false` to `emxTableEdit.jsp`.
`emxFramework.ShowMassUpdate = true`

Configurable Form Properties

- **Specify a JavaScript file with custom method for validating form fields**—For more information, see [User Defined / Custom JavaScript Validation](#) in Chapter 8. The name of the file must be specified in `emxSystem.properties` file with the key `SUITEKEY.UIForm.ValidationFile`. The value can be assigned to any JavaScript file or any JSP file. For example:
`eServiceSuiteSpecificationCentral.UIForm.ValidationFile = emxSpecFormValidation.js`
`eServiceSuiteSpecificationCentral.UIForm.ValidationFile = emxSpecTSValidation.jsp`
`eServiceSuiteProductCentral.UIForm.ValidationFile = <JS file name>`

Application Properties

- **eServiceSuiteSUITE_NAME.Directory**—The name of the directory where files for the application are located. This is typically `Apps/Framework/VERSION/APPLICATIONNAME`.
- **eServiceSUITE_NAME.ApplicationPropertyFile**—The name of the property file for the application. By default, this file is named `emxAPPLICATIONNAME.properties`. All properties files are located in `Apps/Framework/VERSION/UserInterface`.

- **eServiceSUITE_NAME.ApplicationStartPage**—The first displayed page for the application. This page is typically called after the application home page is invoked and loads the application properties file. This page must be within the directory specified in the directory property.

Configuring the User Interface

Overview of Dynamic User Interface

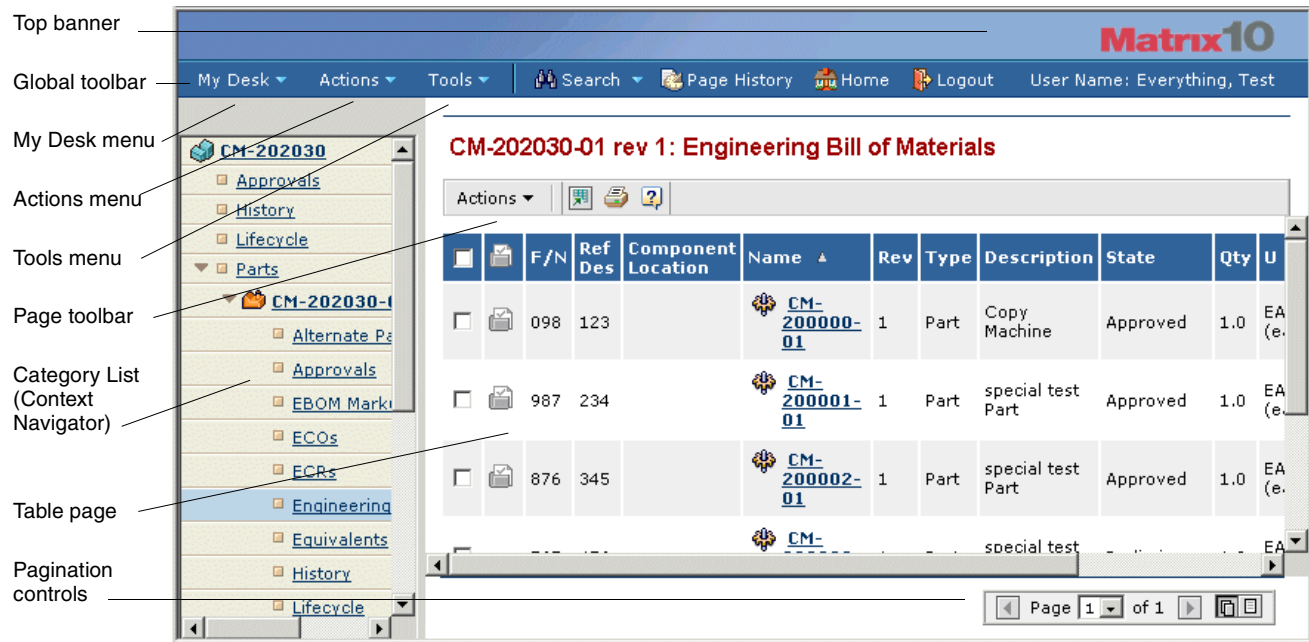
The topics in this section describe how to build an application using the Framework's Web-based dynamic user interface (UI). The dynamic user interface lets you:

- Configure existing applications that are built using the dynamic user interface.
- Add menus and features to existing applications.
- Create a custom application that contains the interface.

Using the dynamic user interface, you accomplish all this by modifying administration objects instead of changing code in JavaServer Pages or JavaBeans.

Components of the Dynamic User Interface

This graphic shows the main elements within the Framework's dynamic user interface. Each component is described in subsequent sections.



Naming Conventions for UI Administrative Objects

The following table lists naming conventions used for UI objects installed with the framework. The naming convention for most objects includes a three-letter abbreviation for the application that uses the object (for example, TMC for Team Central). These abbreviations are listed in the table on the following page. If you create custom objects, create your own abbreviation based on your custom application name. This applies to anyone creating custom objects, including customers, partners, and MatrixOne Professional Services. The only objects that do not use the three-letter abbreviation are the top-level menu objects and the menu objects for the root node of trees. Menus for tree nodes use the symbolic name of the object type (type_Part) and everyone should use these objects. Do not include spaces in the names for UI administrative objects.

Admin Object Type, Usage	Convention	Examples
Menu objects for menu types (installed with framework)	Menu type name	My Desk Actions ToolBar
Menu objects for application submenus within a menu type (installed with applications)	3-letter standard abbreviation for application name (see table below), followed by the menu type with no spaces.	ENCMYDesk TMCActions
Menu objects for root node of trees	Symbolic name of the type the tree is for or the type's parent type.	type_Part type_DrawingPrint
Menu objects for toolbars	3-letter standard abbreviation for application name (see table below), followed by the name of the action or feature the toolbar is used for, followed by "ToolBar", all with no spaces.	ECBOMListToolBar TMCProjectsToolBar APPRouteSummaryToolBar SPCTechSpecActionsToolBar

Admin Object Type, Usage	Convention	Examples
Portal objects for object PowerView	3-letter standard abbreviation for application name (see table below), followed by the feature name, followed by “PowerView”.	SCSRFQPowerView
Channel objects	3-letter standard abbreviation for application name (see table below), followed by feature name, followed by “Channel”.	SCSRFQLineItemsChannel
Command objects for channel tabs	3-letter standard abbreviation for application name (see table below), followed by feature name.	SCSRFQLineItemsAttachments
Command objects for toolbar items and tree categories on pages	3-letter standard abbreviation for application name (see table below), followed by the action or feature name, followed by the type of command (Actions, MyDesk, Tree, Toolbar, or TreeCategory).	ENCCreatePartActions TMCEditProfileToolbar AEFLogoutToolbar
Table objects	3-letter standard abbreviation for application name (see table below), followed by the feature name.	ENCBOMList TMCWorkspaces SCSBUYERDesks
Inquiry objects	3-letter standard abbreviation for application name (see table below), followed by the list name or the feature the inquiry is for.	ENCCBOM TMCPersons SCSPackages
Web Form objects	3-letter standard abbreviation for application name (see table below), followed by the object type or feature name.	ENCPart SCSBUYERDesk
Web Form fields	Name of the field	Name Description Part Classification Originated By

Application	3-Letter Abbreviation Used for Objects Installed with Framework (do not use these abbreviations when creating custom objects)
Application Exchange Framework	AEF
Common Components	APP
Configuration Central	CFC
Document Management Central	DMC
Engineering Central	ENC
Product Central	PRC
Program Management Central	PMC

Application	3-Letter Abbreviation Used for Objects Installed with Framework (do not use these abbreviations when creating custom objects)
Sourcing Central	SCS
Specification Central	SPC
Supplier Central	SUP
Team Central	TMC

Internationalizing Dynamic UI Components

You can internationalize the following text within UI components:

- Page header labels
- Table headers and repeat headers
- Alt text on icons and images
- Table page filter list option
- Types selected in the configurable Type Chooser
- Labels for form page fields
- Form field values

You internationalize this text by entering a string resource ID property name as the value for the text. If you enter standard text as the value or a macro, the text cannot be internationalized. To get the value for a specified property name, the system looks up the appropriate resource file. The appropriate resource file depends on the Registered Suite setting and an additional entry in the `emxSystem.properties` for every application.

Example property setting in `emxSystem.properties` for Engineering Central:

```
eServiceSuiteEngineeringCentral.StringResourceFileId =
emxEngineeringCentralStringResource
```

For Team Central:

```
eServiceSuiteTeamCentral.StringResourceFileId =
emxTeamCentralStringResource
```

For information on internationalizing other text within the applications, such as admin type values and online help, see [Chapter 6, Internationalizing the Framework and Applications](#).

Using Macros and Expressions in Dynamic UI Components

Many strings used in the definition of dynamic UI components (such as label values, hrefs, and settings) can contain embedded macros and select clauses. The `{ }` delimiters identify macro names. Macros are evaluated at run-time and before displaying on the UI. New macros for dynamic UI components are available for directory specification. Some existing macros are also supported. See [Directory Macros](#) and [Select Expression Macros](#).

Strings defined using macros and expressions cannot be internationalized. For instructions on internationalizing text for UI components, see [Internationalizing Dynamic UI Components](#).

Some strings can also include select clauses which are evaluated against the appropriate business object at run-time. The `$<>` delimiters identify select clauses. Because the select

clauses will generally use symbolic names, the clauses will be preprocessed to perform any substitutions before submitting for evaluation. The following example shows a macro being used in the href definition and another macro being used in the Image setting, as well as a select clause being used in the label definition of a tree menu (associated with a `MenuItem` object):

```

MQL<2>print menu type_MenuItem;
menu type_MenuItem
description
label '$<attribute[attribute_EnteredName].value>'
href '${SUITE_DIR}/emxQuoteMenuItemDetailsFS.jsp'
setting Image value ${COMMON_DIR}/iconSmallRFQMenuItem.gif
setting Registered Suite value SupplierCentralSourcing
children
command SCSAttachment
command SCSAttributeGroup
command SCSHistory
command SCSSupplierExclusion
command SCSUDA
nohidden
property original name value type_MenuItem
property installed date value 02-28-2002
property installer value MatrixOneEngineering
property version value Verdi-0-0-0
property application value Sourcing
created Thu Feb 28, 2002 11:12:34 AM EST
modified Thu Feb 28, 2002 11:12:34 AM EST

```

The following example shows a typical business object macro being used in the label definition of a tree menu (associated with a `Company` object):

```

MQL<3>print menu type_Company;
menu type_Company
description
label '$<name>'
href '${SUITE_DIR}/emxTeamCompanyDetailsFS.jsp'
setting Image value ${COMMON_DIR}/
iconSmallOrganization.gif
setting Registered Suite value TeamCentral
children
command TMCBusinessUnit
command TMCLocation
command TMCPeople
nohidden
property original name value type_Company
property installed date value 02-28-2002
property installer value MatrixOneEngineering
property version value Verdi-0-0-0
property application value TeamCentral
created Thu Feb 28, 2002 11:31:57 AM EST
modified Thu Feb 28, 2002 11:31:57 AM EST

```

When using macros, surround it with quotes to ensure proper substitution for values that contain spaces.

Directory Macros

The following table provides the list of directory-specific macros that can be used in parameters and settings for dynamic UI components. If no path is specified for a file, the system looks for the file in the registered directory (as specified in emxSystem.properties for the registered application). If there is no registered directory, the system assumes the file is in the current directory, which is common.

Macro Name	Use to have the system look for the file in:
\${COMMON_DIR}	The “common” directory, which is obtained from the emxSystem.properties file. The key used is eServiceSuiteFramework.CommonDirectory = common.
\${ROOT_DIR}	The ematrix root directory, which is obtained from emxSystem.properties using the key eServiceSuiteFramework.RootDirectory = ..
\${SUITE_DIR}	The application-specific directory, which is obtained from emxSystem.properties based on the Registered Suite setting. For example, the key used for “Engineering Central” is eServiceSuiteEngineeringCentral.Directory = engineeringcentral.
\${COMPONENT_DIR}	The directory that contains common components, which is ematrix/component by default. The directory is obtained from eServiceSuiteFramework.ComponentDirectory = component in the emxSystem.properties file.

Select Expression Macros

The select expression macros are defined as \$<SELECT EXPRESSION>, where the select expression can be any valid MQL select statement. Select expression macros can be used in labels for configurable components and in expression parameters. These expressions are evaluated at runtime against the current business object ID and relationship ID that is passed in. Some examples include:

- \$<type>
- \$<name>
- \$<revision>
- \$<attribute[attribute_Originator].value>
- \$<attribute[FindNumber].value>
- \$<from[relationship_EBOM].to.name>

Other Macros

Macro Name	Description
<code>\${TABLE_SELECTED_COUNT}</code>	Substitutes the number of rows selected in the current table, included those selected on other pages within a paginated table. Use in a confirmation message for a toolbar item to let user know how many rows are selected. Define the confirm message using the Confirm Message setting for a toolbar item command.

Controlling User Access to User Interface Components

This section describes how to control who can access menu items, tree categories, table columns, form rows, and toolbar items.

- [Access to an Application](#)
- [Access Checks for Individual Components](#)
- [Access Parameters and Settings](#)
- [Sample JPO for Controlling Access](#)

Access to an Application

The dynamic user interface defines access more finely than the application level so you can't really assign access to an application. Instead, you define access to menu items, toolbar items, tree categories, table columns, and form rows (fields) using the admin object that defines each of these components. The `emxNavigator.jsp` displays the submenu for an application if the user has access to at least one command object in the submenu. If the user doesn't have access to any command objects in the submenu, the submenu heading isn't displayed.

Access Checks for Individual Components

The system provides four levels of access control for toolbar items, menu commands (which includes commands in the My Desk menu, Actions menu, and global toolbar), tree categories, table columns, and form rows. To set these access checks, add the setting or parameter for the appropriate admin object. For example, to control access for a toolbar item, edit the command object that represents the link. For details, see [Access Parameters and Settings](#).

Assuming all possible checks are enabled, the sequence of access checks are as follows:

1. Role-based—Users assigned to the Access tab for the relevant admin object. These users are usually roles but can be persons and groups. When a role or group is assigned access, access extends to all child roles and groups. This access is the default or minimum access control.
2. Access Mask setting on the admin object—Specifies the accesses the user must have for the current business object in order for the component to be displayed.
3. Access Expression setting on the admin object—A Matrix expression whose value is a valid context-based expression that gets evaluated at runtime.
4. Access Program and Access Function settings on the admin object—The name of a JPO program and method that checks access.

The system performs the access checks in an all or nothing manner. That is, if all the access checks pass, access is granted. If at least one of them fails, the UI component is hidden.

Search Page Access

Clicking Search from the global toolbar opens whichever page is defined at the top of the Search menu, `AEFGlobalSearch`. When the AEF is installed, the default is to show the

Saved Searches page for most users, and the General Search page for users defined in the Administration Manager role. Once all applications have been installed, you should rearrange the Search menu so the most often used Search page is at the top of the menu. It is recommended to move the Saved Searches command to be the last item in the Search menu, since Saved Searches will have no items listed until each user creates and then saves their own searches. For details, see [Building Menus and the Global Toolbar](#).

Additionally, there is a URL parameter called defaultSearch that can be assigned to any search command. If this is set, all users with access will get that search page when they click Search from the global toolbar. If any users do not have access, they will get the first search in the list they have access to. For details, see [Search Properties](#) in Chapter 7

Access Parameters and Settings

This table describes the parameter and settings that control access to menu commands, toolbar items, tree categories, table columns, and form rows.

Parameter/Setting	Description	Accepted Values/Examples
Access tab (user MQL command)	Use to specify the persons, roles, and groups who can access the component. When you assign a role or group, all child roles/groups also receive access. To make the component available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access.	Names of group, role, person administrative objects. <i>Or</i> All (default)
Access Expression setting	Controls access to the component based on a valid Matrix expression. The expression gets evaluated at runtime. If the expression evaluates to True and no other access control prevents access, the component is shown. Otherwise it is hidden. Depending on the type of expression defined, the program may or may not need a valid objectId. Administrators need to make sure that an objectId is available before configuring this setting for configurable toolbar menus and commands. When the setting is used for commands connected to the AEFGlobalToolbar (My Desk, Actions, Tools), the expression is evaluated on the person business object. It does not depend on any objectId to be passed in as a URL parameter. In cases where no objectId is passed, such as a table page originating from the My Desk, the Access Mask setting is ignored. Similarly, the IconMail tree, which does not have an objectId, does not use this setting if defined.	- attribute[attribute_Weight].value > 100 The toolbar item displays only if the Weight attribute on the business object is greater than 100. - attribute[attribute_Originator].value == owner For menu command objects: - context.user.name=="Test Everything" - context.user.assignment == "Design Engineer" - context.user.assignment == "\$<role_DesignEngineer>"

Parameter/ Setting	Description	Accepted Values/Examples
Access Function setting	The name of the JPO method to invoke in the JPO specified for the Access Program setting. The Access function gets the input parameter as a HashMap, which contain all the request parameters that were passed into the JSP page. The JPO method must return an object of class type Boolean. If the returned value is true and no other access control prevents access, the component is displayed. If false, it is hidden. To see a sample JPO method to control access, see Sample JPO for Controlling Access. This is particularly useful if there is a criteria for access has nothing to do with users' roles. For example, suppose a link should only be shown to users who are employees of the host company. The JPO and method might check the user's company and display the link only if the user is from the host company.	The name of an access check method in the JPO specified in the Access Program setting, such as: emxAccessCheck()

Parameter/ Setting	Description	Accepted Values/Examples
Access Mask setting	Specifies the accesses the user must have for the current business object in order for the component to be displayed. When the setting is used for commands connected to the AEFGlobalToolbar (My Desk, Actions, Tools), the expression is evaluated on the Person business object. It does not depend on any objectId to be passed in as a URL parameter. When the setting is used for commands in page toolbars, menus, trees, table columns, and form fields, the access mask is evaluated on a specific business object. This business object is available only when the objectId is passed in as a URL parameter to the JSP. Administrators need to make sure that an objectId is available before configuring this setting for configurable toolbar menus and commands. In cases where no objectId is passed, such as a table page originating from My Desk, the Access Mask setting is ignored. Similarly, the IconMail tree, which does not have an objectId, does not use this setting if defined. If the user does not have all the specified accesses in the business object's policy for the current state, the component is hidden. If the user has the access and no other access control prevents access, the component is displayed. You can specify multiple accesses by separating the accesses with a comma.	Any set of accesses, separated by a comma. For example: Modify Delete ToConnect ToDisconnect FromConnect FromDisconnect Modify,Delete FromConnect,FromDisconnect
Access Program setting	Controls access to the UI component based on the output from a method in the specified JPO program. The program must be defined in Matrix. This setting requires that the Access Function setting also be specified. If Access Function is not set, the Access Program setting is ignored. The following input values are required for the program: • A list of all of the request parameters in a HashMap • Method name as a string • JPO Program Name as a string • Context The output must be a Boolean.	Name of a JPO defined as a program object in Matrix, such as: emxAEFCollectionAccess

Sample JPO for Controlling Access

A JPO method can be defined for controlling access to UI components. The following settings are required for the component whose access you want to control:

- Access Program=<JPO Name>
- Access Function=JPO method name

Access Function gets the input parameter as a HashMap, which contains all the request parameters that were passed into the Form page. This Map can be used by the method to obtain any request parameters that are required for processing and deciding the access.

The return type of this method is a Boolean. The value will be true if the component is accessible to the current user context and the given object, if any. Otherwise the value will be false and the component will be hidden.

Here is a template for a JPO function that controls access for a form field:

```
public static Boolean methodName(Context context, String[] args)
throws Exception
{
    // Define a boolean to return
    Boolean bFieldAccess = new Boolean(true);

    HashMap requestMap = (HashMap) JPO.unpackArgs(args);

    // Get the parameter values from "requestMap" - if required
    String objectId = (String) requestMap.get("objectId ");
    String relId = (String) requestMap.get("relId ");
    String languageStr = (String) requestMap.get("languageStr");

    // Do the necessary processing to define access and assign
    true/false to bFieldAccess
    ....
    bFieldAccess = new Boolean(true);
    // (or)
    // bFieldAccess = new Boolean(false);

    return bFieldAccess;
}
```

Configuring Colors, Font Styles, and Images

All images for the dynamic user interface and all style sheets are in ematrix/common.

Configuring the Banner (Header)

The top banner is made up of two graphics. The right image is the MatrixOne logo. The left image is the entire background of the banner.



Use these properties in emxSystem.properties (located in ematrix/common) to configure the images in the banner. Note that the middle image property is not used.

```
emxNavigator.UIHeader.LeftImage=utilmatrix10NavigatorHeader.jpg
emxNavigator.UIHeader.RightImage = logoMatrix10.gif
```

Configuring Styles

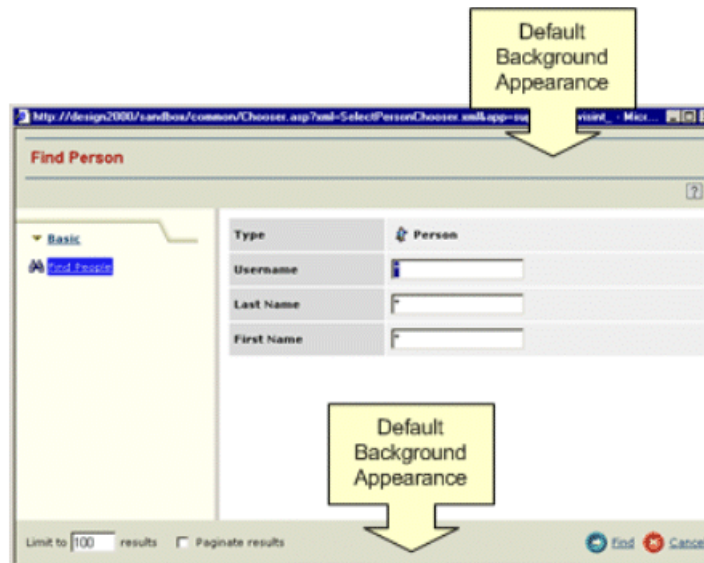
All sheets are in ematrix/common.

Changing a style sheet file will affect ALL pages that use it. Make sure that any changes you make will work on all pages that use the given file. You cannot change the look of an individual page without creating a special style sheet for that purpose.

emxUIChooser.css

To change how choosers appear, edit the style sheet file emxUIChooser.css. The style sheet contains the following style definitions:

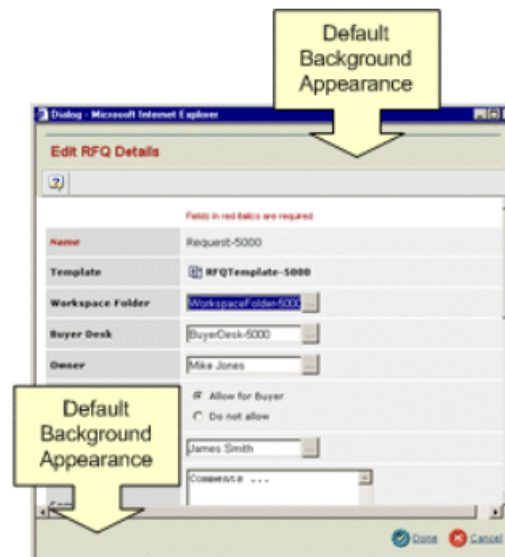
Definition	Name	Categories
body { }	Default Background Appearance	Background



emxUIDialog.css

To change how the dialog's top and bottom frames appear, edit the style sheet file `emxUIDialog.css`. The style sheet contains the following style definitions:

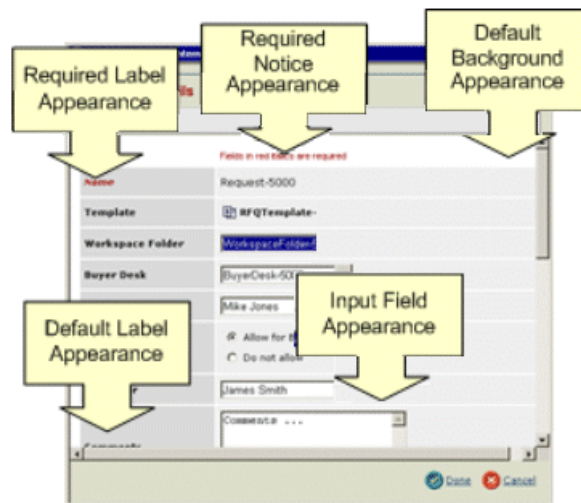
Definition	Name	Categories
<code>body { }</code>	Background Appearance	Background



emxUIForm.css

To change how the dialog's middle frame appears, edit the style sheet file emxUIForm.css. The style sheet contains the following style definitions:

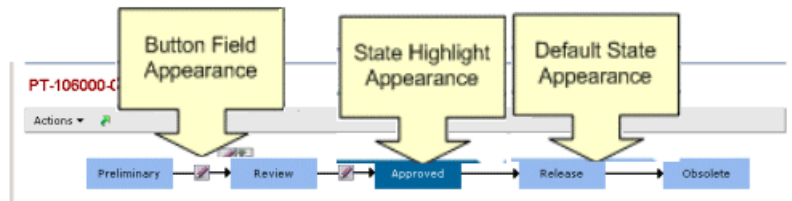
Definition	Name	Categories
body { }	Background Appearance	Background
td.label { }	Default Label Appearance	Background, Font
td.requiredLabel { }	Required Label Appearance	Background, Font
td.inputField { }	Input Field Appearance	Background, Font
td.requiredNotice { }	Required Notice Appearance	Font



emxUILifeCycle.css

To change how the Lifecycle pages appear, edit the style sheet file emxUILifeCycle.css. The style sheet contains the following style definitions:

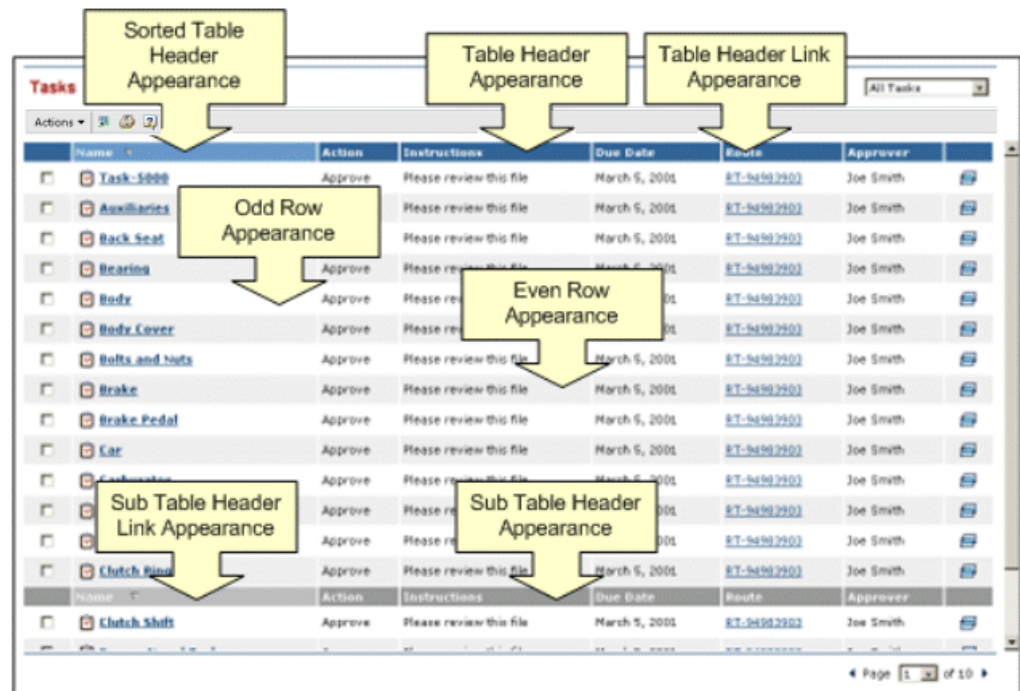
Definition	Name	Categories
td { }	Default Cell Appearance	Position
td.buttonField { }	Button Field Appearance	Background, Position
td.stateName { }	Default State Appearance	Font, Background, Position
td.stateNameHighlight { }	State Highlight Appearance	Background, Font, Position
td.shadow { }	State Shadow Appearance	Background



emxUIList.css

To change how the table pages appear, edit the style sheet file emxUIList.css. The style sheet contains the following style definitions:

Definition	Name	Categories
th { }	Table Header Appearance	Background, Font, Position
th.sorted { }	Sorted Table Header Appearance	Background
th a { }	Table Header Link Appearance	Font
th.sub { }	Sub Table Header Appearance	Background, Font, Position
th.subSorted { }	Sorted Sub Table Header Appearance	Background
tr.odd { }	Odd Table Row Appearance	Background
tr.even { }	Even Table Row Appearance	Background



emxUINavigator.css

To change how the components of the main page appears, edit the style sheet file emxUINavigator.css. The style sheet contains the following style definitions:

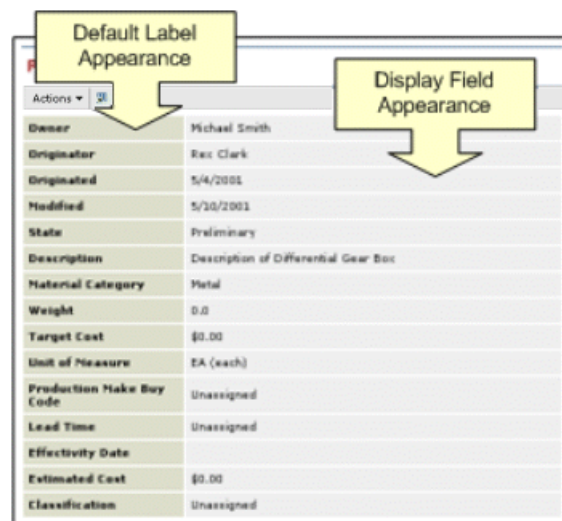
Definition	Name	Categories
body.navigator { }	Body Appearance	Background
#divShrunkLogo { }	Logo Appearance	Position
#divShrunk { }	Header Appearance	Position
#elmUsername { }	Username Appearance	Font, Position
#content { }	Content Window Appearance	Position
.toolbar-row { }	Toolbar Row Appearance	Background, Borders
div.toolbar { }	Toolbar Appearance	Background, Borders, Position
div.toolbar-container { }	Toolbar Container Appearance	Background, Borders
div.toolbar-frame { }	Toolbar Container Frame Appearance	Background, Borders
.icon-button, .text-button,.icon-and-text-button,. overflow-button { }	Button Appearance	Background, Borders, Font
td.menu-arrow { }	Menu Arrow Appearance	Background
td.combo-button { }	Combo Button Appearance	Position
.text-button { }	Text Button Appearance	Position
.overflow-button { }	Overflow Button Appearance	Background
.icon-button { }	Icon Button Appearance	Position
.icon-and-text-button { }	Icon and Text Button Appearance	Background, Position
.icon-button span.down-arrow , .text-button span.down-arrow ,.icon-and-text-button span.down-arrow{ }	Button Arrow Appearance	Background, Position
.button-hover, td.button-hover { }	Button Hover Appearance	Background, Borders
.button-active,.menu-button-active ,td.button-active { }	Button Active Appearance	Background, Borders
.button-disabled{ }	Button Disabled Appearance	Background
.separator{ }	Toolbar Separator Appearance	Borders



emxUIProperties.css

To change how the Properties pages appear, edit the style sheet file emxUIProperties.css. The style sheet contains the following style definitions:

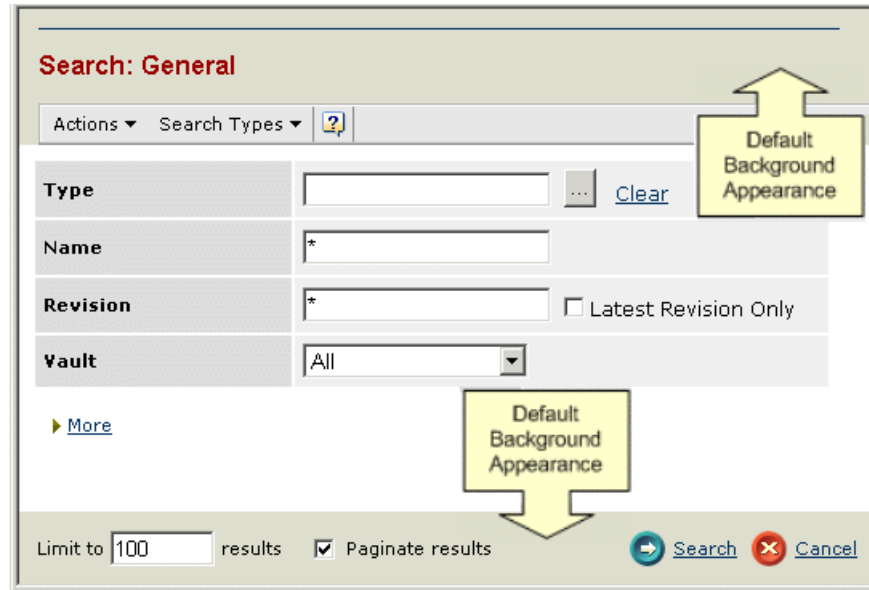
Definition	Name	Categories
td.label { }	Default Label Appearance	Background, Font
td.field { }	Display Field Appearance	Background



emxUISearch.css

To change how the list appears, edit the style sheet file emxUISearch.css. The style sheet contains the following style definitions:

Definition	Name	Categories
body { }	Default Background Appearance	Background

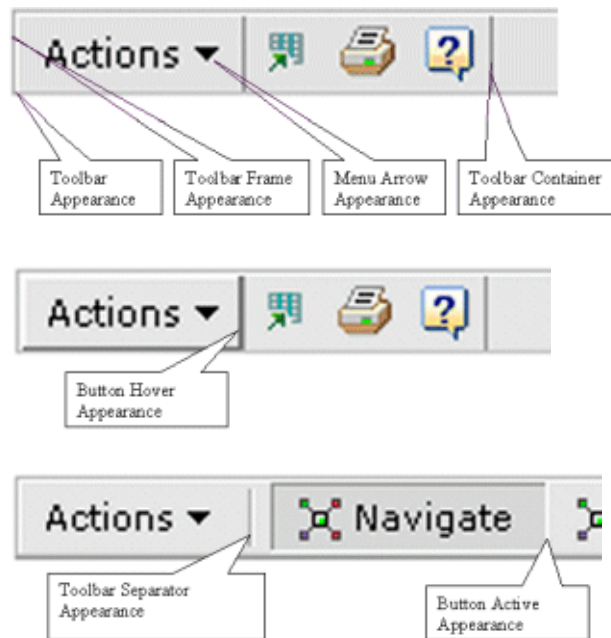


emxUIToolbar.css

To change how the dialog's toolbar appears, edit the style sheet file emxUIToolbar.css. The style sheet contains the following style definitions:

Definition	Name	Categories
div.toolbar { }	Toolbar Appearance	Background, Borders, Position
div.toolbar-container { }	Toolbar Container Appearance	Background, Borders
div.toolbar-frame { }	Toolbar Container Frame Appearance	Background, Borders
.icon-button, .text-button,.icon-and-text-button,.overflow- -button { }	Button Appearance	Background, Borders, Font
td.menu-arrow { }	Menu Arrow Appearance	Background
td.combo-button { }	Combo Button Appearance	Position
.text-button { }	Text Button Appearance	Position
.overflow-button { }	Overflow Button Appearance	Background
.icon-button { }	Icon Button Appearance	Position
.icon-and-text-button { }	Icon and Text Button Appearance	Background, Position

Definition	Name	Categories
.icon-button span.down-arrow , .text-button span.down-arrow , .icon-and-text-button span.down-arrow { }	Button Arrow Appearance	Background, Position
.button-hover, td.button-hover { }	Button Hover Appearance	Background, Borders
.button-active, .menu-button-active, td.button-active { }	Button Active Appearance	Background, Borders
.button-disabled { }	Button Disabled Appearance	Background
.separator { }	Toolbar Separator Appearance	Borders



emxUITree.css

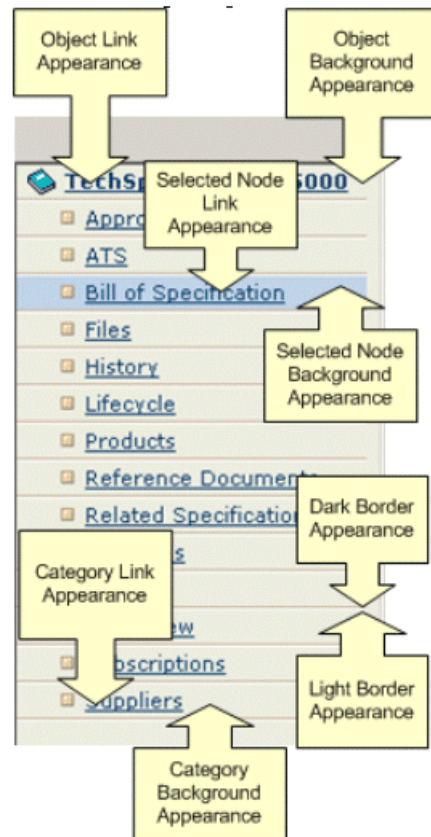
Avoid increasing the font sizes for text in the navigation trees because it will likely break the images or make the tree look distorted.

To change how the navigation trees appear, edit the style sheet file emxUITree.css. The style sheet contains the following style definitions:

Definition	Name	Categories
body { }	Background Appearance	Background
* { }	Default Font Appearance	Font
a { }	Default Link Appearance	Font
td.object { }	Object Background Appearance	Background
td.object a { }	Object Link Appearance	Font
td.category { }	Category Background Appearance	Background

Definition	Name	Categories
td.category a { }	Category Link Appearance	Font
tr.selected { }	Selected Node Background Appearance	Background
tr.selected a { }	Selected Node Link Appearance	Font
tr.lightBorder { }	Light Border Appearance	Background
tr.darkBorder { }	Dark Border Appearance	Background

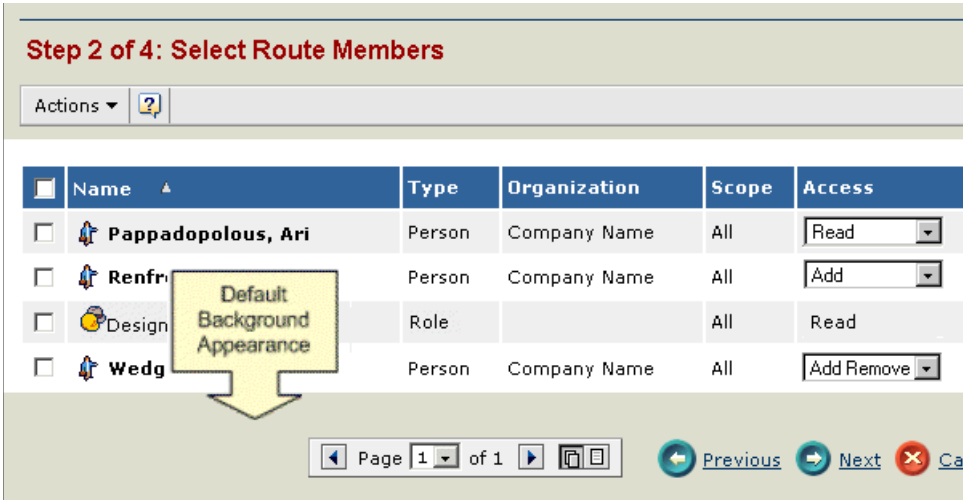
The following image shows how each CSS style affects a details tree.



emxUIWizard.css

To change how the wizard pages appear, edit the style sheet file emxUIWizard.css. The style sheet contains the following style definitions:

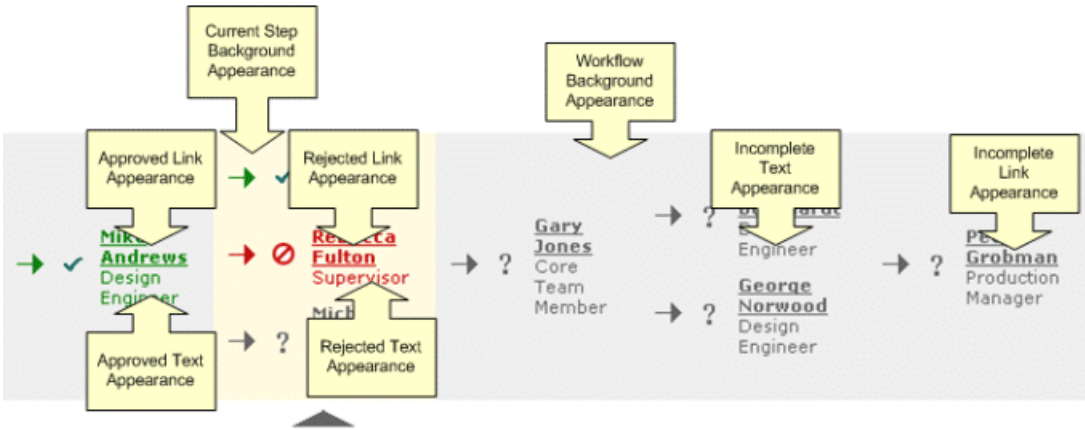
Definition	Name	Categories
body { }	Default Background Appearance	Background



emxUIWorkflow.css

To change how the list appears, edit the style sheet file emxUIWorkflow.css. The style sheet contains the following style definitions:

Definition	Name	Categories
tr.workflow { }	Workflow Background Appearance	Background
td.current { }	Current Step Background Appearance	Background
td.incomplete { }	Incomplete Text Appearance	Background, Font, Position
td.approved { }	Approved Text Appearance	Background, Font, Position
td.rejected { }	Rejected Text Appearance	Background
td.incomplete a { }	Incomplete Link Appearance	Background, Font
td.approved a { }	Approved Link Appearance	Background, Font
td.rejected a { }	Rejected Link Appearance	Background, Font



Configuring the Main Navigator Page

The main Navigator page contains the dynamic user interface components that are common to all applications and all commands: the banner, global toolbar, My Desk and Actions tabs, and content frame (where the table and form pages are displayed). This page is called for all dynamic UI applications and is constructed by `emxNavigator.jsp`. This section describes how to configure the Navigator page components and define how it should appear when a user first logs in.

You can configure how the Navigator page looks when a user first logs in or when the page is called externally from an email message or other Web page:

- the default page to display in the content frame
The content frame is the main work area of the Navigator page, where the table and form pages display, and is called the Home page within the user interface. You can control which page is displayed when users first log in. Users can use the Preferences tool to choose their own Home page preference, which overrides any page that you specify.

You implement this configuration by passing URL parameters to `emxNavigator.jsp`. These parameters are described in the sections that follow. To implement a configuration as a default for when users log in, use the `eServiceSuiteFramework.URL` property in `emxSystem.properties` and include the parameters in the `emxNavigator.jsp` URL.

To implement a configuration when calling `emxNavigator.jsp` from a link in a Web page or email message, add the appropriate parameter to the `emxNavigator.jsp` URL for the link. For information on accessing `emxNavigator.jsp` through an external link, see [Accessing the Applications Externally](#).

In addition to adding parameters to the URL for `emxNavigator.jsp`, specifying the default page for the content frame can also be implemented using the `emxNavigator.Home.ContentPage` property in `emxSystem.properties`. This property and the URL parameters you can use to specify a default home page are described below in [Configuring the Default Home Page for the Content Frame](#).

Configuring the Default Home Page for the Content Frame

There are a number of ways to specify the page that should display in the content frame of the Navigator window when users log in. The content frame is referred to as the Home page in the user interface.

- The `ContentPage` parameter passed to `emxNavigator.jsp`.
The value for the `ContentPage` can be the URL for any JSP plus parameters. For example:

```
emxNavigator.jsp?ContentPage=../engineeringcentral/  
emxMyTasksSummaryFS.jsp
```

Where `../engineeringcentral/emxMyTasksSummaryFS.jsp` is the URL for the Tasks page within Engineering Central, as shown below.

Tasks							
Active							
Actions							
☐	Name ▲	Action	Task Instruction	Due Date	Due Time	Route	WorkSpace
☐	Check figures	Investigate	Please check the figures on page 34.	May 31, 2002	10:00:00 AM	Spec Review	Workspace 200
☐	Comment	Comment	Please add your changes.	May 14, 2002	1:00:00 PM	Route 100	Workspace 100

Currently, the URL assigned to the *ContentPage* parameter can contain only one parameter/value pair as part of the URL. Any additional parameters are ignored.

- The *MenuName/CommandName* parameters passed to *emxNavigator.jsp*. These parameters let you specify the name of a menu and command object pair that should be displayed in the content frame. The system displays the URL associated with the command object attached to the menu. The parameters are:

MenuName—Parameter name to get the name of the menu object for the submenu that has the command object attached. Note that the links in the Actions submenus cannot be used for display in the content frame.

CommandName—Parameter name to get the name of the admin command object from which the href URL is to be obtained.

For example:

```
emxNavigator.jsp?MenuName=TMCMYDesk&CommandName=TMCRoutesMyDesk
```

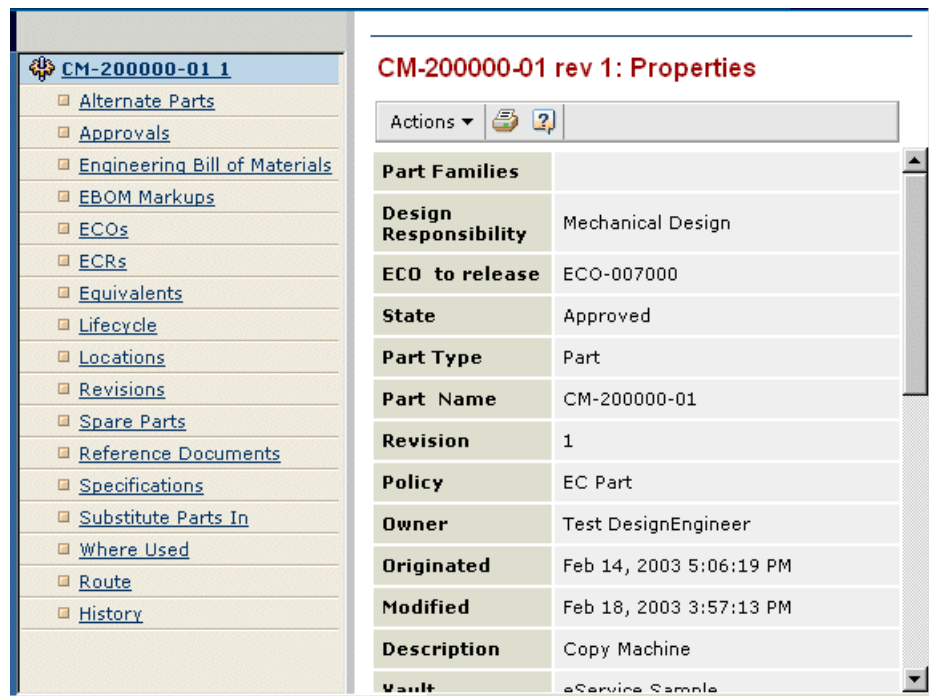
Where *TMCMYDesk* is the name of the menu object for the Team Central My Desk submenu and *TMCRoutesMyDesk* is the name of the command object for the Routes link. This command object contains an href parameter that points to the JSP for the Routes page.
- The *objectId* parameter passed to *emxNavigator.jsp* to display the tree for the business object.

For example:

```
emxNavigator.jsp?objectId= 29547.17050.1266.35700
```

Where 29547.17050.1266.35700 is the ID for the business object for which the tree is to be displayed.

This example will launch the Navigator page with the tree page specific to this business object as shown below.



- The `emxNavigator.Home.ContentPage` property specified in `emxSystem.properties`. Another way to specify the content frame for `emxNavigator.jsp` is to specify a URL in the following property in `emxSystem.properties`:
`emxNavigator.Home.ContentPage = emxDefaultNavigatorContentPage.jsp`
 By default this property is set to the default home page, which is the welcome page or `emxDefaultNavigatorContentPage.jsp`.

If more than one parameter or properties file setting is called, the system uses this precedence order:

1. Any parameter included with `emxNavigator.jsp` that is specified in a URL link.
 Any parameter specified in an external URL link—for example, in an email or Web site—overrides all other settings.
2. User's Home page preference setting.
 The user's Home page preference overrides the content page settings in `emxSystem.properties`. So you can use the properties to configure the content page for when users first log in but users can override this by choosing a preferred Home page.
3. Any parameter included with `emxNavigator.jsp` that is specified in the `eServiceSuiteFramework.URL` property in `emxSystem.properties`. If more than one parameter for the content page is specified in the property, the system uses this precedence order:
 - a) `ContentPage` parameter
 - b) `MenuName/CommandName` parameters that call the URL associated with the command object
 - c) `objectId` parameter for tree for object
4. `ContentPage` property specified in `emxSystem.properties`.

If none of the above parameters are passed or any error conditions occur due to an incorrect configuration, the default home page (`emxDefaultNavigatorContentPage.jsp`) is displayed in the content frame.

Adding a Custom Application

This procedure describes the steps needed to integrate a custom application into the dynamic user interface. The custom application will appear in the My Desk menu along with MatrixOne applications.

To add your own application

1. Create a directory for your application's JSPs within the ematrix directory. This directory is under the staging directory within the EJB/RMI installation directory. This directory is equivalent to the subdirectories for the MatrixOne applications, such as ematrix/teamcentral. Place all your JSPs in this directory.
2. Create an images directory under the subdirectory you created in Step 1. Place all image files for your application in this directory.

You must run the warutil and deploy the archive file when you're done adding files for the application.

3. List your custom application in the DisplayedSuites property in emxSystem.properties. The name you give it here is the name you should use when adding the properties for the suite in Step 4.

Custom applications cannot have the same name as an existing application and they cannot have the same name as an existing application without the "eServiceSuite" prefix. For example, the name for Engineering Central is eServiceSuiteEngineeringCentral so you cannot have a custom application with the name eServiceSuiteEngineeringCentral or EngineeringCentral.

```
eServiceSuites.DisplayedSuites =
eServiceSuiteEngineeringCentral, \
    eServiceSuiteTeamCentral,      \
    eServiceSuiteSupplierCentral,  \
    CustomCentral \
```

4. Add a set of properties to emxSystem.properties to tell the system the name of your application directory, properties files, and other important information. Each MatrixOne application has a set of properties and the set for a custom application called Custom Central is shown below. Create an equivalent set for your application and change the property names and the values to reflect your application. The property names should match the name used in the DisplayedSuites property. For a description of the properties, see [Configuring System Properties](#) in Chapter 7.

```
# Custom Central
```

```
CustomCentral.URL = common/emxNavigator.jsp
CustomCentral.Directory = customcentral
CustomCentral.ApplicationPropertyFile =
emxCustomCentral.properties
# ApplicationStartPage is within 'Directory' specified
CustomCentral.ApplicationStartPage = emxCustomFrames.jsp
CustomCentral.StringResourceId =
emxCustomCentralStringResource
```

```
CustomCentral.PropertyFileAlias=emxCustomCentralProperties
```

5. Define a My Desk submenu for the application. Creating a submenu for the application effectively registers it. The submenu will contain all top-level commands for your application. For instructions on creating menus and commands, see [Building Menus and the Global Toolbar](#).
6. Register new schema and configure existing schema as needed using the instructions in [Configuring the Schema](#) in Chapter 5.

If you add a custom type, you can specify the icon to use for the type's root tree node using the `emxFramework.smallIcon.TYPE_SYMBOLIC_NAME` property in `emxSystem.properties`. For more information, see [Configuring the Icon for a Type's Tree](#).

7. Define additional and configure existing trees, tables, forms, and menus as needed. See [Building Navigation Trees](#), [Building Table Pages](#), and [Building Form Pages](#).

Building Menus and the Global Toolbar

This section describes how to add and configure menus, which includes the My Desk menu, the Actions menu, and the global toolbar. Menus present a list of commands that users can choose.

Building menus for an application replaces the registration process available in previous versions of the framework that used administration wizards.

- [Types of Menus](#)
- [Building the Menus](#)

Types of Menus

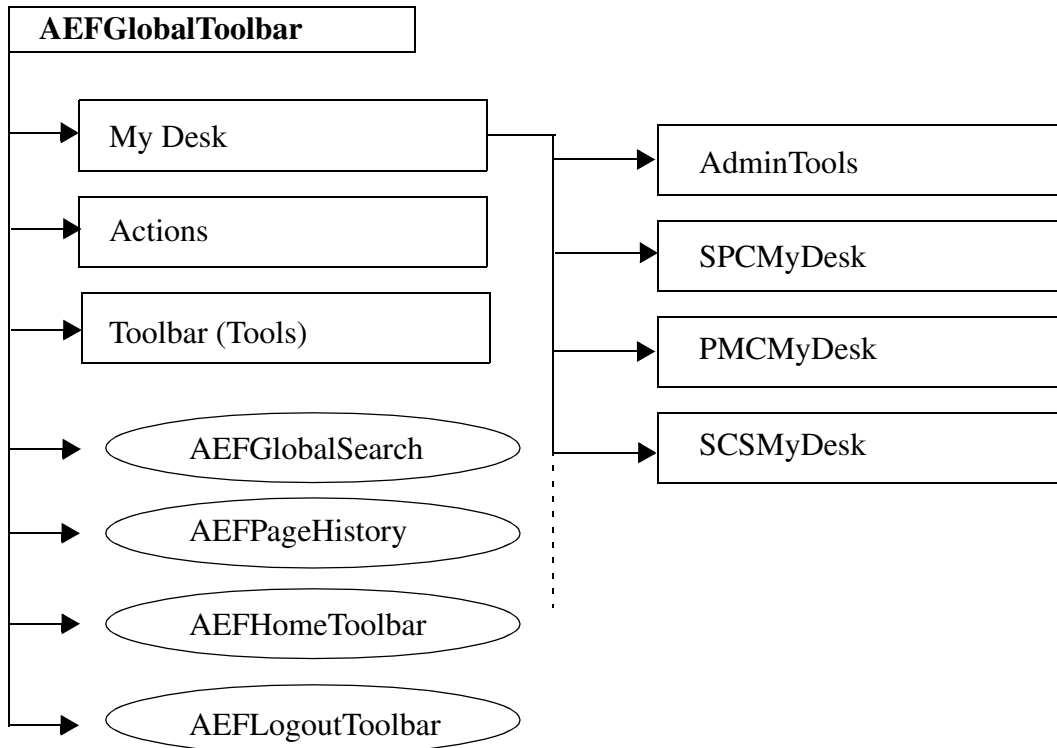
Three types of menus appear within the dynamic user interface. This section describes each of the menu types:

- [Global Toolbar Menu](#)
- [My Desk Menu](#)
- [Actions Menu](#)
- [Tools Menu](#)

Navigation trees can be considered another type of menu and are also represented by menu objects. They are described in a separate section, [Building Navigation Trees](#).

Global Toolbar Menu

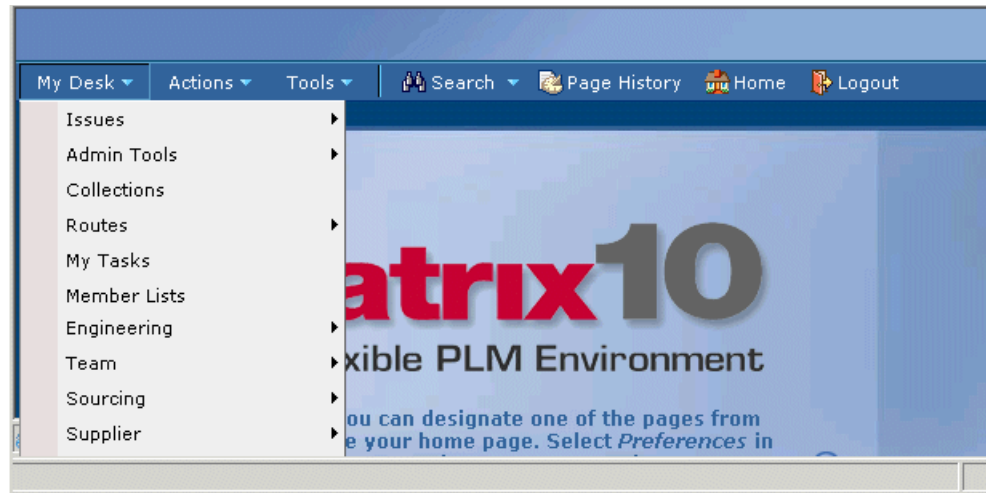
The global toolbar is a top-level toolbar which is connected to the My Desk, Actions, and Tools menus, and other commands.



AEFGlobalSearch, AEFPageHistory, AEFHomeToolbar, and AEFLogoutToolbar are also available under the Tools menu.

My Desk Menu

The My Desk menu contains commands for features that are shared by all applications, such as issues, collections and routes. Also, each application has its own submenu within the My Desk menu. For example, in the graphic below, the My Desk menu has submenus for Engineering Central, Team Central, Sourcing Central, and Supplier Central. Commands on the My Desk menu present top-level lists of items important to the current user.



The My Desk metaphor indicates that the commands included in the My Desk menu should display existing information, as if users are looking through information on their desk. My Desk menu items typically represent commands that present table pages. These table pages list objects that the user owns and/or needs to work on. When selected, My Desk menu items display in the main frame and not in a new popup window.

Actions Menu

Commands in the Actions menu perform an action, such as creating a part, ECR, or product. Like the My Desk menu, each application has a submenu with commands relevant to that application. When selected, Actions menu commands display in a popup window.

The commands on the Actions menu perform actions that are not dependent on a selected item or the current context. Because users usually work within the context of a particular business item (a task, meeting, document, part, ECR, etc.), the commands for most actions appear on pages that pertain to a particular item or set of items and not on the Actions menu. For example, one action a part owner can perform is to delete a part but the owner must first select the part to delete. Therefore, the command for deleting a part is on the Parts page.

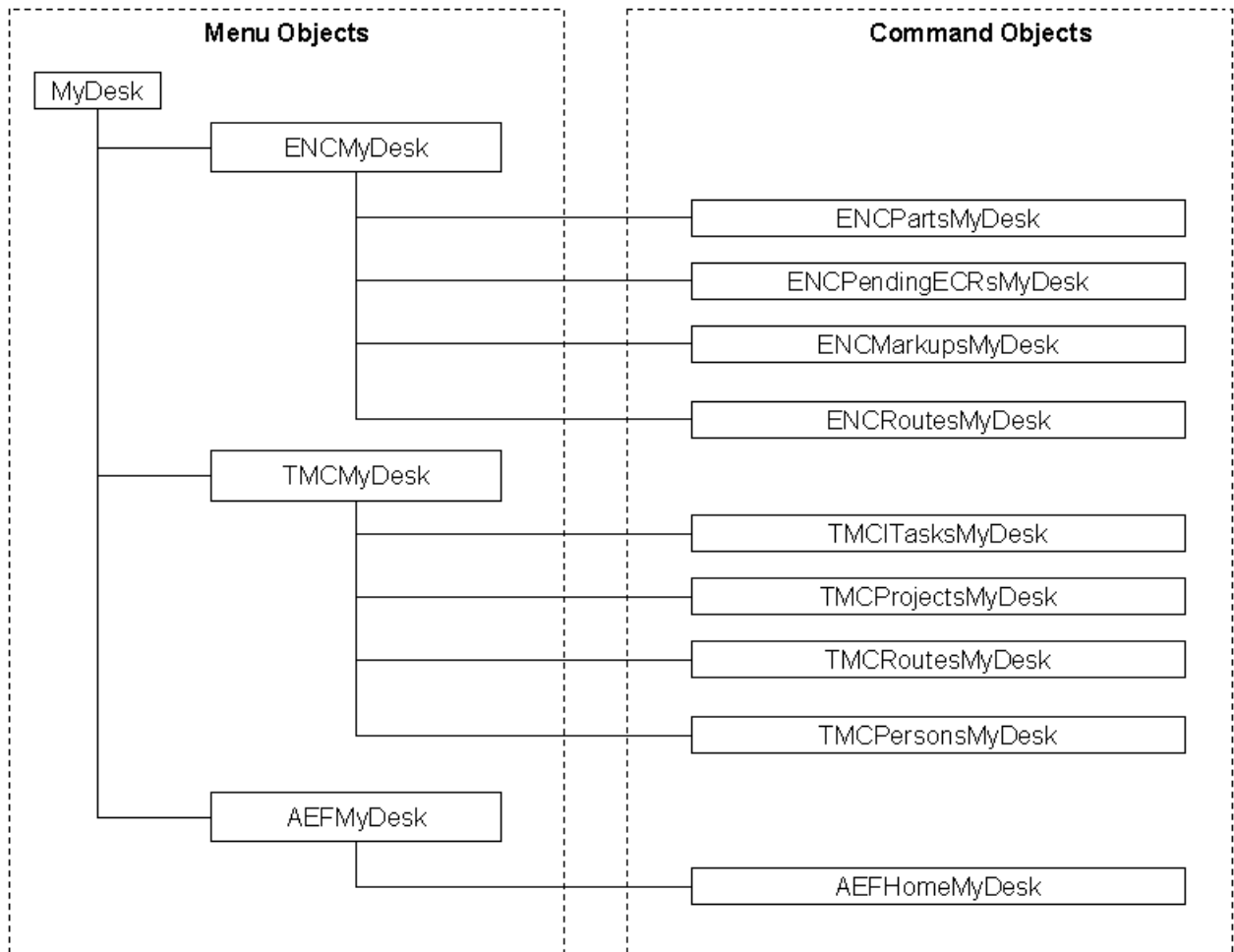
Tools Menu

The Tools menu contains commands that are typically required by all applications and are therefore installed with the framework. For example, the commands for logging out, changing the current user's password or preferences, and returning to the Home page are in the Tools menu. Some of the Tools menu commands also appear in the global toolbar for easy access. These include the Search submenu, and the Page History, Home and Logout commands.

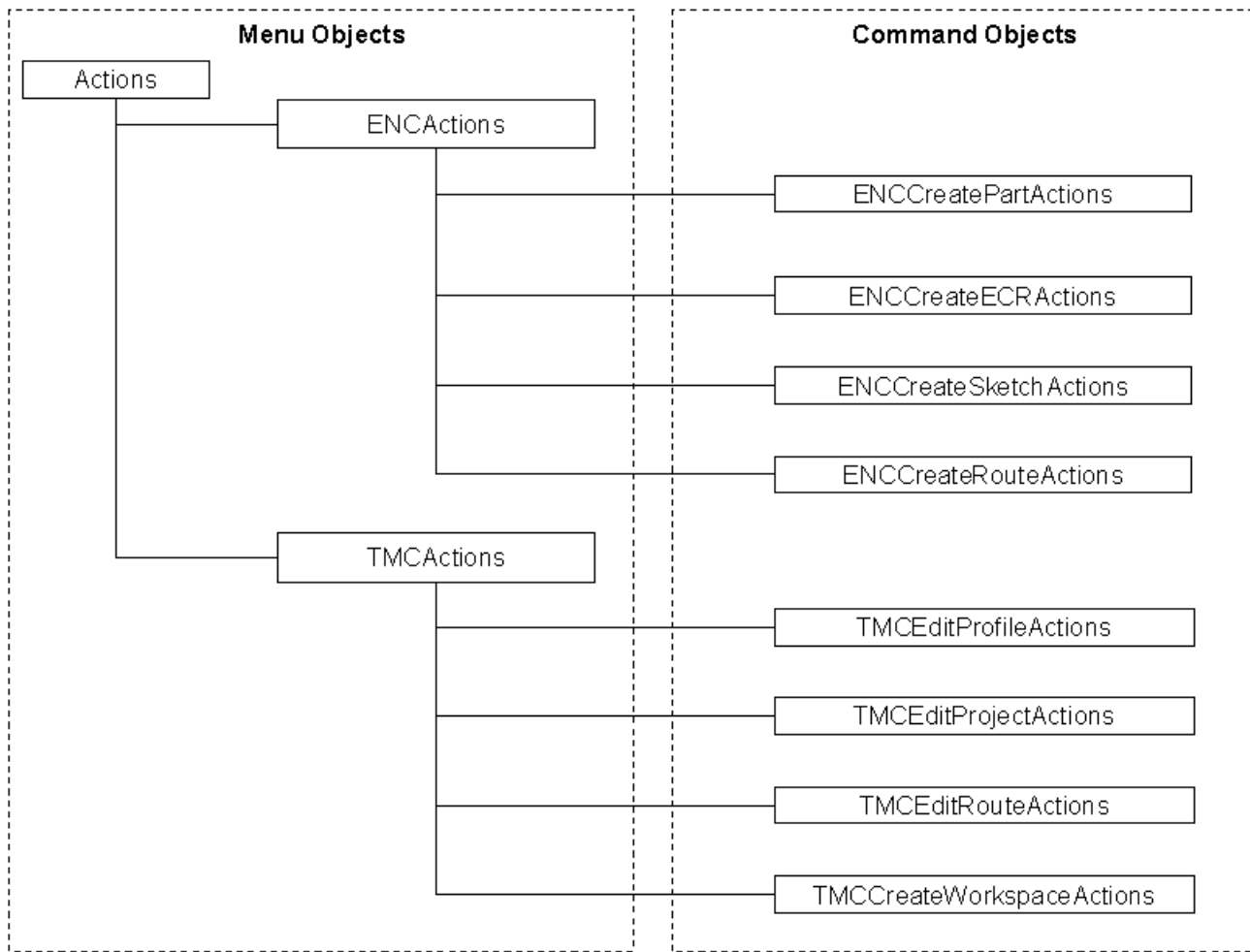
Building the Menus

The framework installs three menu administrative objects, which represent the three types of menus in the dynamic UI: the My Desk menu, the Actions menu, and the Toolbar menu.

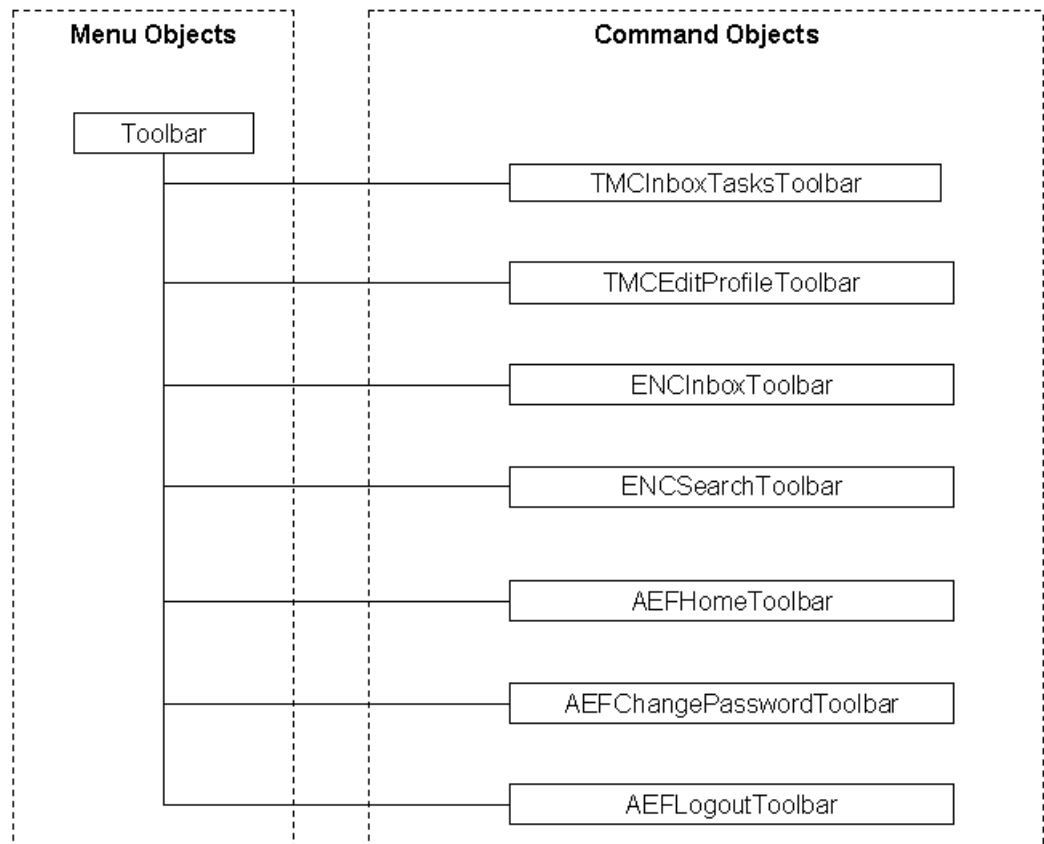
The definition for the My Desk menu object contains a menu object that represents the submenu for each installed application. Each of these submenu menu objects is in turn assigned a command object for each link in the submenu. This graphic shows a sample My Desk admin object structure needed for an installation that has Engineering Central and Team Central.



Similarly, the definition for the Actions menu object should include a menu object that represents the submenu for each application. Each submenu should be assigned a command object for each link in the menu.



Because the global toolbar contains no submenus for applications, the command objects are connected directly to the Toolbar menu object. The names of global toolbar commands installed with the framework are prefixed with “AEF”.



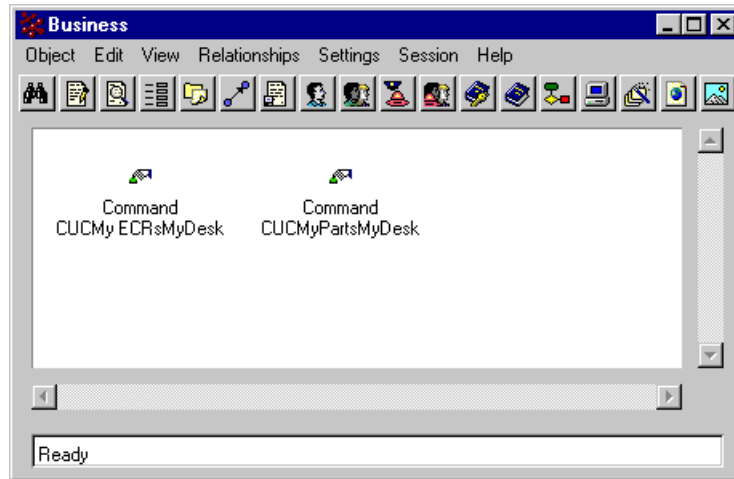
This section lists the main steps for building the menus for an application. The graphics included in the procedure represent the administrative objects needed to add submenus for a sample custom application called Custom Central and add several commands to the application's My Desk submenu, including My ECRs and My Parts commands.

For details on parameters and settings available for menu and submenus, see [Parameters for Toolbar Menu Objects](#) and [Settings for Toolbar Menu Objects](#).

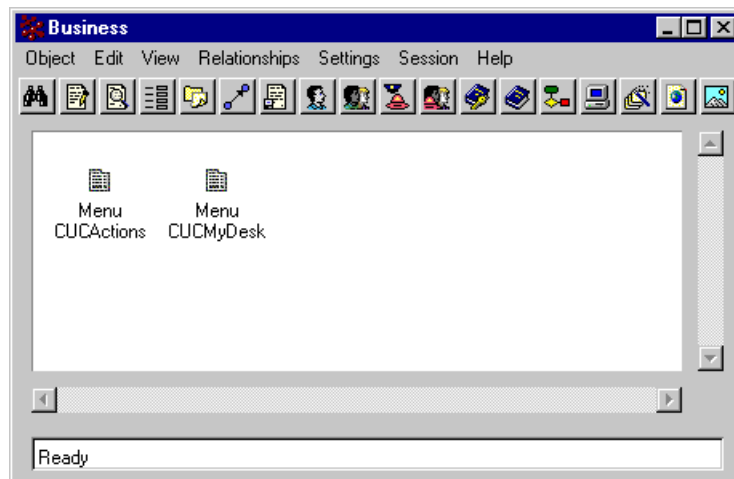
To build menus for an application

1. Create a command object for every link that should be added to the My Desk and Actions submenus for the application. If the application has additional global toolbar tools, create command objects for those also (the global toolbar commands are typically reserved for commands that apply to all applications). A command object can be used in multiple menus so if another application uses the command you want for a menu, you don't have to create another command object.

When you create a command, you define the JSP the command should call when a user clicks it, how the command should appear (for example, its label), and who can access it. For a description of how to fill in the parameters and settings for commands, see [Parameters for Toolbar Menu Objects](#) and [Settings for Toolbar Menu Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).



2. Create a menu object for the application's My Desk submenu and Actions submenu.
 When you create a submenu, you define how it should appear, the commands that should be included in it, and any submenus within it. For a description of how to fill in the parameters and settings for submenus, see [Parameters for Toolbar Menu Objects](#) and [Settings for Toolbar Menu Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

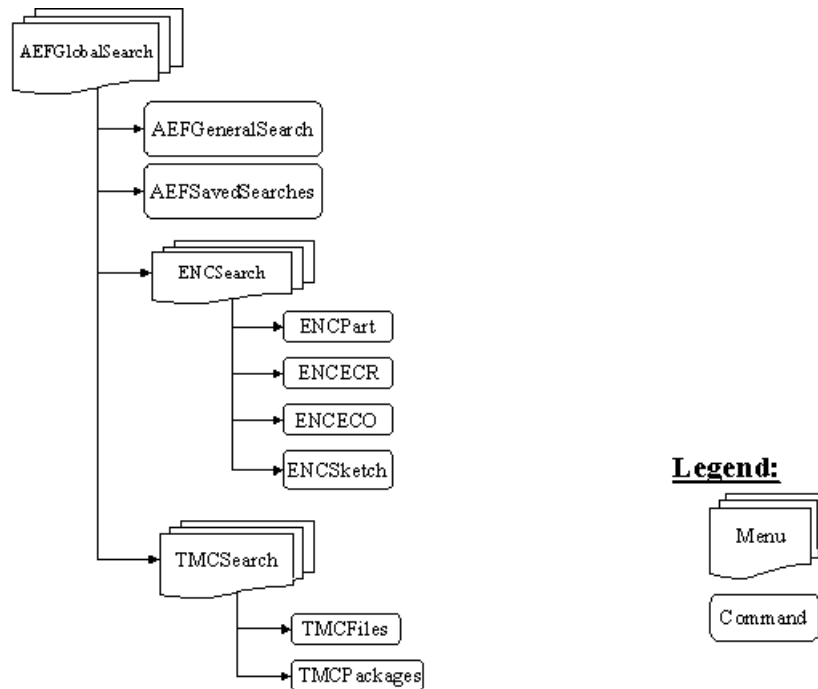


3. Connect the submenus you created to the appropriate top-level menu object. For example, connect the menu object for the application's My Desk submenu to the My Desk menu object and connect the menu object for the application's Actions submenu to the Actions menu object. In Business Modeler, the order the commands are listed in the Items tab of the Edit Menu dialog box are the order they appear on the menu. You can drag and drop the commands as needed to change their sequence.
4. If you are working with the Web-based user interface as you are making changes and want to see your changes in the user interface, click the **Reload Cache** tool in the global toolbar and click the browser **Refresh** button.
 The cache is refreshed automatically when the component age expires. This setting is in `emxSystem.properties`.

Only persons assigned to the Administration Manager role have access to the Reload Cache tool.

Configuring the Global Search Menu

The top level Global Search menu, AEFGlobalSearch, holds all the Generic and Specific search menus/commands as shown in the following menu structure.



Upon clicking on this Global Search button, the search window opens with the JSP page configured in the first command connected to the AEF search menu AEFGlobalSearch. By default, the first command will be the AEF Generic TNRV Search defined using the

command “AEFGeneralSearch.” The following table describes the list of settings for AEFGlobalSearch menu.

Parameter and Settings	Description	Possible Values
Registered Suite	The required suite that the menu belongs to. In this case the menu belongs to the Framework.	Framework
href	emxSearch.jsp	—
defaultSearch	The URL parameter can be passed with the name of the Search command. The JSP page configured for this command is used for displaying the search page when opening the Search window. - If the logged in user does not have access to the designated “defaultSearch” command, the first available and accessible search command will be displayed. - If no “defaultSearch” is passed in, then by default the General TNRV search page will be displayed.	- SPCSCOSearch - TMCSearch - Default is the first available command in search toolbar AEFSearchToolbar. The first available command in the AEF install will be the general search command AEFGeneralSearch.
defaultSearch	The URL parameter can be passed with the name of the Search command. The JSP page configured for this command is used for displaying the search page when opening the Search window. - If the logged in user does not have access to the designated “defaultSearch” command, the first available and accessible search command will be displayed. - If no “defaultSearch” is passed in, then by default the General TNRV search page will be displayed.	- SPCSCOSearch - TMCSearch - Default is the first available command in search toolbar AEFSearchToolbar. The first available command in the AEF install will be the general search command AEFGeneralSearch.
helpMarker	Default help marker for the search page.	emxhelpsearch
toolbar	Name of menu to be used as the toolbar in the search page	AEFGeneralSearchToolbar

Configuring Search Commands and Menus For the Global Search

The Search menus and commands are configured as a toolbar. For details of how to configure and use toolbars, see [Building Toolbars](#).

With only AEF installed, the Global Search menu AEFGlobalSearch will contain two commands, AEFGeneralSearch and AEFSavedSearches. The AEFSavedSearches command is used for displaying the list of saved searches available to the current user. The AEFGeneralSearch is used for displaying the General search page.

Applications create and connect the specific commands and menus to the appropriate location in the Global Search menu object as shown in the menu structure above.

For example, Team Central related searches are connected to one menu called TMCSearch and this menu is connected to the AEF Global Search menu AEFGlobalSearch. Similarly, all other applications group the searches and connect them as required. The commands and menus must conform to the requirements of the toolbar component.

The command configured for any application specific / custom searches must conform to the following settings and rules.

Parameter and Settings	Description	Possible Values
Registered Suite	Suite Name for the search command.	- EngineeringCentral - TeamCentral
href	- To be assigned to the search JSP page to be used for displaying the search content. This specific JSP file must contain a JavaScript method called “doSearch()”, to initiate the search process by calling a processing page. - All original URL parameters passed-in when opening the search window are preserved and passed down to the specific search pages. - The processing page must refresh the whole search window with the results page. Results page may use the AEFSearchResultsToolbar as the toolbar for this page to leverage “Revise Search” and “Add to Collection” support.	\${SUITE_DIR}/emxTeamFilesSearch.jsp?....
Target Location	This must be assigned to “searchContent.”	searchContent

URL Parameters for Generic Search

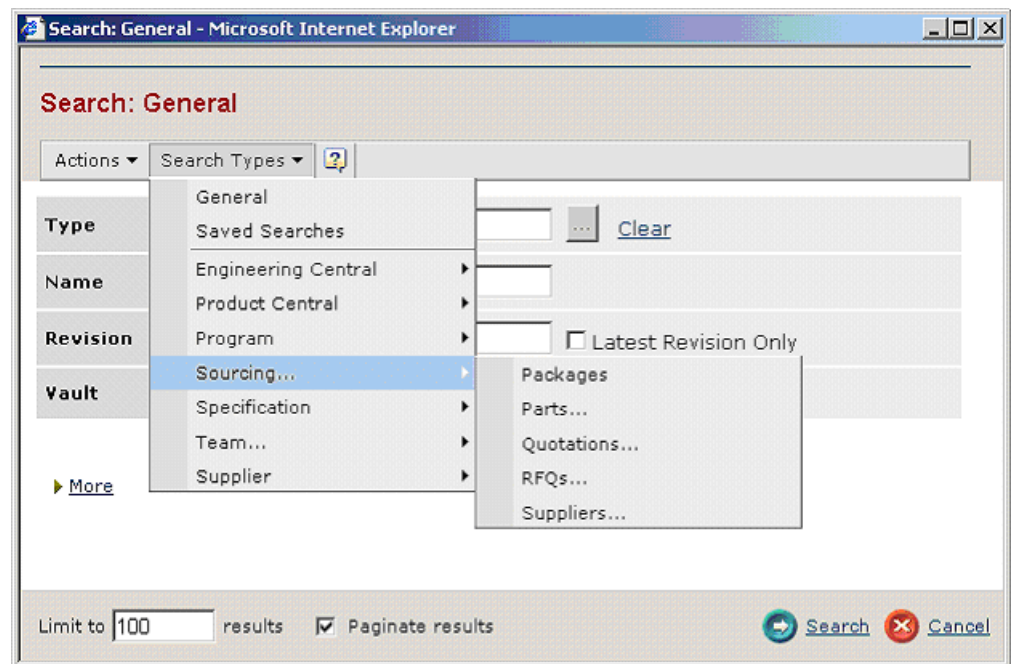
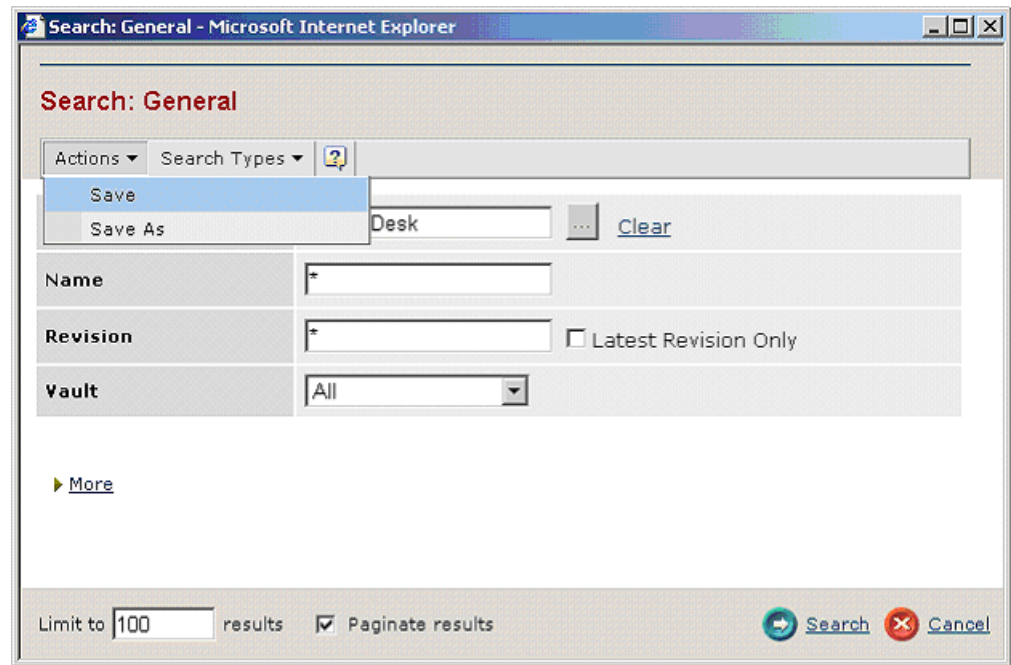
The following table describes the settings and possible values that can be passed in as URL parameters for generic search.

Settings	Description	Possible Values
typeSelection	Determines whether to display the results using radio button (single) or checkbox (multiple)	single multi [default]
useTypeChooser	setting that tells the search component whether to display a type chooser or not for the type field. If this parameter is not passed into the search component, then the default will be true.	true false
editable	setting that tells the search component whether to make the type field editable or not. A value of true will make the text box editable. A value of false will make it read-only. If this parameter is not passed into the search component, then the default will be false. Note: The type field will only be set to true if the browser language setting is set to "English". If this value is set to true and the browser language is NOT set to English, then this setting is ignored and reverted to false.	true false

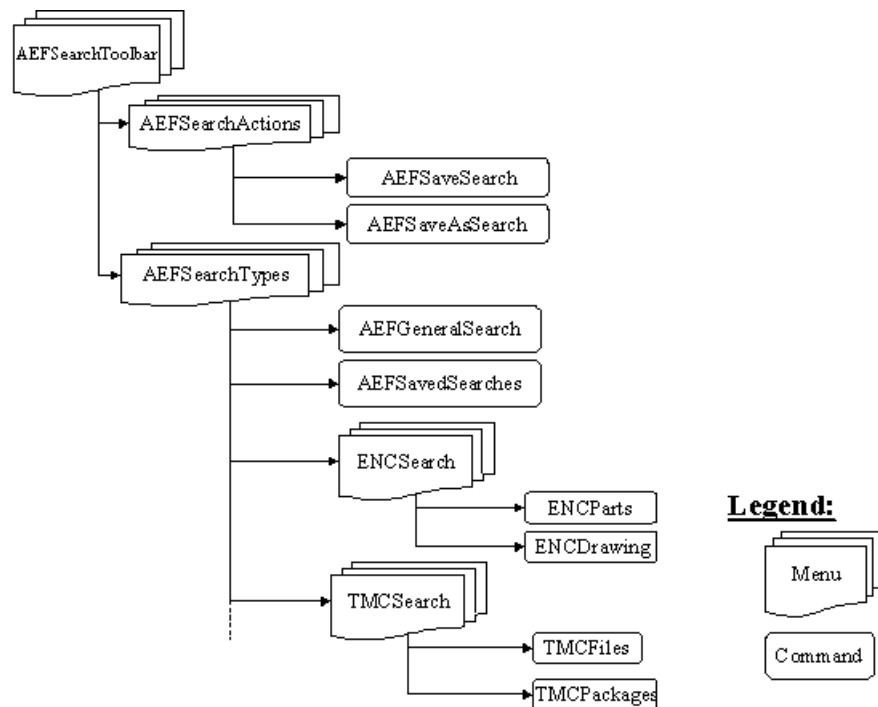
Settings	Description	Possible Values
InclusionList	Means that the type(s) that is passed in will override the list of types defined in the properties file. The type or list of types will be displayed in the type chooser and users can select the types they wish to search for. The Inclusionlist can either be a property value key or a list of types.	InclusionList =type_part, type_ECO InclusionList= emxFramework.GenericSearch.types
ExclusionList	Means that the type(s) that is passed in will not be displayed in the type chooser. All other types will be displayed and users can select the types they wish to search for from the type chooser. The Exclusionlist can either be a property value key or a list of types.	ExclusionList =type_part, type_ECO ExclusionList= emxFramework. GenericSearch.ExclusionList
submitURL	Holds the URL that the result page should call when the submit button is pressed	AnyPage.jsp
defaultSearch	URL parameter can be passed with the name of the Search Command. The JSP page configured for this command will be to be used for displaying the search page when opening the Search window. • If the logged in user does not have access to the designated "defaultSearch" Command, the first available and accessible search command will be displayed. • If no "defaultSearch" is passed in, then by default the General TNRV search page will be displayed.	SPCSCOSearch TMCSearch Default is the first available command in search toolbar "AEFSearchToolbar". The first available command in the AEF install will be the "General Search" command "AEFGeneralSearch."

Search Window with General and Specific Search Toolbar

The Search window contains a Toolbar AEFSearchToolbar with Search Types menu and Saved Searches menu as shown below.



The menu structure used in this page is slightly different than the one at the top level Global Search. All app specific / custom sub menus connected here will be same as in the Global Search menu. The following figure shows the menu structure for the Search toolbar.



Settings for Commands in My Desk

This table shows settings for command objects used for the the My Desk menu. For a list of other parameters and settings that can be used with the My Desk and other menus, see [Parameters for Toolbar Menu Objects](#) and [Settings for Toolbar Menu Objects](#).

Setting	Description	Accepted Values/Examples
Selectable in Preferences	Only for commands in My Desk submenus. Controls whether the link is available as a preferred Home page. If not set, the system assumes true and the command is listed as a Home page preference. Commands that call pages to display in a popup window, such as an object search command, should not be available as a Home page preference.	True False

Building Navigation Trees

This section describes how to add and configure navigation trees. Navigation trees present information related to a particular object.

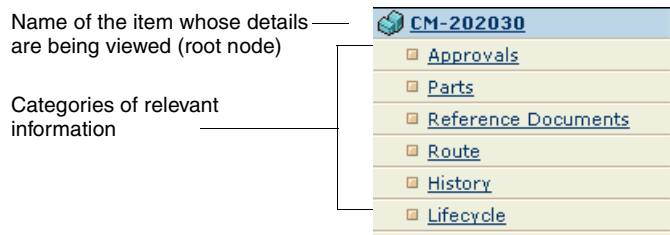
- [Overview of Navigation Trees](#)
- [Two Modes for Tree Display](#)
- [Building a Navigation Tree](#)
- [Tree JavaScript API for Custom JSPs](#)
- [Deprecated Methods](#)
- [Parameters for Tree Menu Objects](#)
- [Parameters for Tree Category Command Objects](#)
- [URL Parameters Accepted by emxTree.jsp](#)
- [Using and Configuring the Default Tree](#)
- [Calling Specific Trees for an Object Type](#)
- [Configuring Dynamic Expand for Tree Categories](#)
- [Guidelines for Writing Structure Tree JPO](#)

Overview of Navigation Trees

Navigation trees let users access information related to a particular business object. Navigation trees consist of a:

- root node, which represents the business object, and
- categories under the root node, which represent information related to the business object, such as history, attached files, and approvals

For example, this graphic shows the navigation tree displayed when a person views detailed information for a part. The name of the object whose details are being viewed, the root node, is always displayed at the top of the tree. The categories of relevant information are displayed in an indented list under the object name.



Users view the information in a tree by clicking a particular category. Most categories display table pages with lists of objects in the category for the selected object. For example, to view the Properties page for the object, which are the main attributes for the item, click the object's name at the top of the tree. By default, the object's name is automatically selected when the tree first opens, which is why the Properties page always shows first when a tree appears for an object. Trees can be configured so a category is selected when the tree first opens (see the `DefaultCategory` parameter for `emxTree.jsp` and the `Default Category` setting for tree menu objects).

Different object types typically require different categories in their tree so you can configure different trees for each type of object. For example, the tree for a part may need a category for related ECRs, while the tree for a supplier may need a category for locations. On the other hand, trees for different object types may have some categories in common, such as history and attached files. The framework comes with a default tree that contains categories that many types use. The system uses this default tree if no tree is defined for a type. For more information, see [Using and Configuring the Default Tree](#).

You can control access to every category in a tree. So some users may see only three categories for the Part tree while others see six categories.

Users access trees by choosing a business object from a table page. (Within the context of the current user interface design, this is the only way to select a tree but a different design could have trees called from other components.) For example, in the graphic below, suppose the user chooses the package in the table.

My Packages

All

Actions

	Name	State	Owner	Response Status	RFQ	Workspace Folder(s)
☐	 Pkq100	Started		0/0	0	Folder1 Subfolder1A Subfolder1B

Page 1 of 1

Pkq100

[Attachment](#)

[PackageRFQ](#)

[History](#)

Pkg100 - : Properties

Actions

Name	Pkg100
Workspace Folder	 Folder1  Subfolder1A  Subfolder1B
Comments	Please see attached disclaimer.
Description	Package for assembly 2922822
Response Status	0/0

The system shows the navigation tree for the object on the left. By default, the root node for the object is selected, so the Properties page for the object is shown on the right. The Properties page is an example of a form page.

Selecting a category typically displays a table page, as shown below for this RFQ tree and ECR category.

The selected category is highlighted in blue

The screenshot shows a web application interface. On the left is a navigation tree with a search icon and a list of categories. The 'ECRs' category is highlighted in blue. On the right, the main content area is titled 'RFQ-5000: ECRs'. It features an 'Actions' menu with a dropdown arrow, a print icon, and a help icon. Below this is a table with columns: Name, RFQ Line Item, Supplier Organization, Originator, and State. The table contains six rows of data, each with a checkbox in the first column. At the bottom of the table is a 'Submit Selected' button. Below the table is a pagination control showing 'Page 1 of 10' with navigation arrows and a refresh icon.

	Name	RFQ Line Item	Supplier Organization	Originator	State
☐	ECR-2123	RLI-5000	Ford	Clark, Rex	Buyer Evaluation
☐	ECR-2323	RLI-5011	Honda	Wedgewood, Peter	Buyer Evaluation
☐	ECR-4567	RLI-5034	Honda	Maguire, Padraig	Buyer Evaluation
☐	ECR-5000	RLI-5056	Subaru	Dorval, Vera	Buyer Evaluation
☐	ECR-6000	RLI-5023	Acme Supply	Clark, Rex	Buyer Evaluation
☐	ECR-5000	RLI-5010	Ford	Clark, Rex	Buyer Evaluation

Within the context of the dynamic UI design, categories in navigation trees are always nouns. They present lists of related items or attributes for the item. Action commands are on specific pages.

Two Modes for Tree Display

When the name of a business object is selected from a table page, its navigation tree and the Properties page for the object replace the table page in the right frame of the browser window. The tree for the object can either replace the existing tree, if there is one, or it can be inserted into the current tree.

This graphic shows a tree that is inserted into the existing tree in the left frame, which is the default behavior. Leaving the existing tree visible provides a frame of reference for users, showing them how they navigated to the current object and what the object is related to. For information on configuring the tree display mode, see [URL Parameters Accepted by emxTree.jsp](#).

When an object is selected in a table, the tree for the object can be inserted into the existing tree.

RFQ 300 1: Line Items

Name	Description	Required Quantity	Required U of M	Receiving Plant	Required Availability	Response Count	Splits
LI-100	Part 392228	1000	LB (pound)		Q4	0/2	
LI-100	Part 392228	300	LB (pound)	Oh		0/2	30.0%
LI-100	Part 392228	700	LB (pound)	Utah		0/2	70.0%

LI-100 : Properties

Name	LI-100
Description	Part 392228
Part	None
Required Quantity	1000
Required U of M	LB (pound)
Required Availability	Q4
Receiving Plant	

You can configure the page that calls a tree so the tree for the selected object replaces the existing tree, as shown below.

RFQ 300 1: Line Items

Name	Description	Required Quantity	Required U of M	Receiving Plant	Required Availability	Response Count	Splits
LI-100	Part 392228	1000	LB (pound)		Q4	0/2	
LI-100	Part 392228	300	LB (pound)	Oh	Q4	0/2	30.0%
LI-100	Part 392228	700	LB (pound)	Utah		0/2	70.0%

LI-100 : Properties

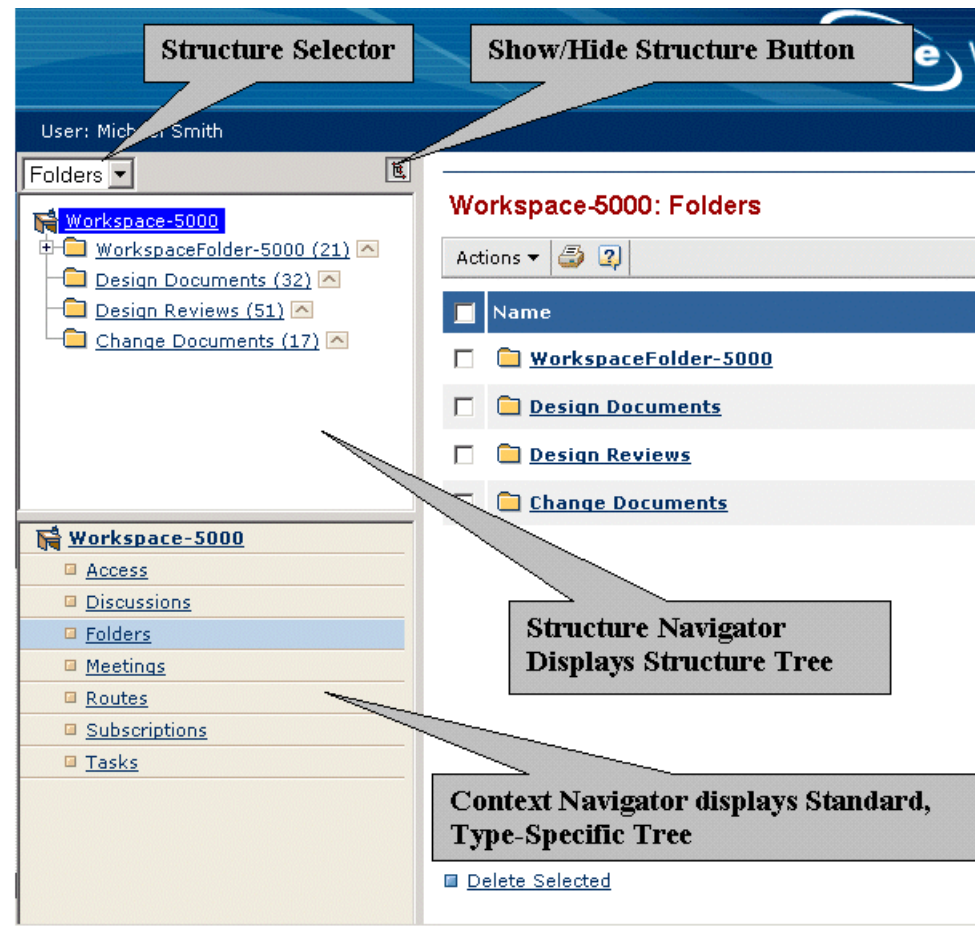
Name	LI-100
Description	Part 392228
Part	None
Required Quantity	1000
Required U of M	LB (pound)
Required Availability	Q4
Receiving Plant	

Alternatively, when an object is selected in a table, the tree for the object can replace the existing tree.

Structure Navigator

Some types of objects are related to other types in a hierarchical structure. For example, workspaces in MatrixOne Team Central and projects in MatrixOne Program Central are hierarchical: they have folders, which can each have subfolders, and each subfolder can have subfolders and so on. These types of objects can be configured so they display in a Structure Navigator, which is a frame above the Context Navigator that displays the

standard navigation trees. The Structure Navigator contains only items within the defined hierarchy and displays only their names, not their categories, allowing users to quickly navigate to the pertinent items in the hierarchy.



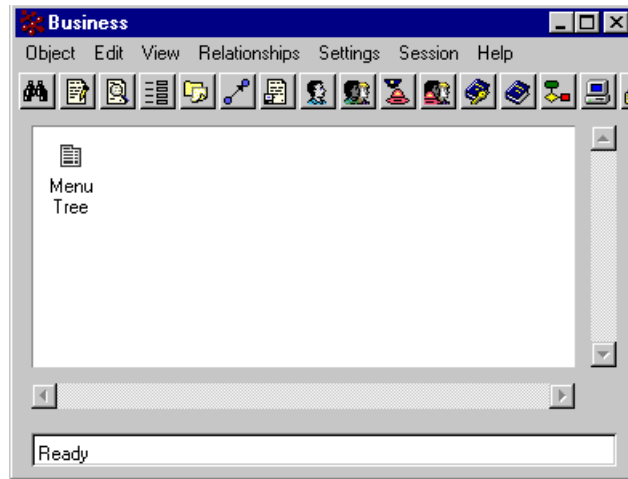
By default, the Structure Navigator opens when a user views details for an object that's been configured as a structured object. Users can hide the Structure Navigator using the Show/Hide Structure button. When a user views an object that does not have a structure defined, only the standard tree and Context Navigator displays.

For instructions on configuring an object so its structure displays in the Structure Navigator, see [Configuring a Structure Tree for an Object Type](#).

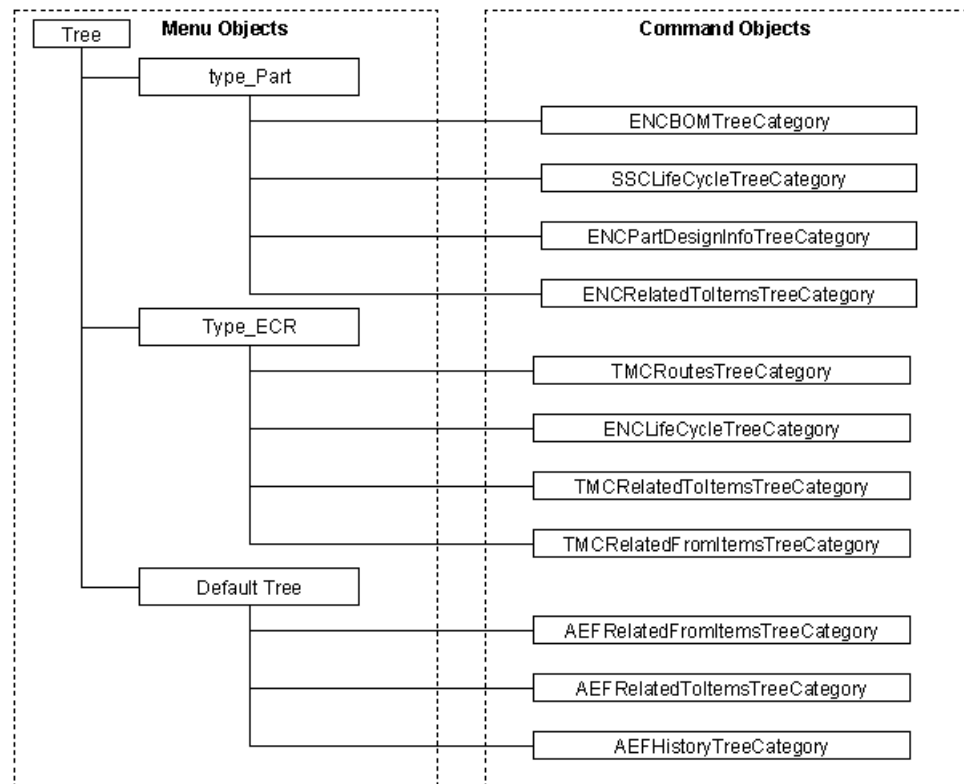
Building a Navigation Tree

Mechanisms that Define a Navigation Tree

The framework installs an administrative menu object, called Tree, which represents the navigation trees in the dynamic UI.



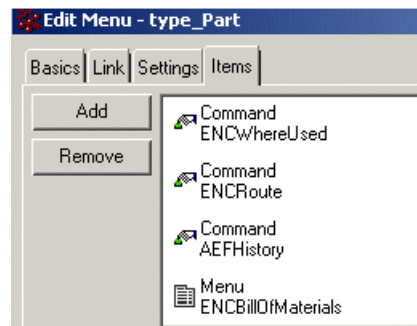
The definition for the Tree menu object includes a menu object that represents the tree for every object type that needs a navigation tree. For example, Engineering Central installs a menu object for parts and one for ECRs, which represent the trees for these types. These menu objects are assigned to the Tree menu object. The menu object for each navigation tree includes a command object for each category in the tree. This graphic shows a sample admin object structure needed for an installation that needs trees for parts, ECRs, and the default tree.



The framework installs a menu object called Default Tree, which is assigned to the Tree menu object, as shown in the above graphic. The system uses the default tree when there is no tree defined for a specific object type or any of its parent types. For information on configuring the default tree, see [Using and Configuring the Default Tree](#).

Configuring Sub-Trees as Categories

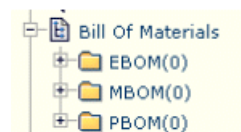
You can configure a tree's category as a sub-tree. You do this by defining the sub-tree in Business Modeler as you would any tree: create a menu object and assign commands to it. You then assign the sub-tree menu to the menu for the tree. You assign menus to menus in the Items tab, where you assign commands to a menu.



Here is an example of a Bill of Materials category that is a sub-tree.



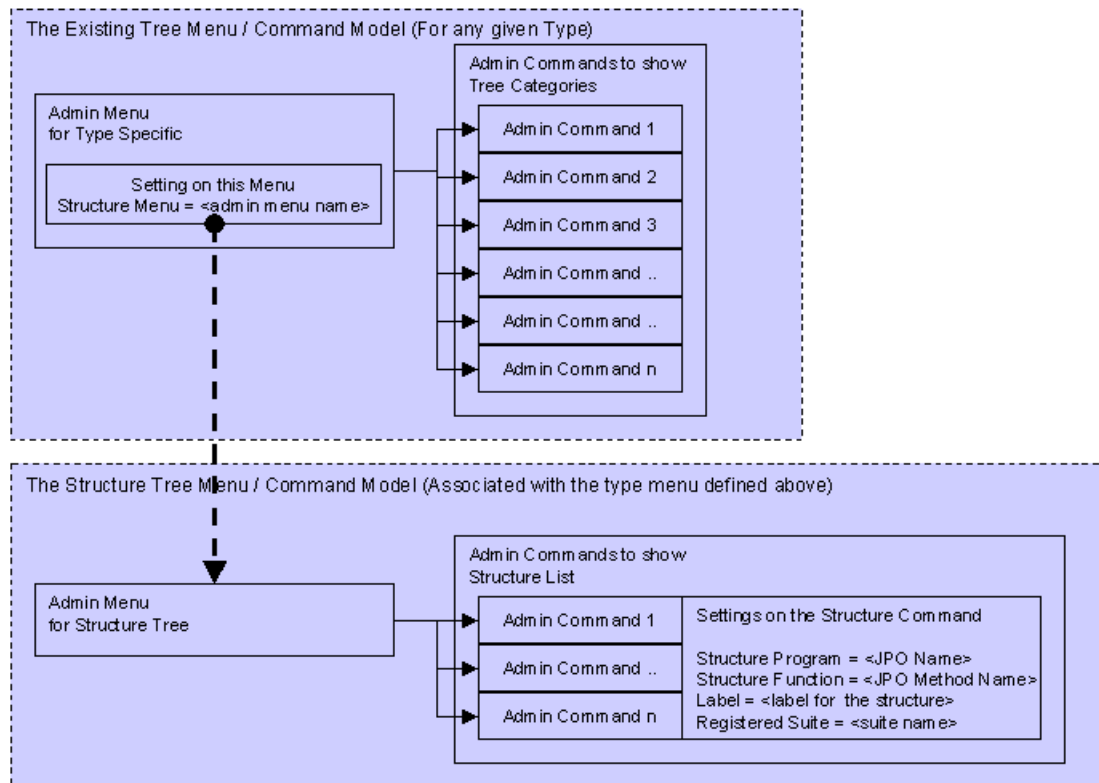
When the sub-tree is expanded, it looks like this.



When clicked, the Bill Of Materials node will not show a page in the right frame, unless one is configured. Clicking the Bill Of Materials node behaves as though you clicked on the + or - signs, expanding and collapsing the sub-tree. If a page is configured for the Bill Of Materials node, that page is displayed in the right frame.

Configuring a Structure Tree for an Object Type

This section describes how to configure an object type so the objects related to it in a hierarchical structure display in the Structure Navigator. The below model shows the two admin menus that need to be defined for implementing an object's structure tree.

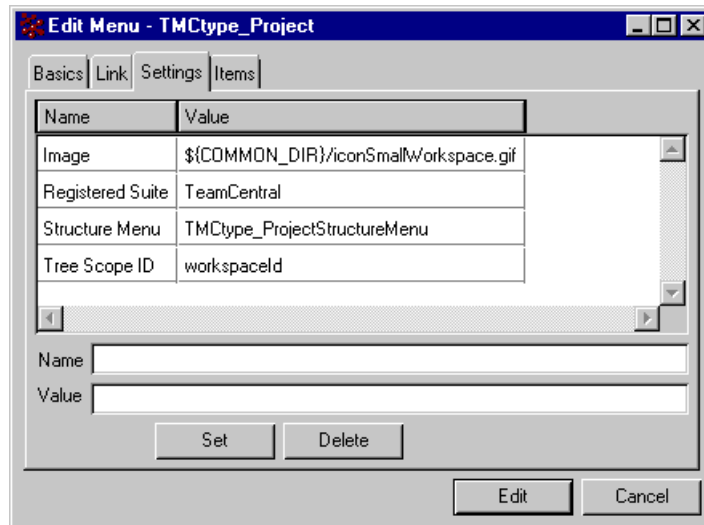


The top menu is the standard tree menu required for any object type to be displayed in a navigation tree. This is the same exact tree menu used currently in all the applications to show the tree component. This menu is defined for a specific object type to show the root node and its categories.

The second, structure menu is associated with the regular menu and is used to display a structure tree along with the regular tree. This second admin menu is associated to the main regular tree menu using a setting on that menu. The setting name is:

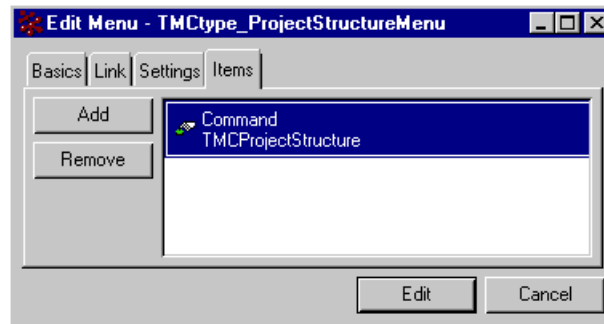
Structure Menu = <the admin menu name to be used for Structure>

For example, this is the menu object for the workspace tree in Team Central. Workspaces were originally called projects so the symbolic name is type_Project. The Structure Menu setting has been added to specify the menu for the structure tree.



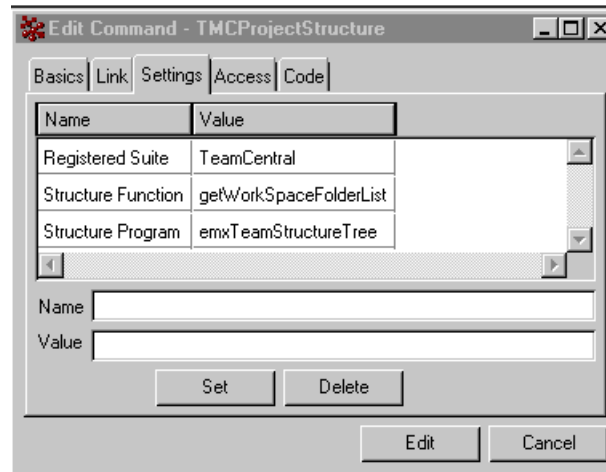
The admin menu that represents the structure tree is connected to one or more admin command objects. Each connected command has the necessary settings to generate a specific structure for that object. The settings define an inquiry or JPO that gets a list of business objects to display. If more than one command is connected to the structure menu, the Structure Selection combo list displays on the Structure Navigator to let users choose the structure they want to see (see [Structure Navigator](#)).

This is the structure menu for Team Central workspaces. The only structured objects for workspaces is folders so only one command is connected to the menu.



Using settings on the structure command objects, you can configure the label for the combo box, and the JPO or inquiry to be used for getting the object list when users navigate through the structure. For descriptions of the settings, see [Structure Tree Settings](#).

This graphic shows the settings for the command connected to the structure menu for Team Central workspaces.



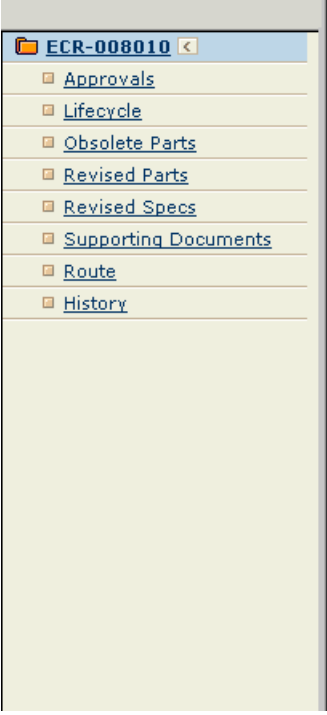
How Navigation Trees are Linked to an Application

In the dynamic user interface, users access navigation trees by selecting a business object from a table page—usually either the column that contains the object name or a popup details column. The JSP that constructs a tree must call the configurable JSP for trees, `emxTree.jsp`, plus any additional parameters, as described in [URL Parameters Accepted by `emxTree.jsp`](#). `emxTree.jsp` is the main JSP to control the display of any tree and it is located in `ematrix/common`. The new tree can be created in a separate popup window or inserted into the existing tree and is presented in the “content” frame.

By default, the `emxTree.jsp` gets the menu object that has the same symbolic name as the selected object’s type and is connected to the top-level tree menu called `Tree`. `emxTree.jsp` requires the ID for the selected object and this is passed to it using the `objectId` parameter. For example, if the selected object is a part, the page will look for a menu object named “`type_Part`”. If no such object exists, the page uses the tree defined for the object’s parent type. If there is no tree for the parent, the page looks for a tree defined for the grandparent, and so on. If the page reaches the top level type and finds no tree defined for it, it uses the default tree. (For information on using application-specific alternate trees and custom trees for an object, see [Calling Specific Trees for an Object Type](#).)

Steps for Building a Tree and Linking it to an Application

The procedure below lists the main steps for creating a tree and adding it to an application. The graphics illustrate the components that would need to be defined for a navigation tree for ECRs. The resulting tree would look something like the following.



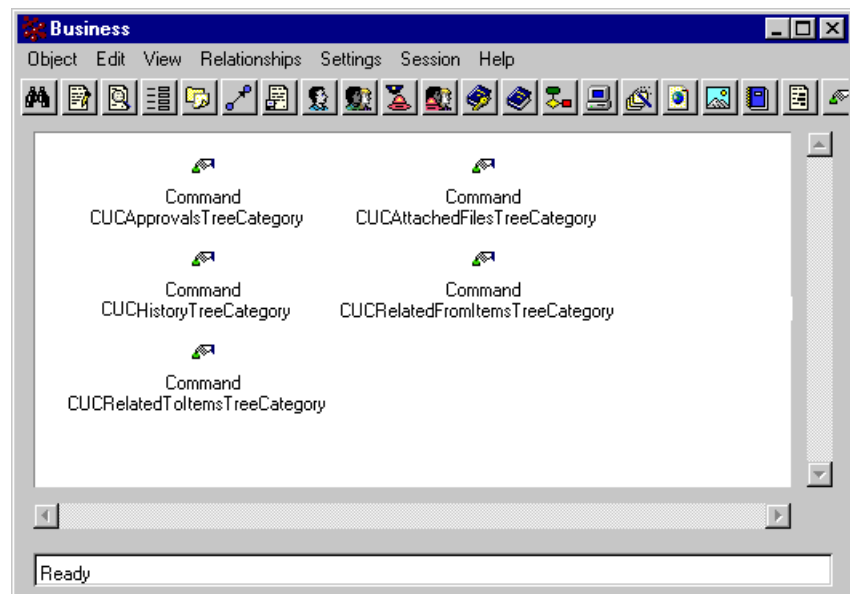
ECR-008010 rev -: Properties

Actions	
Type	ECR
Name	006000
Revision	-
Vault	eService Sample
Description	
Owner	Test DesignEngineer
General Description of Change	Move dimension in zone A-4
Reason for Change	To fix problem in over heating.
Reviewers Comments	
Category of Change	Drafting Error
Responsible Design Engineer	Test1 DesignEngineer
ECR Evaluator	Test1 DesignEngineer
Originator	Test DesignEngineer

To build a tree for an object type and hook it into an application

1. Create a command object for every category that should be included in the tree for that object type. A command object can be used in multiple menus and trees so if a command has already been created for the category, you don't have to create another command object.

When you create a command for a tree category, you define the page that should appear when a user clicks the tree category, how the page should appear, and who can access it. For a description of how to fill in the parameters and settings for tree category commands, see [Parameters for Tree Category Command Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

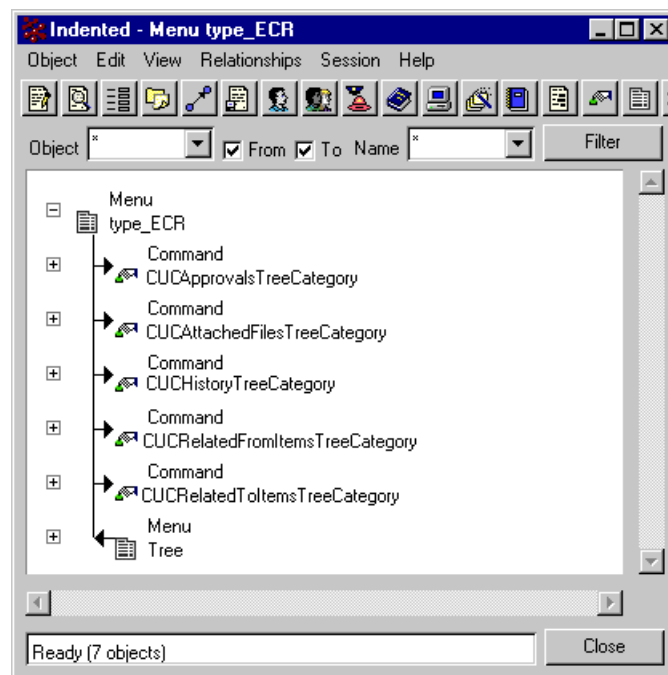


2. Create a menu object for the tree and make sure the object's name is the symbolic name of the type the tree is for. For example, the tree for parts is named type_Part. (Using the symbolic name for the type's tree menu object is the standard method for defining trees. The system will look for this menu automatically. Although not recommended, you can use the treeMenu parameter for emxTree.jsp to create a custom tree menu for an object type and override the standard tree menu. For object types used in multiple applications--such as Part, Buyer Desk, Person-- applications use alternate trees. For more information, see [Calling Specific Trees for an Object Type](#).)

When you create a menu object for a tree, you define how the tree should appear and the categories that should be included in it. For a description of how to fill in the parameters and settings for tree menu objects, see [Parameters for Tree Menu Objects](#).

3. Assign the command objects for the tree categories to the menu object for the tree, which you created in the previous step. In Business Modeler, the order the commands are listed in the Items tab of the Edit Menu dialog box are the order they appear in the tree. You can drag and drop the commands as needed to change their sequence.
4. To assign a sub-tree within the tree, assign the menu object that represents the sub-tree to tree menu object.
5. Connect the menu object for the tree, which you created in Step 2, to the Tree menu object installed with the framework.

This graphic shows the menu object for ECRs with the connected command objects for categories and the connected Tree menu object.



6. Make sure the type that the tree is for has an icon defined for it in emxSystem.properties (or you can use the default icon.) For information, see [Configuring the Icon for a Type's Tree](#).
7. In the JSP that needs to construct the tree, call emxTree.jsp and specify the parameters as needed to display the tree and associated page. Here are some example href values for updating a tree from a JSP. For a description of the parameters, see [URL Parameters Accepted by emxTree.jsp](#).

```

<a href="../common/
emxTree.jsp?objectId=<%=sPartId%>&mode=insert&jsTreeID=<%=
jsTreeID%>" class="object" target="content"><%=sPartName%></a>
<a href="../common/emxTree.jsp?objectId
=<%=OID%>&mode=replcae&jsTreeID=<%= jsTreeID%>" class="object"
target="content"><%=sRTSname%></a>
<a href="common/emxTree.jsp?objectId
=<%=sPartId%>&mode=insert&jsTreeID=" class="object"
target="content"><%=sPartName%></a>
<a href=javascript:window.open("../common/emxTree.jsp?objectId
=<%= sPartId%>&mode=insert&jsTreeID=<%=jsTreeID%>")
"><%=sPartName%></a>

```

To insert the tree into an existing tree, set the target frame to “content” frame, which is the parent frame for the “treeDisplay” frame. The first three examples above specify the content frame. Alternatively, you can open the tree in a new window, as shown in the last example. In this case, you don’t specify a target frame.

Whenever a new tree is presented (inserted or replaced) in the tree frame, the right side frame, called “detailsDisplay”, gets automatically updated with the URL, which is set in the “href” parameter of the menu object associated with the tree.

8. If you are working with the Web-based user interface as you are making changes and want to see your changes in the user interface, click the **Reload Cache** tool in the global toolbar and click the browser **Refresh** button.

The cache is refreshed automatically when the component age expires. This setting is in `emxSystem.properties`.

Only persons assigned to the Administration Manager role have access to the Reload Cache tool.

Tree JavaScript API for Custom JSPs

This section lists JavaScript APIs that can be used in custom JSP applications for manipulating, changing, deleting, and adding nodes for a tree. The methods cover trees that display in the Context Navigator, referred to as details tree, and tree that display in the Structure Navigator, referred to as the structure tree.

To make sure all custom tree code is supported by future releases of the framework, use only the following APIs for any tree operation. Any API not listed below should be considered private and is not intended for use in custom JSPs.

If you are upgrading an application from a pre-version 10 framework to version 10 or higher and have customized the code for navigation trees, you may have to migrate some code. For details, see [Migrating to Version 10 Tree API for Custom Pages](#) in Appendix A.

Obtaining the Details Tree Object

The details tree is always contained in the topmost frameset of the application. To get a reference to the details tree object, use the following:

```
var objDetailsTree = top.objDetailsTree;
```

The following method is deprecated. If existing custom code uses it, the code will work for now. The method will be eliminated eventually though, so it should no longer be used.

```
var objDetailsTree = top.tempTree;
```

Obtaining the Structure Tree Object

The structure tree is always contained in the topmost frameset of the application. To get a reference to the structure tree object, use the following:

```
var objStructureTree = top.objStructureTree;
```

Getting the Selected Node

To retrieve the currently selected (highlighted) node in either tree (Context Navigator or Structure Navigator), use the following:

```
var objSelectedNode = [Tree].getSelectedNode();
```

Getting a Node

There are a variety of ways to get references to nodes, each returning either an `emxUIObjectNode` or `emxUICategoryNode`. If you have the node's `nodeID`, you can use:

```
var objNode = [Tree].getNode(String nodeID);
```

To retrieve a node by its object ID, use this:

```
var objNode = [Tree].findNodeByObjectID(String objectID);
```

Note that this method only returns the first node in the tree with this object ID.

In order to retrieve a node by its name, use this code:

```
var objNode = [Tree].findNodeByName(String name);
```

Getting a Node's nodeID

The `nodeID` is an internal unique identifier for a node. To retrieve it, use:

```
var strNodeID = objNode.nodeID;
```

Getting a Node's Parent Node

The parent node object of any given node can be obtained using the method `getParent()`. To obtain the parent node, using the `jsTreeID` (`nodeID`), use:

```
var parentNode = [Tree].getNode(nodeID).getParent();
```

To obtain the parent node for the current selected node, use:

```
var parentNode = [Tree].getSelectedNode().getParent();
```

Getting Current Tree Root

The current tree root is the root that is currently at the top of the tree, though not necessarily the original root of the tree. To get the current root node, use:

```
var objRootNode = [Tree].getCurrentRoot();
```

Getting Original Tree Root

The original tree root is the root of the tree when it was first loaded. The current root and the original root are equal when the tree is first loaded. To get the original root node, use:

```
var objRootNode = [Tree].getOriginalRoot()
```

Getting Tree Current Root object ID

To get the object ID of the current root, use:

```
var objRootNode = [Tree].getCurrentRoot()  
var strObjID = objRootNode.objectID;
```

Getting Tree Original Root object ID

To get the object ID of the original root, use:

```
var objRootNode = [Tree].getOriginalRoot()  
var strObjID = objRootNode.objectID;
```

Setting the Selected Node

To set the currently selected (highlighted) node in the details tree, use:

```
objDetailsTree.setSelectedNode(emxUIDetailsTreeNode node,  
    boolean refresh);
```

To set the currently selected (highlighted) node in the structure tree, use the following code:

```
objStructureTree.setSelectedNode(emxUIStructureTreeNode node,  
    boolean refresh);
```

Changing the Name of All Nodes Representing a Business Object

To change the name of all nodes representing a business object (for instance, all nodes that point to Part-5000), use:

```
[Tree].getObject(String objectID).changeName(String newName,  
    boolean refresh);
```

The following method is deprecated. If existing custom code uses it, the code will work for now. The method will be eliminated eventually though, so it should no longer be used.

```
[Tree].getObject(String objectID).changeObjectName(String  
newName,  
    boolean refresh);
```

Changing the Object ID of All Nodes Representing a Business Object

To change the object ID of all nodes representing a business object (for instance, all nodes that point to Part-5000), use:

```
[Tree].getObject(String objectID).changeID(String newObjectID,  
    boolean refresh);
```


The following method is deprecated. If existing custom code uses it, the code will work for now. The method will be eliminated eventually though, so it should no longer be used.

```
objNode.changeObjectID(String newObjectID, boolean refresh);
```

Determining if an Object Exists in the Tree

To change the object ID of all nodes representing a business object (for instance, all nodes that point to Part-5000), use:

```
boolean result = [Tree].hasObject(String objectID);
```

Deleting All Nodes Representing a Business Object

To delete all nodes representing a business object (for instance, all nodes that point to Part-5000), use:

```
[Tree].deleteObject(String objectID, boolean refresh);
```

Removing a Single Node

There are two ways to remove a single node: using its nodeID or its objectID. To delete using the nodeID, use:

```
objParentNode.removeChildByID(String nodeID, boolean refresh);
```

To remove a node by using the object ID, use:

```
objParentNode.removeChild(String objectID, boolean refresh);
```

Inserting a New Node into Details Tree

To insert a new node (after creating/connecting a new object) in the details tree, use the following methods:

```
top.addDetailsTreeNode(String objectId, String jsTreeID);  
top.addDetailsTreeNode(String objectId, String jsTreeID,  
                        String emxSuiteDirectory);  
top.addDetailsTreeNode(String objectId, String jsTreeID,  
                        String urlParams);
```

This method adds the new node along with the categories defined in the admin menu object. The menu used for building the new tree is based on the objectId.

emxSuiteDirectory is an optional parameter to insert an application-specific alternate tree.

urlParams is an optional parameter to add any additional URL parameters like “treeLabel=TaskNote&emxSuiteDirectory=engineeringcentral”.

The following method is deprecated. If existing custom code uses it, the code will work for now. The method will be eliminated eventually though, so it should no longer be used.

```
top.addContextTreeNode(String objectId, String jsTreeID);  
top.addContextTreeNode(String objectId, String jsTreeID,  
                        String emxSuiteDirectory);
```

Inserting a New Node into Structure Tree

To insert a new node (after creating/connecting a new object) in the structure tree as well as the details tree in the Context Navigator, use the following methods:

```

top.addStructureTreeNode(String objectId, String
parentObjectId,
    String jsTreeID);
top.addStructureTreeNode(String objectId, String
parentObjectId,
    String jsTreeID, String emxSuiteDirectory);
top.addStructureTreeNode(String objectId, String
parentObjectId,
    String jsTreeID, String urlParams);

```

emxSuiteDirectory is an optional parameter to insert an application-specific alternate tree.

urlParams is an optional parameter to add any additional URL parameters like “treeLabel=TaskNote&emxSuiteDirectory=engineeringcentral”.

The following method is deprecated. If existing custom code uses it, the code will work for now. The method will be eliminated eventually though, so it should no longer be used.

```

top.addStructureNode(String objectId, String parentObjectId,
    String jsTreeID);
top.addStructureNode(String objectId, String parentObjectId,
    String jsTreeID, String emxSuiteDirectory);

```

Deleting Nodes in both Details and Structure Trees

To delete all nodes representing a business object (for instance, all nodes that point to Part-5000) in both the structure and details trees, use:

```
top.deleteObjectFromTrees(String objectId, Boolean refresh);
```

The following method is deprecated. If existing custom code uses it, the code will work for now. The method will be eliminated eventually though, so it should no longer be used.

```
top.deleteStructureNode(String objectId, Boolean refresh);
```

Deprecated Methods

The following table lists deprecated methods for navigation trees. These methods will work for now but should not be used for any future implementations.

Deprecated Method Name	Replaced by New Method
var objDetailsTree = top.tempTree;	var objDetailsTree = top.objDetailsTree;
changeObjectName(String newName, boolean refresh);	changeName(String newName, boolean refresh);
changeObjectID(String newObjectID, boolean refresh);	changeID(String newObjectID, boolean refresh);
top.addContextTreeNode(String objectId, String jsTreeID); top.addContextTreeNode(String objectId, String jsTreeID, String emxSuiteDirectory);	top.addDetailsTreeNode(String objectId, String jsTreeID); top.addDetailsTreeNode(String objectId, String jsTreeID, String emxSuiteDirectory); top.addDetailsTreeNode(String objectId, String jsTreeID, String urlParams);

Deprecated Method Name	Replaced by New Method
top.addStructureNode(String objectId, String parentObjectId, String jsTreeID); top.addStructureNode(String objectId, String parentObjectId, String jsTreeID, String emxSuiteDirectory);	top.addStructureTreeNode(String objectId, String parentObjectId, String jsTreeID); top.addStructureTreeNode(String objectId, String parentObjectId, String jsTreeID, String emxSuiteDirectory);
top.deleteStructureNode(String objectId, Boolean refresh);	top.deleteObjectFromTrees(String objectId, Boolean refresh);

Parameters for Tree Menu Objects

This table describes how to fill in the parameters and settings for menu objects that represent navigation trees for object types. For specific instructions on how to create menu objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Alt	Not applicable to menu objects for trees because the root node for a tree doesn't have a ToolTip.	—
Commands (specified in the Items tab in Business Modeler)	The command objects that represent the categories that should be in the tree. The order of the commands is the order the categories appear in the trees in the user interface.	Names of command objects, such as: ENCBOMTreeCategory AEFRelToItemTreeCategory TMCRoutesTreeCategory ENCPartDesignInfoTreeCategory
href	The URL that gets executed when a user clicks the root node for the tree, which is the Properties page for the object by default. This page is always displayed in the right frame, which is called “detailsDisplay” and not in another window or frame. The value for the href parameter should be a JSP and any associated parameters. You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros .	emxDynamicAttributes.jsp \${COMMON_DIR}/ emxPartInfo.jsp \${SUITE_DIR}/ emxViewPartAttr.jsp
Icon	Icon for the object within the Matrix core. This setting is NOT the image that displays next to the root node on the dynamic user interface. The emxFramework.smallIcon.SYMBOLIC_NAME property in emxSystem.properties is used to specify that image. For more information, see Configuring the Icon for a Type's Tree .	The name of an image file, such as Part.gif.

Parameter	Description	Accepted Values/Examples
Label	The text that should be used to label the tree root node. The label can contain be defined using: - Any of the regular expression macros supported by applications are evaluated at run time to display the current object type name and revision. - Static text to be displayed. - String resource ID defined in the string resource property file. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components. The system initially considers every label text as a string resource ID. If it finds the value for this ID in the resource file, it displays the obtained value. If not, the text is displayed as is. There are two other ways to define the label for a tree node: using the TreeLabel URL parameter passed to emxTree.jsp and using a JPO (specified with the Label Program and Label Function settings). The order of precedence in which the tree handles these methods of getting the root node label is: 1. TreeLabel URL parameter—Overrides all other methods. 2. JPO—Used if TreeLabel parameter is not defined. 3. Label parameter—Used only if TreeLabel and JPO are not defined.	<code><type> <name></code> <code><attribute[attribute_Weight].value ></code> Engine - <code><type> <name></code> Connected ECR emxFramework.Common.Part
Menus	Not applicable to menu objects for trees because trees do not have submenus.	—
Settings	Additional settings that define the behavior and appearance of the tree.	Name/value pairs, as defined in the table below.

Settings for Tree Menu Objects

This table lists and describes the settings for menus that represent navigation trees. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
Currency Converter	Specifies whether the target page should include the Currency Converter tool. In the href URL called when the tree root node is clicked, the system passes a parameter called CurrencyConverter and includes the value specified for this setting. If the value is True, the target page includes the Currency Converter tool. If this setting is not included, the value is assumed to be false and the target page does not display the Currency Converter tool.	True False
Default Category	Specifies the category that should be selected when the tree first opens or is first inserted into another tree. By default, the root node is selected, which means the object's Properties page displays. If the tree contains a category that users frequently want to see, you can use this setting to have it selected instead. Alternatively, you can pass the DefaultCategory parameter to emxTree.jsp. If both are defined, this URL parameter overrides the setting. Note that a tree does not look for default categories defined for its sub-trees (assigned sub-menus).	The name of a category command object that should be selected when the tree first opens. The command object must be assigned to the tree menu object and if it is not, the root node is selected. Default Category=PMCWBS
Help Marker	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help . In the href URL called when the tree root node is clicked, the system passes a parameter called HelpMarker and includes the marker text specified for this setting.	The naming convention for help markers "emxhelp" followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
Label Function	Specifies the function in the JPO specified in the Label Program setting that gets the value for the root node label.	The name of a method in the Label Program JPO.
Label Program	Use to define the label for the root node using a JPO. This parameter specifies a JPO that contains a method to get the value for the label. The method is specified using the Label Function setting. The input for this JPO must include: –A list of all of the Request Parameters in a HashMap –Method Name as a string –JPO Program Name as a string The output should be the string that is returned to the tree and used for the label. There are two other ways to define the label for a tree node: using the TreeLabel URL parameter passed to emxTree.jsp and using the Label parameter for the menu object. For information on the precedence order, see the Label parameter.	The name of a JPO that has been added to the database. <i>If a JPO is configured for a tree category or root node label and the JPO returns an empty string or null, an exception is thrown.</i>

Setting	Description	Accepted Values/Examples
Message URL Label	Specifies the label to use for URL links that call the tree. For example, notifications that users have new tasks include URL links that open the Inbox Task tree for the user's task. By default, the URL link label shows the name of the Inbox Task object, which is an autogenerated name. But if the route creator entered a name for the task, it's better to show that entered name for the URL link. To have the URL link show the task's entered name, you would add the Message URL Label setting to the tree menu object for the Inbox Task. The value would be a select macro for the attribute that contains the entered name, in this case the Title attribute. When building the URL link, the system first looks for the Message URL Label setting on the app specific tree menu, for example, ENctype_InboxTask. If not found there, the system looks for the setting in the common tree for that object type. If the common tree doesn't contain the setting, the system defaults to the object's Type,Name,Revision (for example, Inbox Task IT-0000101).	The value can be any select statement, plain text or a string resource ID. For example: • <code><attribute[attribute_Title].value></code> • Task Title • <code>emxEngineeringCentral.TaskTitle</code> The string resource Id can contain macros also, as in the following example: <code>emxEngineeringCentral.TaskTitle=</code> <code><type> <name> <revision>: Urgent</code> Task
Printer Friendly	Passes the PrinterFriendly parameter and the entered value to the JSP specified in the href URL. JSPs that use the PrinterFriendly parameter, such as <code>emxTable.jsp</code> and <code>emxForm.jsp</code>, show the Printer Friendly tool when the setting is True and hide it when False. If the setting is not included, <code>emxTable.jsp</code> and <code>emxForm.jsp</code> show the tool by default. Users can click the tool to get a version of the current page that can be printed with the browser's Print button. Note that you can also specify the PrinterFriendly parameter in the href URL for JSPs that use it.	True False
*Registered Suite	The application the menu belongs to. The system looks for files related to the tree in the registered directory for that application, which is specified in <code>emxSystem.properties</code>. Based on the application name, the system passes the following parameters in the href URL: • suiteKey • emxSuiteDirectory • StringResourceId	The value must be set without any spaces, for example, <code>EngineeringCentral</code> or <code>Framework</code>. The value must be set to the suite name as defined in the key <code>eServiceSuites.DisplayedSuites</code> within <code>emxSystem.properties</code>. If the suite name starts with "eServiceSuite" then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in <code>emxSystem.properties</code> is "eServiceSuiteEngineeringCentral", then the word "EngineeringCentral", can be assigned as "Registered Suite". In the href URL that is called when the tree root node is clicked, the system passes a parameter called "suiteKey". The value for the parameter is the property name from <code>emxSystem.properties</code> that maps to the setting's value.
Structure Menu	Specifies the menu admin object that defines a structure tree for the object type.	The name of a menu admin object.

Setting	Description	Accepted Values/Examples
Tip Page	Specifies whether the target page should include the Tip Page tool and the URL to call when the tool is clicked. In the href URL called when the tree root node is clicked, the system passes a parameter called TipPage and includes the URL specified for this setting. If this setting is not included, the target page does not display the Tip Page tool.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the table.jsp: TipPage=../doc/customcentral/tippage.html
Tree Scope ID	Passes the current tree's objectId to all its categories, to subtrees inserted into the tree, and to categories in the subtrees, to the nth level, in the parameter name specified as the value for this setting. This setting is valid only for tree menu objects and not for tree category command objects. For example, suppose you want to pass the object ID for the current workspace to all categories and subtrees in the workspace tree. You want to pass the object ID using a parameter called workspaceId. To do this, you would add the Tree Scope ID setting to the menu admin object for workspace trees (which is called type_Project due to admin object renaming) and enter "workspaceId" as the value. The parameter and value "workspaceId=OBJECTID OF THE WORKSPACE" will be available to all the categories and subtrees of the current workspace to the nth level until another workspace object gets inserted or until any explicit URL parameter with the same name "workspaceId" is passed to the tree.	The name of the parameter you want to use to pass the object ID. For example: workspaceId projectId partId <i>You can also pass specific parameters from the tree menu object to its categories using the AppendParameter parameter for the emxTree.jsp.</i>
*Required Setting		

Parameters for Tree Category Command Objects

This table describes how to fill in the parameters for command objects used for tree categories. These apply to command objects for standard navigation trees and for structure

trees. For specific instructions on how to create command objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/ Examples
Icon	Icon for the object within the Matrix core. This setting does not display an image in the UI.	The name of an image file, such as Part.gif.
Label	The text that should be used to represent the tree category. The label can be made up of: 1. Plain text, such as “ECRs”. 2. Text and/or any valid select expression that gets evaluated before displaying the label, such as “Subfolder (\$<attribute[attribution_Count].value>)”. For example, you can configure the label so it shows a count of the number of items in the category, to the right of the label and in parentheses. For example, if the category is subfolders of a folder and a particular folder has three subfolders, the label would show “Subfolders (3)”. To configure the label to include a count, use the macro \$<attribute[attribution_Count].value>. For example: <pre>Label = Subfolder (\$<attribute[attribution_Count].value>)</pre> The Count attribute is assigned to the Workspace Vault type (called folder in the UI). This attribute keeps the count of the number of subfolders created below the folder. At run time, this value is evaluated and is shown in the label for the tree category. 3. A string resource ID, such as “emxEngineeringCentral.DesignTOP.CreatePart”. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components. The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn’t find one, it displays the entered text and/or output from the select expression. To internationalize a label that needs a select expression, you can assign a string resource ID to the label and the string resource ID can contain the macro to be evaluated. For example, using the use case described in option 2 above, you could configure the command object label as follows: <pre>Label = emxFramework.Common.Subfolder</pre> And in string resource file: <pre>emxFramework.Common.Subfolder = Subfolder (\$<attribute[attribution_Count].value>)</pre> Tree category labels can also be defined using a JPO. See the Label Function and Label Program settings. If these JPO settings are defined for a category, they override any label defined using this parameter.	ECRs Subfolder (\$<attribute[attribution_Count].value>) emxEngineeringCentral.DesignTOP.CreatePart

Parameter	Description	Accepted Values/ Examples
href	The URL that gets executed when the tree category is clicked. This URL is always displayed in the right frame and not in another window or frame. The value for the href parameter should be a JSP and any associated parameters. You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros.	emxEcRs.jsp <code>\${COMMON_DIR}/emxEcRs.jsp</code> <code>\${SUITE_DIR}/emxEcRs.jsp</code> <code>\${ROOT_DIR}/emxEcRs.jsp</code>
Alt	Not applicable for tree categories because ToolTips do not display for categories.	—
Access	The persons, roles, and groups who can access the tree category. To make the tree category available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access.	Names of group, role, person administrative objects. or All (default)
Settings	Additional settings that define the behavior and appearance of the command. For a list of the accepted settings, see the table below.	Name/value pairs, as defined in the table below.

Settings for Tree Category Command Objects

This table lists and describes the settings for command objects used for navigation tree categories. Settings for commands in structure trees are listed in a separate table following this one. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
*Registered Suite	The application the command belongs to. The system looks for files related to the tree category in the registered directory for that application, which is specified in <code>emxSystem.properties</code>. Based on the application name, the system passes the following parameters in the href URL: • <code>suiteKey</code> • <code>emxSuiteDirectory</code> • <code>StringResourceFileId</code>	The value must be set without any spaces, for example, <code>EngineeringCentral</code> or <code>Framework</code>. The value must be set to the suite name as defined in the key <code>eServiceSuites.DisplayedSuites</code> within <code>emxSystem.properties</code>. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in <code>emxSystem.properties</code> is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the tree category is clicked, the system passes a parameter called “suiteKey”. The value for the parameter is the property name from <code>emxSystem.properties</code> that maps to the setting’s value.
Currency Converter	Specifies whether the target page should include the Currency Converter tool. In the href URL called when the tree category is clicked, the system passes a parameter called <code>CurrencyConverter</code> and includes the value specified for this setting. If this setting is not included, the value is assumed to be false and the target page does not display the Currency Converter tool.	True False
Expand Function	Specifies the function in the JPO specified in the Expand Program setting that gets the list of child nodes for the tree category.	Name of a method in the Expand Program JPO.
Expand Inquiry	Use to configure the tree category so a list of business objects is inserted under it automatically, without users having to first view details for objects. For example, Folder categories are often configured to display all folders without users having to view the folders first. Use this setting to define the list of business objects using an inquiry admin object. You can also define the list using a JPO with the Expand Program and Expand Function settings. If both an inquiry and JPO are specified, the JPO takes precedence. For more configuration options for this type of category, implications for performance, and rules for configuring the inquiry object, see Configuring Dynamic Expand for Tree Categories.	Name of an inquiry admin object that retrieves the business objects to list for the category: TMCFolder TMCSUBFolders ENCEBOMList
*Required Setting		

Setting	Description	Accepted Values/Examples
Expand Program	Use to configure the tree category so a list of business objects is inserted under it automatically, without users having to first view details for objects. For example, Folder categories are often configured to display all folders without users having to view the folders first. Use this setting to define the list of business objects using a JPO. You can also define the list using an inquiry with the Expand Inquiry setting. If both an inquiry and JPO are specified, the JPO takes precedence. This parameter specifies a JPO that contains a method to get the list of objects to insert under the category. The method is specified using the Expand Function setting. For more configuration options for this type of category, implications for performance, and inputs/outputs for the JPO, see Configuring Dynamic Expand for Tree Categories. Important Note: <i>This feature is recommended primarily for hierarchically organized categories such as Folders and Subfolders. For best performance and consistency of behavior across applications, the preferred method of navigation is to use the standard tree behavior that does not use a dynamic expand. If conditions require the use of this setting, try to restrict its use to categories that will contain relatively few objects.</i>	Name of a JPO program added to the database.
Help Marker	Specifies the name of the help marker to call for context-sensitive help. These markers are defined in a property file for the application's help system. For information about implementing help, see Implementing Context-Sensitive Help. In the href URL called when the tree category is clicked, the system passes a parameter called HelpMarker and includes the marker text specified for this setting.	The naming convention for help markers "emxhelp" followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
*Required Setting		

Setting	Description	Accepted Values/Examples
Image	Name of the image file name that should be displayed next to the label for the tree category. The value for the setting can be one of the following: - Any simple image (gif) file name without any prefix. The system assumes the image is referred from the current directory and prefixes it with “images/”. - An image file name prefixed with <code>\${COMMON_DIR}</code>. The system looks for the file in the images subdirectory of the directory designated as the common directory. The common directory is defined using the <code>eServiceSuiteFramework.CommonDirectory</code> property in <code>emxSystem.properties</code>. - An image file name prefixed with <code>\${SUITE_DIR}</code>. The system looks for the image file in the images subdirectory of the application specific directory. The application directory is defined using the <code>eServiceSuiteSUITENAME.Directory</code> property in <code>emxSystem.properties</code>. For example, the property for Engineering Central is <code>eServiceSuiteEngineeringCentral.Directory = engineeringcentral</code>.	<code>iconSmallPart.gif</code> <code>\${COMMON_DIR}/buttonReports.gif</code> <code>\${SUITE_DIR}/iconPerson.gif</code> <i>The current UI design does not include an image for tree categories so this setting is ignored.</i>
Label Function	Specifies the function in the JPO specified in the Label Program setting that gets the value for the tree category label.	The name of a method in the Label Program JPO.
Label Program	Use to define the label for the category using a JPO. This parameter specifies a JPO that contains a method to get the value for the label. The method is specified using the Label Function setting. The input for this JPO must include: –A list of all of the Request Parameters in a HashMap –Method Name as a string –JPO Program Name as a string The output should be the string that is returned to the tree and used for the label. You can also define the label using the Label parameter for the command object. The JPO label settings override any label defined in the Label parameter setting.	The name of a JPO that has been added to the database. <i>If a JPO is configured for a tree category or root node label and the JPO returns an empty string or null, an exception is thrown.</i>
*Required Setting		

Setting	Description	Accepted Values/Examples
PreExpand Category	Used in conjunction with the Expand Program and Expand Function settings or the Expand Inquiry setting, which inserts a list of objects under the tree category without requiring users to first view details of those objects. Use this setting to determine whether the tree pre-processes the JPO or inquiry to determine if there are any objects to be listed. Pre-processing offers the benefit of determining in advance that there are no business objects in the list so the + sign for expansion is not shown. But pre-processing lowers performance. For more information, see Configuring Dynamic Expand for Tree Categories. <i>Administrators who configure this setting as true need to be aware that there are performance implications to doing so. While it is nice to only display the + sign when data is present, the tree must actually go and get the data at the time the tree displays to do this. Therefore, if there are many tree categories configured with PreExpand Category set to true, each of those categories will be evaluated when the tree is displayed regardless of whether the user actually intends on viewing data in those categories or not. This potentially unnecessary processing lowers performance.</i>	true—Pre-processing is allowed so if there are no objects, the + sign for expanding the list is not shown. false (default)—Pre-processing is not allowed so the + sign is displayed even if there are no business objects in the list. The inquiry or JPO is not processes until the user clicks the + sign.
PreExpand Function	Specifies a method in the JPO defined in the Expand Program setting that configures the category for dynamic expand. This method determines whether there are any business objects in the business object list retrieved from the JPO, thereby determining whether the + sign should be displayed. <i>Use of this setting requires a change in Matrix core that has not yet been implemented.</i>	Name of a method in the Expand Program JPO.
*Required Setting		

Setting	Description	Accepted Values/Examples
Printer Friendly	Passes the PrinterFriendly parameter and the entered value to the JSP specified in the href URL. JSPs that use the PrinterFriendly parameter, such as emxTable.jsp and emxForm.jsp, show the Printer Friendly tool when the setting is True and hide it when False. If the setting is not included, emxTable.jsp and emxForm.jsp show the tool by default. Users can click the tool to get a version of the current page that can be printed with the browser's Print button. Note that you can also specify the PrinterFriendly parameter in the href URL for JSPs that use it.	True False
Tip Page	Specifies whether the target page should include the Tip Page tool. In the href URL called when the tree category is clicked, the system passes a parameter called TipPage and includes the URL specified for this setting. If this setting is not included, the target page does not display the Tip Page tool.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the table.jsp: TipPage=../doc/customcentral/tippage.html
*Required Setting		

This table lists and describes the settings for command objects used for structure tree categories. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
*Registered Suite	The application the command belongs to. The system looks for files related to the tree category in the registered directory for that application, which is specified in emxSystem.properties. Based on the application name, the system passes the following parameters in the href URL: • suiteKey • emxSuiteDirectory • StringResourceFileId	The value must be set without any spaces, for example, EngineeringCentral or Framework. The value must be set to the suite name as defined in the key eServiceSuites.DisplayedSuites within emxSystem.properties. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in emxSystem.properties is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the tree category is clicked, the system passes a parameter called “suiteKey”. The value for the parameter is the property name from emxSystem.properties that maps to the setting’s value.
Label	The label used to show the list of structures in the Structure Selector combo box. If there is only one command attached to the structure menu, no combo box is needed so the Label setting is ignored.	A string resource ID or text string: emxTeam.common.Folders Discussions EBOM
PreExpand	Determines whether to let the JPO program decide whether to add the [+] icon or to simply add the [+] without running the JPO.	true—The Structure Program and Function defined for the Structure command is run to get the list of objects. Then for each object, it runs the same program to determine whether a [+] icon is needed. If the list of objects is greater than 1, a plus [+] is added to the structure node. false (default)—The JPO Program and Function is run to get the list of objects, but the same program and function is not run again for each object. Instead, a plus icon [+] is automatically added for each structure node.
Structure Function	JPO method name to be used for getting the object list.	The name of a method in the JPO specified in the Structure Program setting, such as: getWorkspaceFolders getEBOMs
*Required Setting		

Setting	Description	Accepted Values/Examples
Structure Inquiry	Inquiry admin object name that gets the object list to display within the structure. This is an alternative to the JPO approach and the JPO approach takes precedence. <i>This setting is not implemented.</i>	The name of an inquiry admin object: SCSWspFolderStructure ENCEBOMList
Structure Program	JPO program that contains the method to be used for getting the object list to display within the structure. The method is specified in the Structure Function setting. For instructions on writing the JPO program, see Guidelines for Writing Structure Tree JPO.	The name of a JPO program added to the database. SCSWorkspaceStructure ENCBOMStructure
*Required Setting		

URL Parameters Accepted by emxTree.jsp

This table lists the parameters that you can specify for emxTree.jsp. When you specify emxTree.jsp to be called from within a JSP, you can add these parameters. Note that the name of the menu object for the tree is not required because the system automatically

looks for the tree menu object that has the same name as the symbolic name of the object's type.

Parameter	Description	Accepted Input Values
AppendParameters	Passes common parameters to the tree categories in the tree. To use the parameter, set it to true and then include the parameters to pass. Parameters added to emxTree.jsp that are not with AppendParameters are passed only to the root tree node and not to categories. <i>You can also use the Tree Scope ID setting for the tree menu object to pass the current tree's object ID to all categories and subtrees.</i>	To make Param1 and Param2 available to the children categories: <pre><a href='../common/emxTree.jsp?objectId=<%=sPartId%>&m ode=insert&jsTreeID=<%= jsTreeID %>&AppendParameters=true&Param1=Value1&Param2=Value2" class="object" target="content"><%=sPartName%></pre> To make ProjectId and Workspace available to the tree categories: <pre><a href='../common/emxTree.jsp?objectId=<%=sPartId%>&m ode=insert&jsTreeID=<%= jsTreeID%>&AppendParameters=true&ProjectId=<%=sProjId%>&Works pace=<%=sWorkspace%>" class="object" target="content"><%=sPartName%></pre> To make ProjectId and Workspace available only to the root tree node and not to the tree categories: <pre><a href='../common/emxTree.jsp?objectId=<%=sPartId%>&m ode=insert&jsTreeID=<%= jsTreeID %>&&ProjectId=<%=sProjId%>&Wo rkspace=<%=sWorkspace%>" class="object" target="content"><%=sPartName%></pre>
DefaultCategory	Specifies the category that should be selected when the tree first opens or is first inserted into another tree. By default, the root node is selected, which means the object's Properties page displays. If the tree contains a category that users frequently want to see, you can use this parameter to have it selected instead. Alternatively, you can use the Default Category setting for the tree's menu object. If both are defined, this URL parameter overrides the setting. Note that a tree does not look for default categories defined for its sub-trees (assigned sub-menus).	The name of a category command object that should be selected when the tree first opens. The command object must be assigned to the tree menu object and if it is not, the root node is selected. DefaultCategory=PMCWBS

Parameter	Description	Accepted Input Values
jsTreeID	The ID of the existing tree object, which is required to update the tree with “insert” mode. - When the tree is constructed for the first time, the “jsTreeID” parameter does not have any valid value. - Whenever a category within the tree is clicked, the system passes the current “jsTreeID” to that page and if that page needs to update the tree, it must call emxTree.jsp with the same “jsTreeID” value.	node567590204694.5947
mode	Controls whether the tree for the object is inserted into the current tree or replaces the current tree. For information, see Two Modes for Tree Display.	Insert (default)—Insert the tree for the selected object into the current tree, if there is one. Replace—Replace the current tree, if there is one, with the tree for the selected object.
objectId	This is the business object ID for which the tree will be constructed. For example, if the object is of type ECR, the tree object of menu type “type_ECR” will be constructed. If the menu “type_ECR” does not exist, the tree will be based on the “Default Tree” configuration.	2233.5567.2323.4678
treeLabel	Use to override the default label for a tree, which is defined in the “Label” parameter of the tree menu admin object. When this parameter is available to the emxTree.jsp page, the page uses this value as the Label for tree menu base node. This parameter also overrides any JPO specified for the label in the settings Label Program and Label Function.	emxTree.jsp?treeLabel=custom Part Label&objectId=3434.345.4564.7755
treeMenu	Use to override the standard tree menu for the object’s type (or any alternate tree defined by an application). The standard tree menu for an object is named using the symbolic name of the type. For example, the standard tree menu object for quality part plans is type_QualityPartPlan. For more information on calling specific trees, see Calling Specific Trees for an Object Type. <i>Using the custom tree menu by overriding the system-defined tree for any object type is NOT RECOMMENDED.</i>	Name of a menu object for a tree: customPartTree

Using and Configuring the Default Tree

The framework installs with a default tree that contains general categories used by many types. The system uses the default tree for any type that doesn't have a tree defined for it (or any type higher in the hierarchy) in Matrix. For details on how the system determines which tree menu object to use for an object, see [Calling Specific Trees for an Object Type](#).

You specify the tree to use as the default tree using the following property in `emxSystem.properties`. The value for the property should be the symbolic name of the menu object that you want to use as the default tree.

```
#Setting to pick up default tree name
emxNavigator.DefaultTree.TreeName = menu_DefaultTree
```

You can configure the default tree as you would any tree: change the categories by changing the connected command objects, changing the image for it, and changing parameters and settings for the root node and command objects. For more information, see [Building a Navigation Tree](#).

Calling Specific Trees for an Object Type

Typically, when a user clicks an object's name in the MatrixOne applications to see details about the object, `emxTree.jsp` uses the tree menu object that has the same symbolic name as the object's type. For example, when a user clicks the name of a Part Quality Plan, the system uses the tree menu object named `type_PartQualityPlan`. We'll refer to the menu object that matches the symbolic name for a type as the standard tree menu for an object.

There are cases when the standard tree menu for an object isn't suitable. For example, some object types are used by more than one MatrixOne Application and each application may have a unique tree menu. In this case, the application that "owns" that object type installs the standard tree menu object and all other applications that use the type install an alternate tree menu for the type. The "owner" application is the one that has primary responsibility for creating and managing that type of object. For example, most MatrixOne applications use parts but Engineering Central is the owner of the Part type. Therefore, Engineering Central installs the `type_Part` menu and other applications install an alternate menu for parts. Specific installations may also need to define custom tree menus. (Using custom tree menus instead of the system's standard or pre-defined alternate menu is not recommended.)

When a user clicks a link to view details for an object, the system uses the following methods to determine which tree to display for the object, in this order of precedence:

1. **Override all trees using `treeMenu` parameter**—The system first looks for the `treeMenu` parameter passed to `emxTree.jsp` URL. The value for the `treeMenu` parameter is the name of a menu object for a navigation tree and the parameter can be used to override all other trees that could be used for the object.

Use the `treeMenu` parameter when you want to use a tree menu other than the standard (as described in item 3 below) or alternate (as described in item 2 below) tree menu for the object's type. Also use the `treeMenu` parameter to call an application's alternate tree menu for an object when the link is external to the application. For example, if the link is within IconMail or email, the system will not pick up an application's alternate menu because the link isn't within a particular application (so it wouldn't know which application's alternate tree menu to use).

2. **Use application's alternate tree**—If no treeMenu parameter is passed, the system gets the suite directory that called emxTree.jsp and checks if that application has an alternate tree menu defined for the object type. MatrixOne applications define alternate trees for objects used by more than one application. For example, parts, persons, and buyer desks are all used by multiple applications and each application has its own alternate tree menu. The alternate tree menus need not be assigned to the top-level menu called Tree but the standard tree menus must be.

Application-specific alternate menus are defined using the property SUITEKEY.emxTreeAlternateMenuName.TYPE_SYMBOLIC_NAME in the application's property file, such as emxEngineeringCentral.properties. SUITEKEY is the same as the value assigned to the property "eServiceSuites.DisplayedSuites" in the emxSystem.properties file. The naming convention for the alternate menus is the type's symbolic name prepended with the 3-letter abbreviation for the application, for example, ENCtype_Part. If the specified alternate menu does not exist in the database, the standard tree menu for that type, for example "type_Part", is used. (The naming conventions for the application abbreviations are listed in [Naming Conventions for UI Administrative Objects](#).)

For example, the emxEngineeringCentral.properties file contains this property:

```
eServiceSuiteEngineeringCentral.emxTreeAlternateMenuName.type_Part=ENCtype_part
```

In special cases, such as when forwarding or redirecting the response to emxTree.jsp, the alternate tree menu may not display correctly. In these cases, the URL parameter "emxSuiteDirectory" must be passed in with the appropriate directory to emxTree.jsp. This directory is the same as defined in the emxSystem.properties (SUITE_NAME.Directory = ...), where the value is the application directory installed below the ematrix directory.

3. **Use type's standard tree**—If there is no application-specific alternate menu defined for the object type, the system looks for a menu object with the same name as the type's symbolic name. If none is found, it looks for a menu object with the symbolic name for the type's parent type, then grandparent type, and so on until it reaches the top of the type hierarchy.
4. **Use Default Tree**—If there is no menu object for the object's type or parent types, the system uses the Default Tree menu object. For more information on the default tree, see [Using and Configuring the Default Tree](#).

Configuring the Icon for a Type's Tree

When displaying the tree for an object type, the system uses the image file specified in the emxSystem.properties file for that object type. For example, this property defines the icon for a mechanical part:

```
emxFramework.smallIcon.type_MechanicalPart =  
iconSmallMechanicalPart.gif
```

This property is also used to display an icon for the type (in addition to the column's data) in a table column when the Show Type Icon setting is true for the column. If there is no property defined for the type's icon, the system looks for a property for the type's parent type icon, then grandparent's type and so on. If no property is defined for any parent type's icon, the system uses the default image, defined by this property:

```
# default icon name for the type  
emxFramework.smallIcon.defaultType = iconSmallDefault.gif
```

One exception to the above is that if the system uses a parent's tree menu because no tree menu is defined for the subtype, then the icon associated with the parent type is used instead of the subtype.

Configuring Dynamic Expand for Tree Categories

You can configure a tree category so a list of business objects is inserted under it automatically, without users having to first view details for objects. For example, Folder categories are often configured to display all folders without users having to view the folders first.

This is different from the normal behavior for tree categories which displays the + only after a user has viewed the details for an object within the category. Without the dynamic expand, when the user clicks the + sign for the category, the tree expands to show only the objects that have been viewed. For example, the + only appears next to the Routes category after the user views details about a route. When the user clicks the +, the tree expands to show only the viewed route(s).

Important Note:

This feature is recommended primarily for hierarchically organized categories such as Folders and Subfolders. For best performance and consistency of behavior across applications, the preferred method of navigation is to use the standard tree behavior that does not use the Expand Inquiry. If conditions require the use of this setting, try to restrict its use to categories that will contain relatively few objects.

There are two ways to get the list of business objects to insert under the category:

- **JPO and method**
The objects are obtained using a method in a JPO. You specify the JPO using the Expand Program setting and you specify the method using the Expand Function setting for the command object. If both the JPO and inquiry settings are added for a category command object, the JPO takes precedence.
- **Inquiry admin object**
The objects are obtained from the database using an inquiry object. You configure this behavior using a setting called Expand Inquiry for the command object that represents the category. The value for the setting is the inquiry object name. For example:
`Expand Inquiry=TMCFolders`

By default, the JPO or inquiry is not run until the user clicks the + sign. This means that the JPO or inquiry could retrieve 0 business objects, even though the category displays the + sign, implying that there are objects to display. The setting PreExpand Category=true can be used to run the JPO or inquiry as soon as the tree displays so the + can be hidden if there are no business objects. However, pre-processing can impact performance.

Requirements for Dynamic Expand JPO

To specify a JPO and method that gets the list of business objects to insert under a dynamic expand tree category, use the Expand Program and Expand Function settings for the command. The input for the JPO must include:

- A list of all of the Request Parameters in a HashMap
- Method Name as a string

–JPO Program Name as a string

The output should be a maplist that contains the list of business object IDs to process and insert under the category. The list of IDs determines which types of object are being processed and which tree to use for the objects.

Examples of Dynamic Expand Inquiries

You can create the required inquiry admin object using MQL scripts or Business Modeler. Examples for creating the inquiry by MQL are shown below. For a description of the parameters available for inquiry objects, see [Parameters and Settings for Inquiry Objects](#).

Inquiry Example 1: TMCFolders

```
*****
# Create the inquiry object for Folder List.
add inquiry TMCFolders
code "expand bus \"${ID}\" from relationship \"${WS_VAULT}\"
select bus id dump |
where '( (owner == \"${USER}\") || (current.access[read] ==
true))'
    argument WS_VAULT relationship_ProjectVaults
    pattern *|*|*|*|*|*|${OID}
    format ${OID}
*****
```

Inquiry Example 2: ENCPartList

```
*****
# Create the inquiry object for Part List Table.
add inquiry ENCPartList
    code "temp query bus ${PART} * * select id dump |"
    argument PART type_Part
    pattern *|*|*|${OID}
    format ${OID}
;
*****
```

Inquiry Example : ENCBOMList

This example, ENCBOMList, uses \${ID} as a macro but this is not same as the Runtime Program Environment (RPE) variable \${OID}. This \${ID} macro gets substituted at run time, if there is a valid object Id available to emxTree.jsp.

```
*****
# Create the inquiry object for BOM Table for a specific Part
add inquiry ENCBOMList
code "expand bus '${ID}' from relationship '${EBOM_REL}' select
businessobject id select relationship id dump |"
    argument EBOM_REL relationship_EBOM
    argument ID dummy
    pattern *|*|*|*|*|*|${BUSID}|${RELID}
    format ${RELID}~${BUSID}
;
*****
```

A JPO defined for a structure tree category must have the business logic to fetch the object list required for structure navigation.

JPO Method Signature and Input Argument

The JPO class must have the `mxMain` method implemented. The JPO method is defined with the following signature.

```
public static MapList methodName(matrix.db.Context context,
String[] args)
throws Exception
{
}
}
```

The input arguments:

- Context—passed in always.
- String[] args—has one element of object type HashMap.

Returns: A Map List containing the object list.

This input argument is a HashMap, which contains the details of the object to be processed. The data structure of this input parameter (HashMap) is defined in this table.

Key Name	Data Type	Description
objectId	String	Assigned to the business object Id (OID) to be used in the current form page.
relId	String	Assigned to the relationship id (RelID) to be used for the form page.
requestMap	HashMap	Assigned to a HashMap, which contain a set of key/value pairs as available in the request object. The keys are of type String (parameter names) and the values are of type String (parameter value).The request Map will contain an additional key/value parameter for language setting. The key “languageStr” is assigned to a value obtained by the client using “request.getHeader(“Accept-Language”)”.

The JPO method can extract the information from the input argument for processing the data.

```
public static Object MapList (matrix.db.Context context,
String[] args) throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    MapList objectList = new MapList();

    // Get the required parameter values from "programMap" - as
    required
```

```

String  objectId = (String) programMap.get("objectId ");
String  relId = (String) programMap.get("relId ");
String  languageStr = (String) requestMap.get("languageStr");
String  workspaceId = (String) requestMap.get("workspaceId");
String  suiteKey = (String) requestMap.get("suiteKey");

...

return objectList
}

```

The structure tree can also be configured to display the BOM structure for a part object. For example, the following code snippet is a JPO that can be configured to display the EBOM structure tree for a part.

```

public MapList getEBOMs (Context context, String[] args)
    throws Exception
{

    HashMap programMap = (HashMap) JPO.unpackArgs(args);
    HashMap paramMap = (HashMap) programMap.get("paramMap");

    String partId = (String) paramMap.get("objectId");
    MapList ebomList = new MapList();

    try {
        Part partObj = new Part(partId);
        StringList selectStmts = new StringList(1);
        selectStmts.addElement(partObj.SELECT_ID);
        selectStmts.addElement(partObj.SELECT_TYPE);
        StringList selectRelStmts = new StringList(1);

        selectRelStmts.addElement(partObj.SELECT_RELATIONSHIP_ID);
        ebomList =
        partObj.getEBOMs(context, selectStmts, selectRelStmts, false);
    }

    catch (FrameworkException Ex) {
        throw Ex;
    }

    // System.out.println("ebomList : " + ebomList);
    JPO.packArgs(ebomList);
    return ebomList;

}

```

In both cases, the method returns an object of type MapList.

The MapList will hold the list of object ids. Each object id will be defined within a HashMap. The HashMap will have a key “id” and assigned to an OID.

Optionally, the HashMap can contain another key “type” that is assigned to the type of the object. If the type is supplied, then the structure tree will use the image cache to get the type icon to display with the structure node.

If the type is not passed into the structure tree, then the image used on object type's tree menu will be used. If the command does not have an image specified, then the type icon will be assigned to the value set to the `emxFramework.smallIcon.defaultType` key in the `emxSystem.properties` file.

Building Table Pages

- [Overview of Table Pages](#)
- [Table Body](#)
- [How the Pagination Controls Work](#)
- [Table Pages can Be Specific to an Object or Not](#)
- [Building a Table Page](#)
- [Parameters for Table Objects](#)
- [Refreshing the Table After Adding, Editing, or Deleting Objects](#)
- [Defining Table Column Data](#)
- [Improving Performance of Table Columns](#)
- [Sample JPO for Getting Table Column Values](#)
- [Parameters and Settings for Inquiry Objects](#)
- [URL Parameters Accepted by emxTable.jsp](#)

Overview of Table Pages

Table pages contain three main sections: the header, body, and footer. The primary component of a table page is the table, which is a list of business objects or business objects plus relationships. Table pages also contain other elements, such as a heading, actions menu, and pagination controls. For example, the graphic below shows a table page for a list of tasks. Each row represents one business object (or relationship) and each column represents a particular attribute or other characteristic about each business object/relationship.

The screenshot illustrates a table page for tasks, divided into three main sections: Header, Body, and Footer. The Header section includes a title 'Tasks' with a 'Label' field, a 'Filter List' dropdown set to 'Active', and an 'Actions Menu' with icons for adding, deleting, and refreshing. The Body section contains a table with columns: Name, Action, Task Instruction, Due Date, Route, and Workspace. The first row shows a task named 'Check figures' with an 'Investigate' action, a due date of 'May 31, 2002', and a workspace of '200'. The second row shows a task named 'Comm...' with a due date of 'May 14, 2002' and a workspace of '100'. The Footer section includes 'Pagination Controls' showing 'Page 1 of 1' and a 'Page Toolbar' with icons for navigation and printing.

Name	Action	Task Instruction	Due Date	Route	WorkSpace
Check figures	Investigate	Please check the figures on page 34.	May 31, 2002	Spec Review	200
Comm...		Please add your...	May 14, 2002	Route 100	Workspace 100

Table Body

The table body is an html FORM with the name “emxTableForm. This FORM holds the list of objects to be displayed. The table can be configured to display check boxes or radio buttons. The name of the check box or radio button control is “emxTableRowId.

The checkbox/radio buttons are assigned with the values of ObjectId and RelId, if available. If relId is available, the values are separated by “I”:

```
emxTableRowId = <objectId>
```

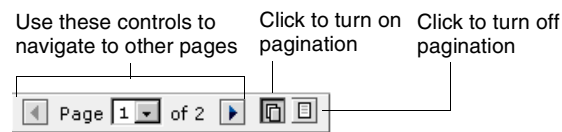
or

```
emxTableRowId = <relId>|<objectId>
```

How the Pagination Controls Work

Paginating a table means the table shows a certain number of rows per page. If the number of rows in the table exceeds that number, the user must navigate to the next page to see the additional rows. The number of rows per page is one of the configurable parameters for a table page. The parameter is called pagination and can be passed to `emxTable.jsp`. The pagination parameter can also be used to turn off pagination.

If the pagination parameter is not passed to `emxTable.jsp`, the system uses the pagination number specified for the `emxFramework.PaginationRange` property in `emxSystem.properties`. The default for this property is for a new table page to be created for every 10 rows. This number can be changed using the **Pagination** option in the **Preferences** page.



Users can turn on and off pagination using the **Paginate** and **Don't Paginate** buttons on the right side of the **Pagination** control panel. When pagination is turned off, all rows are listed on the first table page and the user must scroll to see the rows. The navigation controls are removed because there is only one page.

When a table page is paginated and configured so users can select one or multiple rows, the system remembers users' selections across pages. For example, suppose a table page lists parts for a company and there are 18 parts, 10 on the first page and 8 on the second. The user can check a part on the first page and a part on the second page and then click on a toolbar item to process both parts, such as a **Delete** or **report** action.

Remembering selections across pages can be turned off globally using the `emxNavigator.UITable.Pagination.RememberSelection` `emxSystem.properties` key and turned on and off per table page using the `rememberSelection` parameter for `emxTable.jsp`.

When configuring a toolbar item to process selected rows on a table page, you can tell the user how many rows are selected using the macro `${TABLE_SELECTED_COUNT}`. For example, in a toolbar item command object, you can use the macro in the **Confirm Message** setting to display a message such as "You are about to delete `${TABLE_SELECTED_COUNT}` parts. Click OK to continue..."

Table Pages can Be Specific to an Object or Not

A table page is typically accessed from a feature listed in a **My Desk** submenu or from a tree category. How a table is accessed affects some of the behavior for the page. For example, users could access a **Routes** table page by clicking **Routes>My Routes** from the **My Desk** menu.

Routes

Actions ▾   

		Name ▾	Description	Route Base Purpose	Status	Scope	Due Date	Owner	
		 Route-5000	Product development route	Standard	Started	All	2/15/2003	Smith	
		 Route-3217	Camshaft Diagram	Standard	Started	All	11/30/2002	Smith	
		 Route-3216	Circuit Board Change	Standard	Started	All	5/10/2002	Smith	
		 Route-3215	Jet Engine Turbine Blade	Approval	Finished	All	2/10/2002	Smith	

Page 1 of 1  

Users could also access a page with the Routes table by clicking a Route category within a tree. The table would be the same as the one used in the above graphic but the page might need a different heading or different actions. For example, in the graphic below, the Route category has been chosen from within a tree for a specific Part. The page that displays lists only the Routes for the Part whose tree we are working in.

PT-106000

- [Approvals](#)
- [History](#)
- [Lifecycle](#)
- [Parts](#)
- [Reference Documents](#)
- [Route](#)**

Routes

Actions ▾   

		Name ▾	Description	Route Base Purpose	Status	Scope	Due Date
		 Route-5000	Product development route	Standard	Started	All	2/15/2003
		 Route-3217	Camshaft Diagram	Standard	Started	All	11/30/2002

Page 1 of 1  

No matter how a table page is accessed, clicking the name of an object (or relationship) in the table opens the tree and the Properties page for the object. For example, clicking an package name from the My Packages table page brings up the tree for the package, as shown below.

My Packages All

Actions   

	Name	State	Owner	Response Status	RFQ	Workspace Folder(s)
☐	 Pkg100	Started		0/0	0	Folder1 Subfolder1A Subfolder1B

Page 1 of 1    

 **Pkg100**

- Attachment
- PackageRFQ
- History

Pkg100 - : Properties

Actions   

Name	Pkg100
Workspace Folder	Folder1 Subfolder1A Subfolder1B
Comments	Please see attached disclaimer.
Description	Package for assembly 2922822
Response Status	0/0

Similarly, clicking the name of an object from a table page that was called from a tree also opens the tree and Properties page for the object. But in this case, the tree for the object is inserted into the existing tree, as shown below with the line item tree that appears within the RFQ tree.

 **RFQ 300 1**

- Attachment
- LifeCycle
- Line Item Attribute Group
- Line Item UDA
- Line Item
- Previous Round
- RFQ Quotation
- Supplier
- Version
- History
- ECRs
- Supplier Header Attributes

RFQ 300 1: Line Items

Actions   

	Name	Description	Required Quantity	Required U of M	Receiving Plant	Required Availability	Response Count	Splits
☐	 LI-100	Part 392228		LB (pound)		Q4	0/2	 
☐	 LI-100	Part 392228		LB (pound)	Ohio	Q4	0/2	30.0%  
☐	 LI-100	Part 392228		LB (pound)	Utah	Q4	0/2	70.0%  

Page 1 of 1    

 **RFQ 300 1**

- Attachment
- LifeCycle
- Line Item Attribute Group
- Line Item UDA
- Line Item
 -  **LI-100**
 - Attachment
 - Attribute Group
 - UDA
 - History
- Previous Round
- RFQ Quotation
- Supplier
- Version

LI-100 : Properties

Actions   

Name	LI-100
Description	Part 392228
Part	None
Required Quantity	1000
Required U of M	LB (pound)
Required Availability	Q4
Receiving Plant	

Methods for Filtering a Table

There are several ways to configure a table page to let users filter table data.

The screenshot displays a web interface for managing RFQs. At the top, there's a header with 'RFQs' and a 'Filter drop-down' menu set to 'All'. Below this is a toolbar with an 'Actions' dropdown and several icons. The main table has columns: Name, Round, State, Owner, Quote Requested By Date, Response Status, Package, and Workspace Folder(s). Above the table, there's a 'Filter Table Data tool' section with a 'State' dropdown set to 'Create' and a 'Filter...' button. To the right of this is a 'Custom Filter' button. The table contains five rows of data, each with a checkbox, a document icon, an RFQ number (e.g., RFQ-1000), a round number, a state (e.g., Started), an owner name, a date, a response status, a package name, and a workspace folder link. At the bottom right, there's a pagination control showing 'Page 1 of 10'.

	Name	Round	State	Owner	Quote Requested By Date	Response Status	Package	Workspace Folder(s)
☐	RFQ-1000	1	Started	John Smith	Feb 28, 2003 9:00:00 AM	0/4	Package-2244	Aurora Dashboard
☐	RFQ-2000	2	Sent	Rebecca George	Feb 28, 2003 9:00:00 AM	1/3		Taurus Suspension
☐	RFQ-3000	4	Response Completed	Alice Tenon	Feb 28, 2003 9:00:00 AM	0/2		Explorer Wheel
☐	RFQ-4000	2	Started	Albert Swanson	Feb 28, 2003 9:00:00 AM	2/2		
☐	RFQ-5000	1	Sent	Thomas Wataki	Feb 28, 2003 9:00:00 AM	0/2		Jeep Rollbar

- **Filter drop-down**—Use the filter drop-down to predefine several lists of objects so users can select the list they want. The filter drop-down is best used for filtering that is not based on the column data. For example, the filter list on the RFQs page lets users select a list of all RFQs they own or a list of all RFQs they receive in a route. You configure a table page to show a filter list by passing inquiry object names or a JPO.
- **Filter tool**—Use the Filter tool to define the columns whose data users can filter the table by. Use the `Auto Filter=true` setting for a table column to let users filter by the column. For example, if you want to let users filter the table by state, you could set `Auto Filter=true` for the state column. Users can filter the table so it only shows objects in specific states. For instructions on using the tool, click help from the Auto Filter Selection page.
- **Custom filter**—Use a custom filter to place filter controls between the table page header and body frames. Implement the custom filter using a custom JSP and by passing the `FilterFramePage` and `FilterFrameSize` parameters to `emxTable.jsp`. For details on implementing the custom filter, see [Implementing a Custom Filter](#).

Building a Table Page

Mechanisms that Define a Table Page

The configurable JSP `emxTable.jsp` creates table pages. This JSP accepts many parameters that define the content and behavior of the page. These parameters are passed to the JSP through the href URL.

The main components of the table page—the table and its columns, the query for getting the list of business objects for the table, and the menus—are defined as admin objects. The names of these objects are passed to the `emxTable.jsp` through its URL parameters.

There should be a table admin object for each unique set of table columns. For example, suppose some ECR pages need to show the name, revision, and state for ECRs but other

pages need to show the name, revision, owner, and description. You would need a table admin object for each of these tables, maybe called ENCECRBasic and ENCECROwner.

The rows in a table are defined using inquiry or program admin objects. By populating a table page with different inquiries/programs, you can get different data with the same table. For example, an ECRs table can be used on one page to show a list of ECRs owned by the user and the same table can be used on another page to show a list of ECRs for a particular part.

How Table Pages are Linked to an Application

Users typically access table pages by selecting a link from a My Desk submenu or tree category. The component that calls the table page must specify the configurable JSP for table pages, `emxTable.jsp`, in the href parameter. The href should include any parameters needed to configure the page, as described in [URL Parameters Accepted by `emxTable.jsp`](#).

For example, the following URLs create a table page:

```
${COMMON_DIR}/
emxTable.jsp?inquiry=BuyerDesk,BuyerDeskAssigned&inquiryLabel=emxQuoteCentral.Common.All,emxQuoteCentral.Common.Assigned&table=BuyerDesk&selection=multiple&sortColumnName=Name&sortDirection=ascending&header=BuyerDesk&toolbar=SCSBuyerDesktoolbar
```

```
${COMMON_DIR}/
emxTable.jsp?program=emxBuyerDesk:getBuyerDesk,emxBuyerDesk:getAssignedBuyerDesk&programLabel=emxQuoteCentral.Common.All,emxQuoteCentral.Common.Assigned&table=RTSSummary&selection=multiple&header=RFQs&toolbar=BOM&sortColumnName=Name&sortDirection=ascending
```

Steps for Building a Table Page and Linking it to an Application

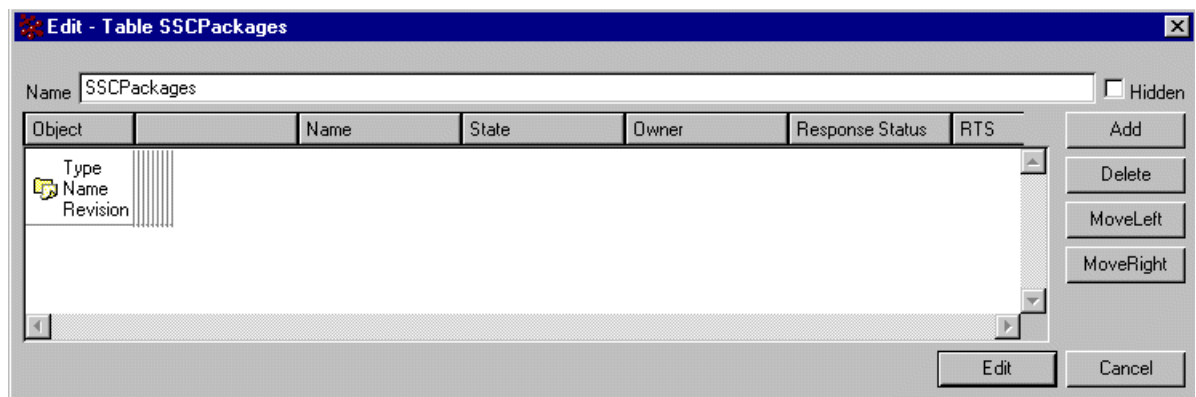
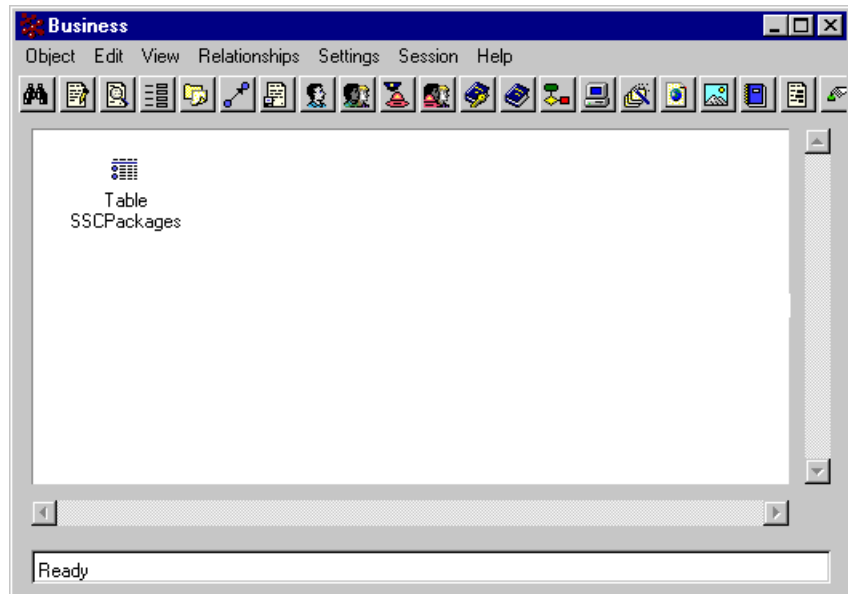
The following procedure lists the main steps needed for creating a table page and connecting it to an application. The graphics illustrate the components needed to create a table page for Packages called from a link on the My Desk submenu for Sourcing Central.

	Name	State	Owner	Response Status	RTS	Workspace
☐	Pkg100	Started	sdbuyer	0/0	0	

To create a table page and add it to an application

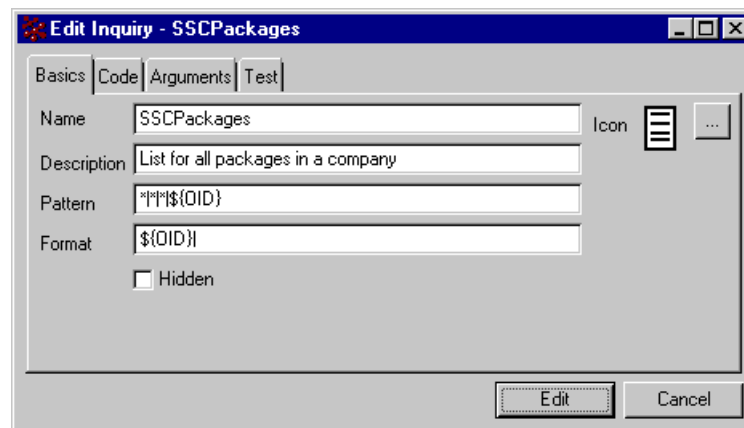
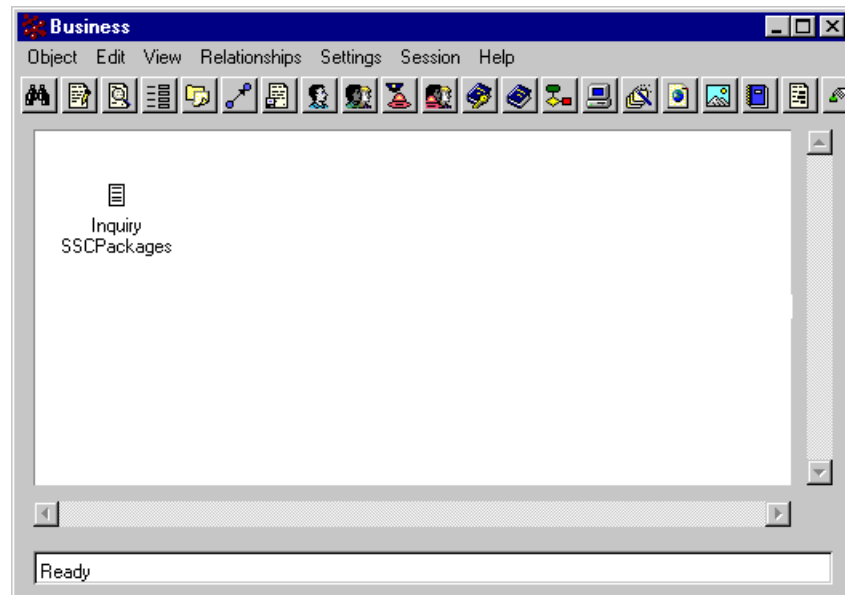
1. Create a table object and define the columns for it.

When you define columns, you define the data that should be in the column, the column heading, the order in which the columns should appear, and who can access each column. For a description of how to fill in the parameters and settings for tables and columns, see [Parameters for Table Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).



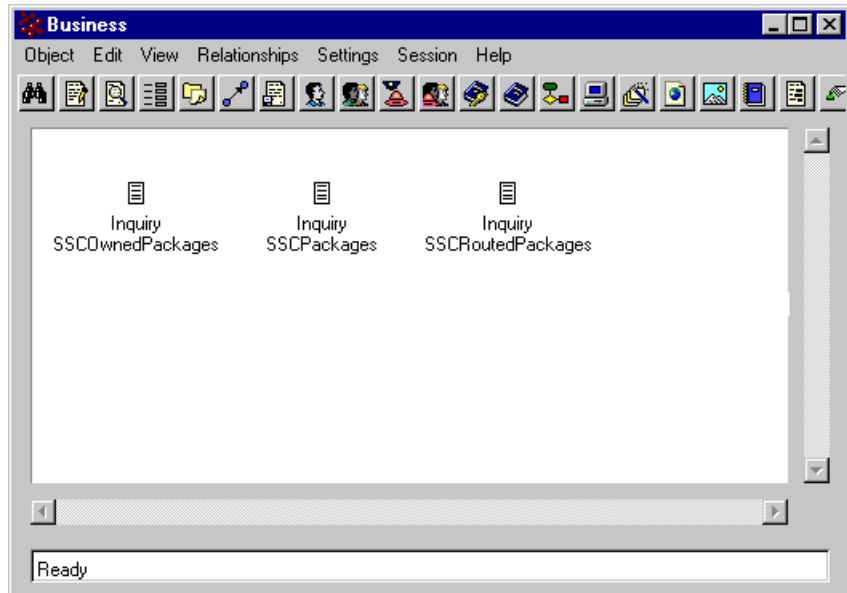
2. Create an inquiry object or program object to define how the system should get the list of business objects (and relationships, if appropriate) when the table page is first loaded.

For example, the Packages page should display all packages for a person's company. For a description of how to fill in parameters for inquiry objects, see [Parameters and Settings for Inquiry Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).



3. Create inquiry or program objects for each filter list option that is different from the inquiry/program needed when the page is loaded.

For example, the filter list for the Package page should contain options for showing only packages that are owned by the user and only packages routed to the user.



4. Make sure the menu and command objects are created for the action menus needed for the table page.

For instructions, see [Building Toolbars](#).

5. In the href parameter for the menu or command object that should call the table page (or in the JSP if a JSP is calling the page), enter `emxTable.jsp` and specify the parameters as needed to display the page. Since `emxTable.jsp` is in the `ematrix/common` directory, the first part of the URL should usually be: `${COMMON_DIR}/emxTable.jsp`. Here are two example href values. For a description of the parameters, see [URL Parameters Accepted by emxTable.jsp](#).

A sample URL for calling `emxTable.jsp` with the inquiry option:

```
${COMMON_DIR}/
emxTable.jsp?inquiry=BuyerDesk,BuyerDeskAssigned&inquiryLabel=emxQuoteCentral.Common.All,emxQuoteCentral.Common.Assigned&table=BuyerDesk&selection=multiple&sortColumnName=Name&sortDirection=ascending&header=BuyerDesk&toolbar=SCSBuyerDeskToolbar
```

A sample URL for calling `emxTable.jsp` with program option:

```
${COMMON_DIR}/
emxTable.jsp?program=emxBuyerDesk:getBuyerDesk,emxBuyerDesk:getAssignedBuyerDesk&programLabel=emxQuoteCentral.Common.All,emxQuoteCentral.Common.Assigned&table=RTSSummary&selection=multiple&header=RFQs&toolbar=BOMToolbar&sortColumnName=Name&sortDirection=ascending
```

A sample URL for calling `emxTable.jsp` with a custom filter:

```
${COMMON_DIR}/
emxTable.jsp?inquiry=PartList,PTParts,CMParts&inquiryLabel=emxEngineeringCentral.Common.All,PTParts,CMParts&table=DemoPartTable&header=emxEngineeringCentral.Common.Parts&toolbar=DemoPartToolbar&TipPage=emxBlank.jsp&headerRepeat=10&sortColumnName=Name&sortDirection=ascending&FilterFramePage=${COMMON_DIR}/emxTableFilterIncludeSample.jsp&FilterFrameSize=40&selection=multiple
```

6. If you are working with the Web-based user interface as you are making changes and want to see your changes in the user interface, click the **Reload Cache** tool in the global toolbar and click the browser **Refresh** button.

The cache is refreshed automatically when the component age expires. This setting is in `emxSystem.properties`.

Only persons assigned to the Administration Manager role have access to the Reload Cache tool.

Configuring an Editable Table

Tables can be configured to display in an editable mode. The edit link can be configured in two ways:

1. By passing in a URL parameter called `editLink=true`. This approach has no access control.
2. By configuring the toolbar command shown in the table with href assigned to `emxTableEdit.jsp`. This approach can be used when access control is needed for the edit command. It also needs to have the following associated settings:
 - `Submit=true`
 - `Popup Modal=true`
 - `Target Location=popup`

For example, this graphic shows a standard view-only table that lists SCOs. When the user clicks the Edit link...

	Name	Description	State	Business Unit	Region	Spec Office	Owner	
☐	SCO-5000	Description of the Specification Change Order	Draft	Tissues and Towel	North America	Department 1 - Spec Office 1	Haggarty, Dan	
☐	SCO-5003	Description of the Specification Change Order	Active	Tissues and Towel	North America	Department 1 - Spec Office 1	Haggarty, Dan	
☐	SCO-5004	Description of the Specification Change Order	Draft	Tissues and Towel	North America	Department 1 - Spec Office 1	Haggarty, Dan	
☐	SCO-5005	Description of the Specification Change Order	Draft	Tissues and Towel	North America	Department 1 - Spec Office 1	Haggarty, Dan	

...the same table opens in a popup window but some column values are editable, as shown below.

Dialog - Microsoft Internet Explorer

SCOs Filter: Owned

Column Value Apply to selected Apply to all

☐	Name	Description	State	Business Unit	Region	Sp
☐	SCO-5000	Description of the Specification	Draft	Tissues and Towel	North America	D
☐	SCO-5003	Description of the Specification	Active	Tissues and Towel	North America	D
☐	SCO-5004	Description of the Specification	Draft	Tissues and Towel	North America	D
☐	SCO-5005	Description of the Specification	Draft	Tissues and Towel	North America	D

Done Cancel

Like form pages displayed in edit mode, you can configure which columns can be edited. For example, in the above graphic, the Name and State columns cannot be edited but the Description, Business Unit, and Region can.

After users change the data as needed, they save the data by clicking Done. If the column type is not businessobject or relationship, an update program and update function must be written and specified to update the table data.

The editable table does not support standard table controls such as the Actions menu or toolbar, or filters. The currently-selected filter is displayed, as shown above, but users cannot change the selection. The editable table also does not support pagination controls. It shows whatever is shown in the view-only page.

Performing Mass Updates

Several or all rows on a page can be updated with a single operation.

1. Mass update is enabled by default. The default for the property setting `emxFramework.ShowMassUpdate` is true. This can be adjusted on the page level by passing the `massUpdate=false` URL parameter to `emxTableEdit.jsp`.
2. Select the rows to update using the check boxes at the beginning of each row. This step can be skipped if this change will be applied to all rows displayed.
3. Select the **Column** to update from the combo-box in the header. The **Value** field will automatically refresh to display one of the following:
 - a text box to enter a parameter
 - a combo-box to select a value from a range of values
 - a text box with a chooser

This will mirror the controls displayed for the individual fields shown in the form in the body frame.

4. Select values from the **Value** field.
5. Click **Apply to selected** to apply the changes to only the selected items

Or

Click **Apply to all** to apply to all records displayed.

Mechanisms that Define an Editable Table

The table and its columns are defined using the same table admin objects used to define view-only tables. Similar to form fields, when defining a table column that should be editable, you need to specify additional settings to tell the system how to handle edits to the column. For example, you need to specify the type of input control, any special validation for the data, and any update program for it.

The configurable JSP `emxTableEdit.jsp` creates editable table pages. This JSP is called automatically when the Edit link is clicked on a table page. The Edit link is included whenever `editLink=true` is passed to `emxTable.jsp`, as described below.

Adding an Editable Table to an Application

You add an editable version of a table page by adding the following parameter to the URL that calls the table:

`editLink=true`

If not passed, the parameter is assumed to be false. When `editLink=true` is included, the table page includes an Edit link in the toolbar or Actions menu. This link is automatically configured to call `emxTableEdit.jsp`. The system automatically passes the `suiteKey` and header parameters to the jsp, using the same values defined for the `emxTable.jsp`. This means the title for the editable version of the table is the same as the title for the view-only version.

In addition to passing this parameter, follow these steps to add an editable table:

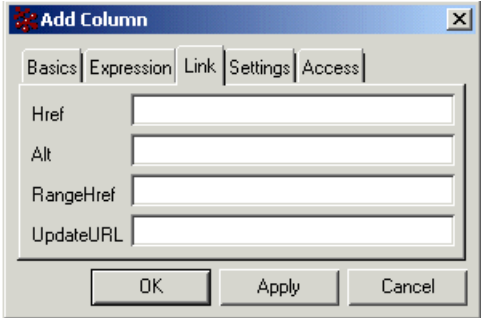
- Make sure all columns that should be editable include the `Editable=true` setting and other settings that define the type of input control, validation, and update programs needed to edit the data. For a list of parameters and settings, see [URL Parameters Accepted by `emxTable.jsp`](#). For examples of different types of editable columns, see [Defining Editable Table Columns](#).

Write any programs needed to get data or update data and create program objects to represent them.

Parameters for Table Objects

This table describes how to fill in the parameters for table objects. Except for the table name, a table object is just a set of column definitions so these parameters apply to columns within a table. For specific instructions on how to create table objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Access (user MQL command)	The persons, roles, and groups who can access the column. When you assign a role or group, all child roles/groups also receive access. To make the column available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access. Also see Controlling User Access to User Interface Components .	Names of group, role, person administrative objects. or All (default)
Alt	Applicable for table columns only when the column type is set to Icon.	Name of what the icon represents.
Applies To	The item to apply the select expression to: Business object or relationship. Applying an expression to a relationship is used to get a table of objects related to a known object. For example, getting all EBOMs for a part.	Business Object Relationship
Expression	The select expression to be used to get the column data. This expression is applied to either the relationship or business object, as specified in the Applies To options. Note that the Expression and Applies to options in Business are equivalent to the businessobject and relationship MQL commands. For more information on specifying column data using an expression, see Column Values as Select Expressions .	For business objects: - type - name \$<attribute[attribute_Originator].value> For relationships: \$<attribute[attribute_FindNumber].value> \$<attribute[attribute_Qty].value>
Heading (label MQL command)	The text that should appear on the column header. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components . The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	emxEngineeringCentral.common.N ame emxTeam.Common.ProjectName Description
href	The URL that gets executed when the column data is clicked. When a user clicks the hyperlinked column data, the system passes the objectId parameter as part of the URL. By default, the value for the objectId parameter is the ID of the business object for the current row. Using the Alternate OID expression setting, you can configure the field to pass the ID for a different business object. The value for the href parameter should be a JSP and any associated parameters. You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros .	\${COMMON_DIR}/emxTree.jsp \${SUITE_DIR}/ emxpartEditPartDialog.jsp

Parameter	Description	Accepted Values/Examples
Name	The name of the column used as identifier for the column within the table object.	Name Revision Originator
Name (of table)	Name of the table admin object. For naming conventions, see Naming Conventions for UI Administrative Objects .	ENCBOMList
RangeHref (range MQL command)	When configuring a table column in Business Modeler, the RangeHREF parameter is set on the Link tab.  Use to configure a textbox that has a Browse (...) button that calls a chooser or custom window from which users can select a value to populate the textbox. This Owner textbox is an example with a range helper.  In the RangeHref parameter, specify the href URL to display the window. For example, the href might call a custom selection page. Range helpers are used for table pages only in Edit mode. The only control that can be configured with a range helper is textbox. To specify the control type, set Input Type=textbox. To see an example of a field configured with a RangeHref, see Field with Popup Range Helper. Also see Implementing Range Helpers for Choosers or Custom Pages.	You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros. For example: <pre> \${COMMON_DIR}/ emxSelectVault.jsp \${SUITE_DIR}/ emxSelectUser.jsp emxTypeChooser.jsp?type peList=type_Part,type _Document ../common/ emxTypeChooser.jsp?&S electType=multiselect &SelectAbstractTypes= true&InclusionList=eS erviceEngineeringCent ral.Types&observeHidd en=true&ShowIcons=tru e </pre>
Settings	Additional settings that define the behavior and appearance of the column.	Name/value pairs, as defined in the table below.
sorttype	Defines how the column values should be sorted. <i>This parameter currently is not available in Business Modeler and can only be set and modified using the sorttype command in MQL. For example:</i> <pre> add table TestTable1 system column name Type label emxEngineeringCentral.Part.Type user all businessobject type setting "Registered Suite" EngineeringCentral sorttype numeric ; </pre>	numeric—Sorts the column values numerically. alpha (Default)—Sorts the column values alphabetically. other—Sorts the column values using a custom algorithm defined as a JPO and specified in the Sort Program setting for the column, or will use settings defined in Sort Type.

Settings for Table Objects

This table lists and describes the settings for table objects. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
Access Expression Access Function Access Mask Access Program	All these settings can be used to control access to table columns just as they can be used to control access to other UI components. For details, see Controlling User Access to User Interface Components .	—
Admin Type	Use to translate columns whose values are admin types. Based on the admin type, the value is translated and presented. For attributes, it is necessary to provide the symbolic name of the attribute, which starts with “attribute_” and is followed by the attribute name with no spaces.	Type State Role Relationship attribute_UnitOfMeasure
Allow Manual Edit	When true, users can manually edit the table column. Applicable only when the range parameter is set to a URL or when the setting format is assigned to date. It is ignored in all other cases. When this setting is true, the Admin Type setting is ignored.	true false (default)
Alternate OID expression	By default, when a column’s data is configured to show as a hyperlink using the href parameter, the system passes the ID of the object for that row in the objectId parameter. Using this setting, you can configure the hyperlink so a different objectId is passed. The system passes the ID for the object returned from the expression defined in this setting.	<code><to[relationship_NewPartPartRevision].from.id></code> <code><to[relationship_EBOM].from.id></code> For more information on configuring a column using an alternate object ID’s expression, see Column Values Using Alternate OID in href and Select Expression and Column Values Using Alternate OID and Type Icon in href with Select Expression .
Alternate Policy expression	This setting is required to display the state of any connected objects and show the value translated. This setting is applicable only when the Admin Type setting is set to State.	Any select expression that evaluates to a policy of a connected object.
Alternate Type expression	When the Show Alternate Icon setting is true, this expression is used to obtain the object type. Based on the obtained type, the corresponding icon is displayed. For more information on configuring a column using an alternate type expression, see Column Values Using Alternate OID and Type Icon in href with Select Expression .	<code><to[relationship_NewPartPartRevision].from.type></code> <code><to[relationship_EBOM].from.type></code>

Setting	Description	Accepted Values/Examples
Auto Filter	Determines whether users can filter the table rows based on data in the column. If at least one column in the table has Auto Filter set to true, the Filter  tool displays in the page toolbar. This setting cannot be applied to any column that has multiple values for a single data column cell. For more information on table filtering, see Methods for Filtering a Table. Configuring many columns for auto filtering or applying auto filter to inappropriate columns can potentially have high performance impact when viewing the filtered list. For optimum performance, define a limited number of columns with Auto Filter. Also choose columns that have a limited set of possible values on which the user may want to filter the table data. For example, state and owner columns are good candidates. Columns such as description and name are inappropriate because of the unlimited number of values that will result.	true—The Filter tool displays in the page toolbar. When clicked, the column is available on the Auto Filter Selection page. false (default)—The column is not available in the Auto Filter Selection page.
Column Icon	Use to display an icon for the column's data instead of other data. Required when the setting "Column Type" is assigned to "icon". Also see Column Values as Icons.	Name of an image file such as: images/NewWindow.gif images/EditItem.gif

Setting	Description	Accepted Values/Examples
Column Type	The setting “Column Type” is used when no expression is defined for the column data. For recommendations on improving the performance of table columns whose data is generated with a JPO program (Column Type set to program or programHTMLOutput), see Improving Performance of Table Columns. For a sample JPO program to get column data, see Sample JPO for Getting Table Column Values.	The column type can be one of the following: • program—The values for this column are obtained from a program (JPO). With this setting, the program and function name are required as settings. For more information on configuring a column using a program, see Column Values as Program Output. • programHTMLOutput—Same as the “program” setting above, except the column value output is in HTML format. Column values are placed in the table cell between <td> and </td> tags. This setting ignores other column settings such as Show Type Icon, href, format, and Alternate OID expression. • icon—Used when the column values are shown as an icon. The setting “Column Icon” must be defined with the icon to be displayed. For more information on configuring a column using a program, see Column Values as Icons. • checkbox—Used when the column values are check boxes shown grayed or not-grayed based on the access to the object in that row. This access can be based on business logic and defined in a JPO or it can be role-based and defined by assigning roles to the column. If the check boxes have no access restrictions, this setting is not required and you can create check boxes by passing in the parameter selection=multiple to emxTable.jsp. Also see Column Values as Check Boxes. • separator—Used to define a column of white space between standard data columns. A separator is especially useful to separate two groups of columns.
Currency Converter	Specifies that the currency shown in the column needs to be converted when the Currency Converter is run. When this setting is true for any column in a table, the table page automatically includes the Currency Converter tool. If this setting is not included, the value is assumed to be false and the target page does not display the Currency Converter tool.	–true –false (default)

Setting	Description	Accepted Values/Examples
Currency Expression	The name of the currency to convert from. Required for columns whose values are monetary and that can be converted from one currency to another using the Conversion tool and defined exchange rates. The value should be a select clause that returns the value for the Currency attribute for a specific object or relationship, or the actual value. The currency to be converted from should be the currency in which the user entered the data in (the As Entered currency).	For example, to convert currency data for an RFQ Quotation, the following select clause returns the supplier's currency format. <pre>to[Supplier Line Item].attribute[Currency]</pre>
Display Format	Specifies the number of the date format for any column where the value of the <i>format</i> setting is "date."	3 - SHORT (12/12/52) 2 - MEDIUM (Dec 12, 1952) 1 - LONG (December 12, 1952) 0 - FULL (Tuesday, December 12 1952 AD) Default is set in emxSystem.properties: emxFramework.DateTime.DisplayFormat=MEDIUM. emxSystem.properties uses words, but the Display Format setting uses numbers.
Display Time	Controls whether the time is displayed along with the date for columns whose format is set to date.	–true –false Default is set in emxSystem.properties for the property emxFramework.DateTime.DisplayTime = false
Editable	Use to indicate whether the column is displayed as editable or read only. Only applies for Edit mode. View mode ignores the setting.	true (default)—Users can edit the column data when shown on the Edit mode. false—Users cannot edit the column data when shown in the Edit mode. The column looks just like it does in View mode except it is never hyperlinked.
Effective Date Expression	Used for columns whose values are monetary and that can be converted from one currency to another using defined exchange rates. The value should be a select clause that provides the value for the Effectivity Date attribute on a specific object or relationship. The system uses this date to get the currency conversion whose rate period falls within this date. If this setting is not added, the current date is used.	For example, for currency data for an RFQ Quotation, the following select clause returns the effectivity date. <pre>to[Supplier Line Item].attribute[Effectivity Date]</pre>
Export	Specifies whether column data is exported or not. Use this to change the export value on a column-by-column basis. By default, all column types except programHTMLOutput are exported. If you want to additionally include programHTMLOutput, or exclude other column types from the export, change this setting. See Column Values as Program Output.	–true –false

Setting	Description	Accepted Values/Examples
format	Specifies the format to display the column data. If at least one column has the format set to currency or UOM, the Conversion tool displays in the table page's page toolbar. When a user clicks the tool, the system opens a new window and displays all column data defined with format=currency and UOM to the currency and unit of measure selected in preferences. For information on the currency and unit of measure preferences, see Currency Conversion and Unit of Measure Conversion .	date—Displays the column values as a date. Uses the tag lib “emxUtil:lzDate” to format the display. currency—Displays the column values as currency. UOM—Displays the column values as Unit of Measure and enables the Unit of Measure conversion interface. email—Displays the column values as an email address. When a user clicks the email address, the email editor configured in the client is presented.
function	The name of the method to call within the JPO program specified in the program setting. This method within the JPO is used to get the column values if the setting “Column Type” is set to “program” or “programHTMLOutput” or “checkbox”. See Column Values as Program Output .	getAssignedBuyerDesk getPackageAccess getParentPart getCurrentState
Group Header	Defines header text to display over several consecutive columns. For example, if you want a group header over three consecutive columns, add this setting to each column and assign the same value for each. You may want to separate grouped columns using a separator column, which is just a column of white space. Add a separator using Column Type=Separator. To see an example, see Grouped Columns and Column Separator .	Static text or string resource id.
Help Marker	Specifies the name of the help marker to call for context-sensitive help. In the href URL called when the column data is clicked, the system passes a parameter called HelpMarker and includes the marker text specified for this setting.	The naming convention for help markers “emxhelp” followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
Input Type	Only used for tables in Edit mode. Specifies the type of HTML control to display for user input. You can also designate the size of the input boxes to allow for appropriate spacing. This is important for short numeric entry fields.	textbox—This is the default. The column has a single-line box for typing text. textarea—The column has a multi-line box for typing text. combobox—The column has a drop-down list of options and users can only select one value. Use combo boxes for attributes that have defined ranges and for attributes whose ranges are determined with a Range Helper URL. To see an example of a field with a combo box, see Column Values Editable from Combo Box, Values from JPO .
Nowrap	If the setting is true for any column, then the text displayed in that column will not wrap at any time. The default is false.	–true –false

Setting	Description	Accepted Values/Examples
Popup Modal	If the setting Target Location is set to popup, the window can be configured as modal or non-modal.	true—the popup window is modal false—the popup window is non-modal If the setting is not specified, the window is non-modal.
Printer Friendly	Passes the PrinterFriendly parameter and the entered value to the JSP specified in the column's href URL. JSPs that use the PrinterFriendly parameter, such as emxTable.jsp and emxForm.jsp, show the Printer Friendly tool when the setting is True and hide it when False. If the setting is not included, emxTable.jsp and emxForm.jsp show the tool by default. Users can click the tool to get a version of the current page that can be printed with the browser's Print button. Note that you can also specify the PrinterFriendly parameter in the href URL for JSPs that use it.	–true –false
program	The name of the JPO program to use to get the column's data. This program is used to get the column values when the setting "Column Type" is set to "program" or "programHTMLOutput" or "checkbox". <i>Using a JPO to populate column data is recommended only when a select expression cannot be used to obtain the values.</i> For more information on configuring a column using a program, see Column Values as Program Output .	emxSCSBuyerDesk emxTMCPackages emxENCParentPart emxCommonTableUtil
Range Function	Use to specify the name of the method in the JPO specified in the Range Program setting. See the Range Program setting below for more information.	The name of a function in the Range Program JPO, such as: - getAssignedRange - getClassificationRange - getPartUOM
Range Program	Use to specify the name of a JPO that contains a method to get the column value ranges (choices). A range program is used only when the Input Type setting is combobox or popup. The corresponding method which runs to get the values of the column should not have any HTML code. To see an example of a field configured with a range program, see Column Values Editable from Combo Box, Values from JPO .	The name of a JPO, such as: - SCSBuyerDeskForm - TMCPackagesForm - ENCPartForm - AEFUtilForm

Setting	Description	Accepted Values/Examples
*Registered Suite	The application the column belongs to. The system looks for files related to the column in the registered directory for that application, which is specified in <code>emxSystem.properties</code>. Based on the application name, the system passes the following parameters in the href URL: • <code>suiteKey</code> • <code>emxSuiteDirectory</code> • <code>StringResourceFileId</code>	The value must be set without any spaces, for example, <code>EngineeringCentral</code> or <code>Framework</code>. The value must be set to the suite name as defined in the key <code>eServiceSuites.DisplayedSuites</code> within <code>emxSystem.properties</code>. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in <code>emxSystem.properties</code> is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the column data is clicked, the system passes a parameter called “suiteKey”. The value for the parameter is the property name from <code>emxSystem.properties</code> that maps to the setting’s value.
Show Alternate Icon	If the column value to be displayed with the icon is different from the current row’s object icon, set this setting to true. In order to get the right alternate icon, the Alternate Type expression setting must be defined with the expression to obtain the object type. For more information, see Column Values Using Alternate OID and Type Icon in href with Select Expression.	–true –false
Show Type Icon	If set to true, the column value is displayed along with the object type icon as defined in the <code>emxFramework.smallIcon</code> property in <code>emxSystem.properties</code>. The icon is displayed to the left of the column data. If no property is defined for the type’s icon, the system looks for a property defined for the parent type, then grandparent type and so on. If no property is defined for any type in the hierarchy, the system uses the default icon specified in the <code>emxFramework.smallIcon.defaultType</code> property.	–true –false (default)
Sortable	Controls whether users can sort the table based on data in the table column. If a column is sortable, users can click the column heading to sort the table based on that column.	–true (default) –false
Sort Program	Specifies the JPO program name that contains a custom sorting algorithm. This setting is used only when the column parameter <code>sorttype</code> is assigned to “other”. For guidelines and parameters that must be used for the custom sorting program and sample programs, see Custom Sorting Programs.	Name of a JPO added as a program in the database, such as: <code>FindNumberSort</code> <code>CustomSort</code>
Sort Type	Determines how the column is sorted. If the <code>sorttype</code> parameter is specified, this setting overrides the parameter.	date integer real

Setting	Description	Accepted Values/Examples
Target Location	Controls where the page specified in the href parameter appears or is targeted.	content—The page replaces the content frame. popup—Page appears in new window. The window modality can be set with the Popup Modal setting. _top—Page replaces the entire body of the browser window. listHidden—A hidden frame within the table frameset. Use this frame for Target Location for any processing in the context of a table. hiddenFrame—The frame called hiddenFrame is part of the top level Navigation window frameset and can be used to submit the Table frame and carry out background processing. This frame is not available for popup windows though so listHidden is a better frame to use. This value can be set to any valid frame name, which is available to the table body frame.
Tip Page	Specifies whether the target page should include the Tip Page tool. In the href URL called when the column data is clicked, the system passes a parameter called TipPage and includes the URL specified for this setting. If this setting is not included, the target page does not display the Tip Page tool.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the table.jsp: TipPage=../doc/customcentral/tippage.html
UOM Expression	The name of the Unit of Measure to convert from. Required for columns whose values are measurements that can be converted from English units to Metric units or vice versa. The value can be a select clause that returns the value for the Unit of Measure attribute for a specific object or relationship or the actual value. The unit of measure to convert form should be the unit of measure in which the user entered the data (the As Entered unit of measure).	For example, for measurement data for RFQ Quotations, the following select clause returns the supplier's unit of measure format. <pre>to[Supplier Line Item].attribute[Unit of Measure]</pre>
Update Function	Specifies a method name in the JPO given in the Update Program setting.	The name of a function in the Update Program JPO, such as: • setAssignedBuyerDesk • setPackage Access • setPartClassification

Setting	Description	Accepted Values/Examples
Update Program	Specifies a JPO that contains a method to set the column value when the column is displayed on an Edit mode table and when the user clicks Done. This program is used only when the Field Type setting is program or programHTMLOutput. <i>Using an update program is recommended only when the Field Type is not attribute or basic.</i> See Using JPO to Update Table Values for more information.	The name of a JPO, such as: - SCSBuyerDeskForm - TMCPackagesForm - ENCPartForm - AEFUtilForm
*Required Setting		

Refreshing the Table After Adding, Editing, or Deleting Objects

The refreshTablePage() JavaScript API method is available to refresh the table to update the content after adding a new item in the list, updating the selected item in a table, or deleting selected items in a table.

This function must be called with reference to the top window object. For example, to refresh the table page, call:

```
// JavaScript Code
top.refreshTablePage();
// JavaScript Code
```

The refreshTablePage() method can be used to update the “detailsDisplay” (frame which is the right side frame of “treeDisplay”) or the “content” frame.

Getting the Object or Relationship ID of a Selected Table Item

Once you've configured a table page to display objects and/or relationships in a table, you'll typically want to add links to the page so users can work with the objects in the table. Links that apply to objects or relationships in a table should be displayed as hyperlinked data in the table or as toolbar items at the bottom of the page. Toolbar items should be commands that are independent of the table data, such as a Create New or Show Report link.

The JSP or JPO that gets called when one of these links is clicked needs the object or relationship ID(s) to work on. This table summarizes how the dynamic UI components make the IDs available to the requested page depending on the kind of link that is used.

If the table page link is:	Clicking the link results in:	For details, see:
Hyperlinked data within the table	The objectId, relId or parentOID parameter automatically being put into the request. The availability of the parameter name depends on whether the column is displaying business object or relationship data. The objectId and relId are values for the selected row(s) and parentOID is the parent objectId, which was passed to the Table page.	Defining Table Column Data
A toolbar menu command that requires one or more selected table rows (selected using check boxes or a radio button in the left table column)	The emxTableRowId parameter is available to the requested URL as long as the Submit setting for the toolbar item is True. The emxTableRowId parameter contains: - for a table generated by a temp query Inquiry or JPO program returning object ID: one or more object IDs (one for each checked checkbox) - for table generated by an expand Inquiry or a JPO program returning object ID and rel ID: one or more object ID, relationship ID pairs (one for each checked check box or one for the selected radio button). The relationship ID and object ID values are separated by a pipe character.	For instructions on configuring the toolbar item, see Building Toolbars .

When programming custom JSPs, make sure you use built-in parameters instead of custom parameters. For more information, see [Use Built-In Parameters to Make Pages Reusable](#).

Defining Table Column Data

This section describes how to use the parameters and settings for a table admin object to define the data for columns. A table column can contain information about a business object or relationship, data from a program, check boxes, or icons.

Column Values as Select Expressions

One way to get column data is based on a select expression applied to a set of OIDs or RELIDs. For this option, the column definition must include a valid select expression for a business object or relationship.

Examples of select expressions:

- type
- name
- `$<attribute[attribute_Originator].value>`
- `$<attribute[Originator].value>`
- `attribute[Originator].value`
- `$<attribute[attribute_FindNumber].value>`

Here is an MQL script snippet that defines a table with column values as select expressions. The first two columns define expressions on business objects and the third has an expression on a relationship.

```
add table ENCParts system
column
    name Name
    businessobject name
column
    name Originator
    businessobject $<attribute[attribute_Originator].value>
column
    name FindNumber
    relationship $<attribute[attribute_FindNumber].value>
```

Column Values as Program Output

The column values can also be obtained based on specific business logic defined within a method in a Java Program Object (JPO) and not based on a select expression. Instead of a select expression, the column setting called “Column Type” is used and set to one of the following:

- Column Type = program: This first option is used when the program results contain only the column value.
- Column Type = programHTMLOutput: This option is used when the program results contain the complete HTML tag (including the column value) to be displayed in the table column.

The above options require a JPO for getting the column value results. The name of the JPO and the method to be used for this column is also configured as settings for the table column:

- program = JPO_NAME: The name of the JPO program object to be used for getting the column values.
- function = JPO_METHOD_NAME: The name of the method within the JPO program which will be implemented to return the results of column value.

For columns with the setting “columnType = program,” the corresponding method which runs to get the values of the column should not have any html code. If the method returns any html code then the same code will be written in the PrinterFriendly pages also and so you will see the links in the PrinterFriendly pages. In case of export, the html tags will be present in the exported file.

If you want to return the html code from the method, then use the setting “Column Type = programHTMLOutput” instead of “Column Type = program.”

In order to get the column values exported when the Column Type is “programHTMLOutput,” you must also set “Export= true” for the column.

To take care of not returning the html code or href links for PrinterFriendly pages and export files, the JPO method should have some conditional logic to decide when to return the values with or without html code. The parameter “reportFormat” available to the JPO must be used for checking the condition.

For regular web page display, the reportFormat will be null or empty string. The reportFormat parameter will contain a valid value only for the export mode or Printer Friendly mode.

Valid reportFormats are CSV, HTML, TXT and ExcelHTML.

The following is sample code:

```
public static Vector functionName (Context context,String args[]) throws Exception {
    Vector retVal = new Vector();
    String reportFormat = (String)paramMap.get("reportFormat");

    String name = "name to Displayed in the column";
    String outString = "html code";

    for (each object) {
        if(reportFormat != null && reportFormat.length() > 0)
        {
            outString="<a name=" + name + "href=../common/
emxTree.jsp?treeMenu=type_TechnicalSpecificationTemplate&mode=insert&jsTreeID="+jsTree
ID+"&objectId="+objectId+" ><img src=../common/images/iconSmallSpecification.gif
border=0>" + name + "</a>";
        } else {
            outString=name;
        }
    }

    // If anything special has to be done for specific report formats then users can
    check reportFormat with corresponding string and write the code like the following if
    needed

    if("CSV".equals(reportFormat)) {
        //special code for CSV format
    } else if("TXT".equals(reportFormat)) {
        //special code for TXT format
    } else if("HTML".equals(reportFormat)) {
        //special code for HTML format
    } else if("ExcelHTML".equals(reportFormat)) {
        //special code for ExcelHTML format
    }
```

```

    }

    retVal.add(outString);
}
return retVal;
}

```

For a sample program to get column values for Column Type set to program, see [Sample JPO for Getting Table Column Values](#). For recommendations on improving the performance of a programmatic table column, see [Improving Performance of Table Columns](#).

The JPO for the columns must follow these rules:

- The program must define a method with the exact name specified in the column setting “function”.
- The method takes an input argument as a HashMap containing details of the object list to be processed. The data structure of this input parameter (HashMap) is defined below:

Key Name	Data Type	Description
objectList	MapList	This MapList contains a list of HashMaps with the values of OIDs and RelIDs. Every HashMap contains the value for OID and RelID belonging to one table row. The size of the MapList passed in equals the list of rows to be presented in the table. The structure of each HashMap belonging to a table row is defined below: • Key “id” is assigned to the business object Id (OID) to be used in the current table row. • Key “id[connection]” is assigned to the relationship id (RelID) to be used for the current row.
paramList	HashMap	This HashMap contain the all the request parameter key / value pair which were available in the JSP page level. An additional parameter “languageStr” is available to the HashMap. This key is assigned to the value with current browser language setting (like en, fr, ja etc). This value may be used for the internationalization purpose within the JPO program.

- The method must return a vector of String or StringList, containing the column values to be displayed. The vector size must be equal to the size of the input MapList.

Here is a template for the JPO function.

```

public static Vector methodName(Context context, String[] args)
throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);

```

```

        MapList relBusObjPageList =
(MapList)programMap.get("objectList");
        HashMap paramMap = (HashMap)programMap.get("paramList");

        Vector columnValues = new Vector(relBusObjPageList.size());
        String bArr [] = new String[relBusObjPageList.size()];
        String rArr[] = new String[relBusObjPageList.size()];
        StringList bSel = new StringList();
        StringList rSel = new StringList();

// Get the required parameter values from "paramMap" - if
required
String languageStr = (String)paramMap.get("languageStr");

// Get the object elements - OIDs and RELIDs - if required
for (int i = 0; i < relBusObjPageList.size(); i++)
{
    // Get Business object Id
    bArr [i] =
(String)((HashMap)relBusObjPageList.get(i)).get("id");

// Get Relationship Id
    rArr [i] =
(String)((HashMap)relBusObjPageList.get(i)).get("id[connection]
");
}

// Add any Business object selects if required
bSel.add("current");
    bSel.add("policy");
    bSel.add("attribute[Originator].value");

// Add any Relationship selects if required
rSel.add("attribute[Find Number].value");

// Process the OIDs to get the results - if required
BusinessObjectWithSelectList bws1 =
context.selectBusinessObjectData(bArr, bSel);

// Process the RelIDs to get the results - if required
RelationshipWithSelectList rws1 =
context.selectRelationshipData(rArr, rSel);

// Code for processing the result data obtained - if required

//Build the Vector "columnValues" with the list of values to
be displayed in the column

    return columnValues;
}

```

Column Values as Check Boxes

There are a couple of ways to configure a column of check boxes:

- Pass in the parameter `selection=multiple` to `emxTable.jsp` to add a small column of check boxes on the left side of the table page. There is no access control for these check boxes so they are enabled for all users. For more information on the selection parameter, see [URL Parameters Accepted by emxTable.jsp](#).
This parameter takes precedence over any check box column added to the table admin object (using `Column Type=checkbox`). So if `selection=multiple` is passed to `emxTable.jsp`, any check box column added to the table is not used. To use a check box column added to the table object, pass `selection=none` to the `emxTable.jsp`.
- Add a column with the setting `Column Type=checkbox`. For this type of column, the column value is a check box that should be grayed or active based on specific business logic or based on the roles assigned access to the column.
If access should be based on business logic, and not role-based, the business logic to decide the access for every row is implemented in a JPO. Therefore, the “program” and “function” settings must also be defined for this option.
The JPO is written in the same way as described in [Column Values as Program Output](#). The only difference is that the method must return Vector containing the strings either “true” or “false”. If the value of Vector elements contains “true”, the column cell is displayed with an active check box and if “false”, the column cell is displayed with a grayed check box.

Column Values as Icons

The column value can also be an icon. The right-most column in most tables, which contains the popup window  icon and displays details for the current object, is an example of this kind of column. To configure a column so its data is an icon, the table column must be configured with “Column Type” setting as “icon”. The icon to be displayed is set in the column setting “Column Icon”.

Column Values Using Alternate OID in href and Select Expression

Another way to configure column data that is hyperlinked is to use a select expression and an alternate OID (one different from the OID for the row’s object). This alternate OID is passed as parameter “objectId” as part of the href when the column data is clicked.

In order to achieve this use case, use the setting Alternate OID expression and assign it a valid select expression. This expression must be a select clause for obtaining an OID connected/related to the current object in the table row.

The Workspace column in the following graphic is an example of this kind of column data. The href for the hyperlinked workspace name, for example Aurora Dashboard, is appended with the objectId parameter. The value for the parameter is obtained from the select expression assigned to the Alternate OID expression setting. If the Alternate OID expression setting is not used in a column, the current row’s objectId is appended to the href value. For example, the objectId parameter used in the Name column is the current row’s objectId.

RFQs										All
Actions    										
State		Create	Filter...							
☐		Name	Round	State	Owner	Quote Requested By Date	Response Status	Package	Workspace Folder(s)	
☐		 RFQ-1000	1	Started	John Smith	Feb 28, 2003 9:00:00 AM	0/4	Package-2244	Aurora Dashboard	
☐		 RFQ-2000	2	Sent	Rebecca George	Feb 28, 2003 9:00:00 AM	1/3		Taurus Suspension	
☐		 RFQ-3000	4	Response Completed	Alice Tenon	Feb 28, 2003 9:00:00 AM	0/2		Explorer Wheel	
☐		 RFQ-4000	2	Started	Albert Swanson	Feb 28, 2003 9:00:00 AM	2/2			
☐		 RFQ-5000	1	Sent	Thomas Wataki	Feb 28, 2003 9:00:00 AM	0/2		Jeep Rollbar	
 Page 1 of 10  										

Column Values Using Alternate OID and Type Icon in href with Select Expression

In addition to using a select expression to show the hyperlinked name of an alternate objectId (as described in the previous section), the column can include the icon for the alternate object's type. To include this icon, use the Alternate Type expression setting and assign it a valid select expression and assign the Show Alternate Icon setting to true. The select expression must be a select clause for obtaining a TYPE of the connected/related object. The type name returned by this select expression is used to obtain the icon name to be displayed.

The ECOs column in this graphic is an example of column data that includes a type icon. The icon appears to the left of the object name. The icon is obtained based on the Alternate Type expression and Show Alternate Icon settings. This column is also configured to use the Alternate OID expression setting for the hyperlinked data.

Type	Name ▾	Rev	State	ECOs	EBOM	Weight	Originated	Vault	
Part	 CM-200000-01	3	Release	 ECO-007020		0.0	Apr 19, 2002 11:10:16 AM	eService Sample	
Part	 CM-200000-01	2	Release	 ECO-007010		0.0	Apr 19, 2002 11:10:14 AM	eService Sample	
Part	 CM-200000-01	1	Release	 ECO-007000		0.0	Apr 19, 2002 11:10:04 AM	eService Sample	
Part	 CM-200001-01	1	Release			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	
Part	 CM-200002-01	1	Release			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	
Part	 CM-200003-01	1	Preliminary			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	
Part	 CM-200004-01	1	Preliminary			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	
Part	 CM-200005-01	1	Preliminary			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	
Part	 CM-200006-01	1	Preliminary			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	
Part	 CM-200007-01	1	Preliminary			0.0	Apr 19, 2002 11:10:06 AM	eService Sample	

Here is an example MQL code snippet for this use case:

```
//*****
column
  name ECOs
  label emxEngineeringCentral.Common.ECO
  user "all"
  businessobject
"$<to[relationship_NewPartPartRevision].from.name>"
  setting "Alternate OID expression"
"$<to[relationship_NewPartPartRevision].from.id>"
  setting "Alternate Type expression"
"$<to[relationship_NewPartPartRevision].from.type>"
  setting "Show Alternate Icon" true
  href "emxTree.jsp"
  setting "Registered Suite" "EngineeringCentral"
//*****
```

Grouped Columns and Column Separator

You can create a group of columns by adding a header over several consecutive columns and be separating the columns with a separator. To specify a group header, add the setting

Group Header for each column in the group. To add a separator, use the Column Type=separator setting. Here is an example of grouped columns with separators.

The screenshot shows a software interface with a table titled "Parts" and subtitle "Bill of Material Level 1". The table has columns for Name, Rev, State, State, and a group of columns for Related Items (EBOM, ECO, Estimated Cost, Weight, UOM). A callout points to the "Group Header" for the Related Items group. Another callout points to a "Separator" column, which is a column with a blue background and a white border, used to separate the columns in the group. The table contains several rows of data, including parts like CH-202100-01, CH-202110-01, CH-202110-01A, CH-202120-01, and CH-202120-01. The bottom of the interface shows a pagination bar with "Page 5 of 21" and buttons for "Next" and "Cancel".

Name	Rev	State	State	EBOM	ECO	Estimated Cost	Weight	UOM
CH-202100-01	1	Approved	Approved	CH-203050-01 CH-203040-01	ECO-007000	0.0	0.0	EA (each)
CH-202110-01	1	Approved	Approved		ECO-007000	0.0	0.0	EA (each)
CH-202110-01A	1	Approved	Approved		ECO-007000	0.0	0.0	EA (each)
CH-202120-01	1	Approved	Approved		ECO-007000	0.0	0.0	EA (each)
CH-202120-01	1	Approved	Approved	CH-203070-01 CH-203060-01	ECO-007000	0.0	0.0	EA (each)

Column Values as Special Information from Business Object

A table column can be configured to show any information obtained by the method used for getting the object list. The list generated by inquiry or JPO may contain additional information other than OID and RelID. This “special information” may not be available at a later point for displaying as column values. For example, when doing an expand on a part for relationship EBOM, the level information is available, but cannot easily be obtained from the resulting object list.

To make this special information available for column values, it can be obtained at the time of fetching the object list and stored in the object list map with a pre-defined key. To display these values in a column, configure a column as Column Type=program and use the Object List Map to obtain the values and return it to display as column values.

Defining Editable Table Columns

This section shows examples of how to configure table columns to be editable.

Column Values Editable in Text Box

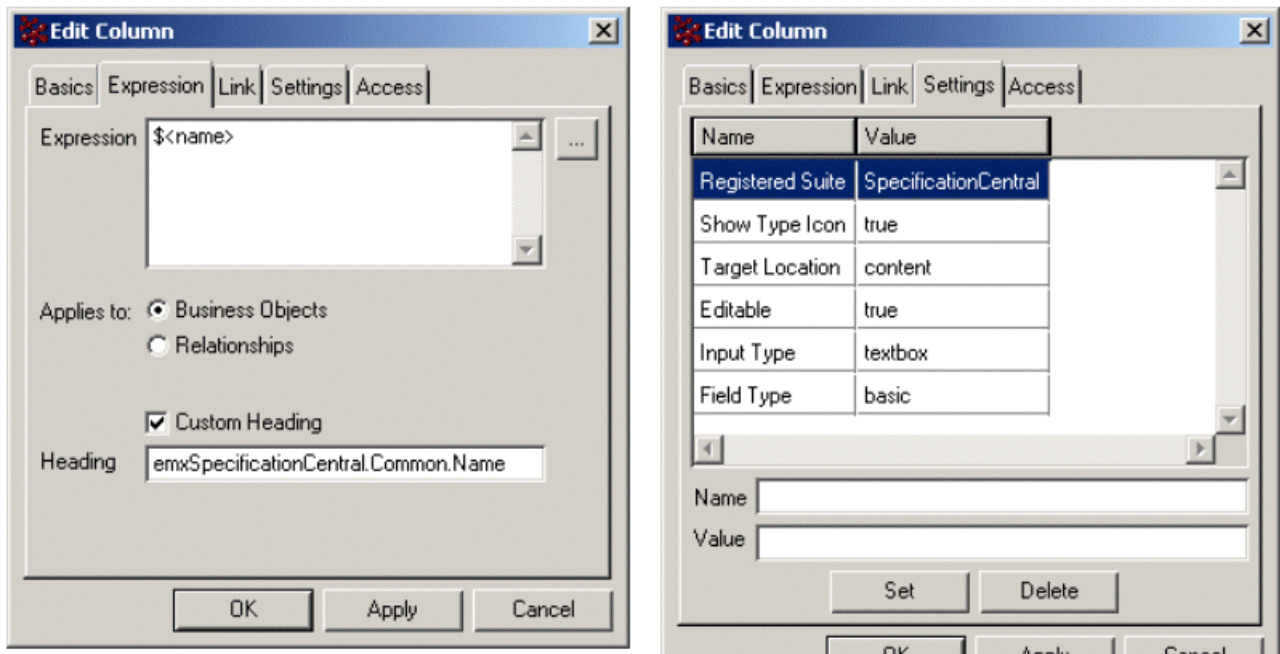
One way to let users edit column values is to let them type in new or changed information in a text box. For example, to get data for a Name column, you can use a select expression applied to the current business object and let users edit the name using a simple text box. This graphic shows examples of a Name column for a table displayed in Edit mode.

The screenshot shows a text box for editing a Name column value. The text box is labeled "Name" and contains the text "SCO-5000". Below the text box is a button labeled "SCO-5005".

To configure a column so the data is obtained from a select expression and displayed in an editable text box, use these parameters and settings

- Expression parameter—Enter the select expression that should be applied to a business object or relationship to get the column values.
- Applies to—Choose what the expression should be applied to: Business Object or Relationship.
- Field Type setting—When the expression is a business object basic or an attribute and the column is editable, you must include the Field Type setting. This lets the system correctly update the basic or attribute property.
- Editable setting—This setting must be true to let users edit the column data.

The below graphics show how the Name column shown previously is configured in Business Modeler.



Column Values Editable from Combo Box, Values from JPO

There are cases where a column can be edited but users shouldn't be able to just type in values. The values can only be changed to a specific set of values. If the set of possible values for the column is fairly small, they can be presented in a combo box and users can choose the value to change to. For example, the 3 columns shown below let users choose different values from combo boxes.

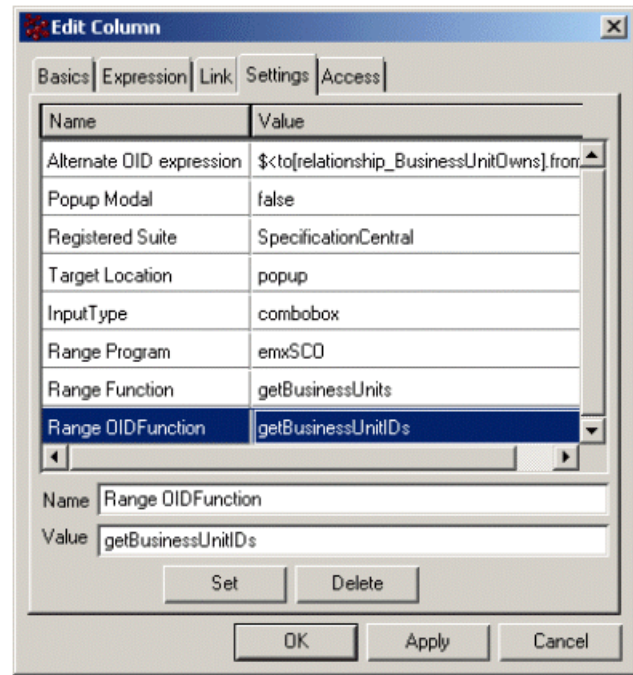
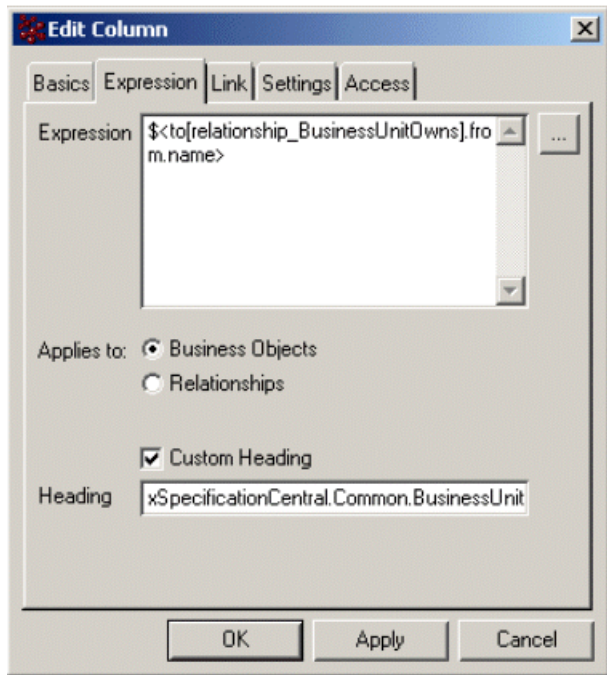
Business Unit	Region	Specification Office
BU 1	Region 1	Spec Office 1
BU 1	Region 1	Spec Office 1
BU 1	Region 1	Spec Office 1

One way to get the values to populate the combo boxes with is to from a method in a JPO program. The method is defined to obtain the values based on specific business logic.

To get column range values using a method in a JPO, use these parameters and settings

- Column Type setting—Set to `program` when program results contain only column values. Set to `programHTMLOutput` when program results contain complete HTML tag (including the column value) to be displayed in the column.
- Input Type setting—`combobox`
- Range Program setting—The JPO from which the choices are obtained.
- Range Function—The method in the Range Program JPO that returns the choices in the object type string list.
- Editable setting—This setting must be true to let users edit the column data.

These graphics show a column configured to display editable choices in a combo box, where the choices are retrieved from a JPO. The currently-selected value for the field, which is displayed on the view-only version of the table, is retrieved from an expression.



Column Values as Editable Dates

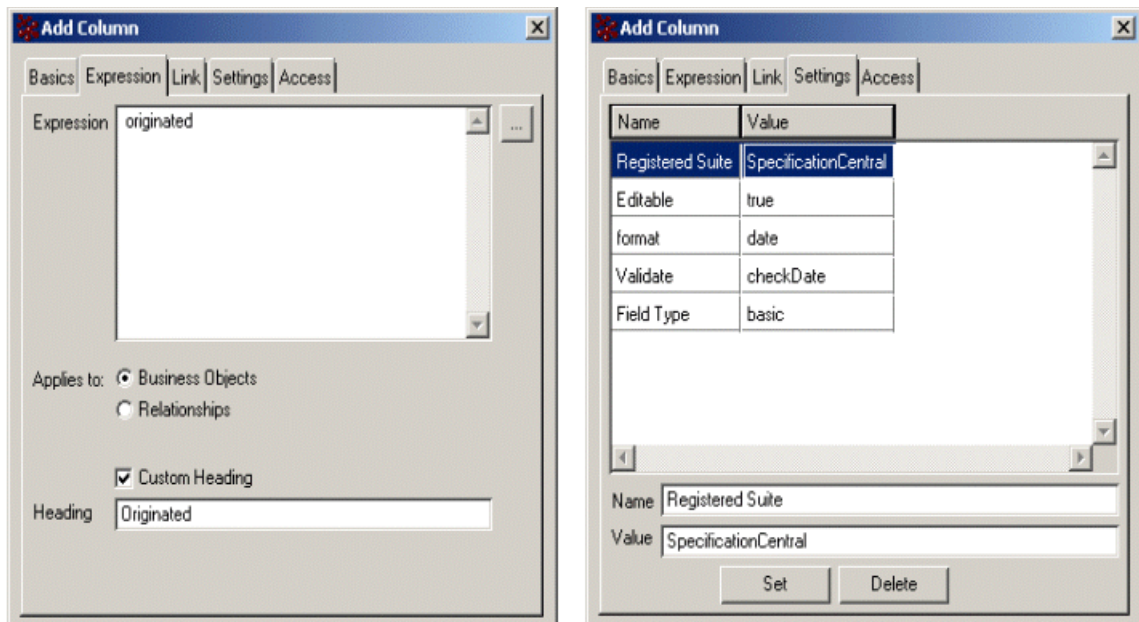
Column values can be configured to display dates and to let users change the date using a calendar chooser. The value must be a valid date string. The date value is displayed based on the system and browser locale setting using the “IzDate” taglib. For example:

Planned Build Date (MMM d, yyyy)	Actual Build Date
Apr 8, 2004	Apr 14, 2004
Apr 7, 2004	Apr 8, 2004

The format of the value displayed in the column should match the format listed in parenthesis following the date control.

- The first column, Planned Build Date, uses the following settings:
format = date
Editable = true
Allow Manual Edit = true
(Note: The column header shows the date format depending on the browser locale setting.)
- The second column, Actual Build Date, uses the following settings:
format = date
Editable = true

For example, the following graphics show a column set up to display the originated date for the current object. The column is also set to validate the data.



Using JPO to Update Table Values

To store the values of the related objects, JPOs can be used for saving the values. See [Writing a JPO for Updating Column Values](#).

The following are the two column settings that need to be saved through JPOs:

- Update Program—The JPO that contains the method that saves the values.
- Update Function—The name of method in the JPO that saves the data.

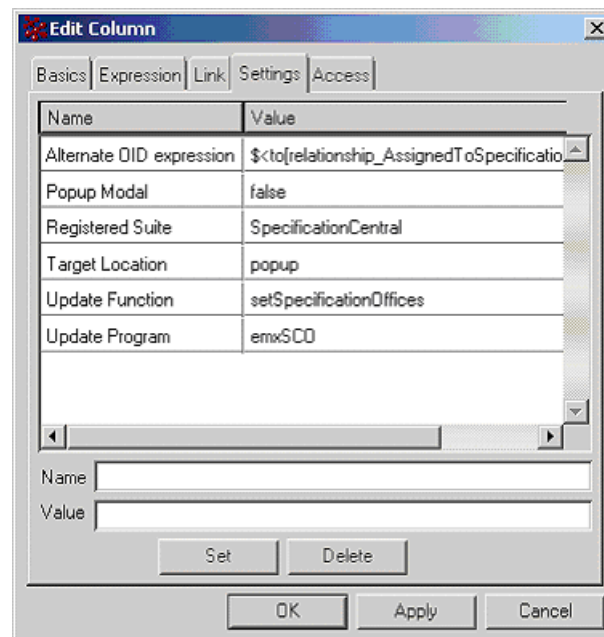


Table Column Calculations

Tables can be configured to provide calculations for columns and objects. Depending on the pagination setting, calculations are given per page, or for all items in the table. The calculation section header shows a count of the number of objects displayed on the page or in the table. Column values can be totaled and manipulated to give sum, average, maximum, minimum and median values. Each column can have one or more calculation types applied to it. The results are shown at the bottom of the table.

- Calculations can be performed only on numerical values.
- Null or empty rows are considered as a value of zero. This allows calculations to be performed on data sets that may be missing values for some rows.
- More than one calculation can be performed on the same column. For example, a column may have a sum and an average specified for it. This will result in two rows in the calculation section, one for each.
- The sum calculation should be placed as the first calculation in the section at the bottom of the page. This prevents confusion as to what is being totaled.
- Page calculations are optional, since they may not be useful in some circumstances. You may want to perform the overall calculations only and show them only on the last page of the listing. By default, page level calculations are disabled.

Table Item Counts

All configurable tables show in the calculation section header a count of the number of objects used in the calculations. For page calculations, it shows the number of items shown on the current page. If table pagination is off, then it shows the count of all objects across all pages. This clarifies the scope of the calculations displayed on the page. For example:

- table page 1 has 10 objects
- table page 2 has 10 objects
- table page 3 has 5 objects

If pagination is on and you are looking at page 2, then the item count is displayed as **Page (10 items)**.

If pagination is off, then the item count is displayed as **All (25 items)** in the final calculation section.

To enable page level calculations, set the property `emxFramework.ShowPageCalculations` to true in `emxSystem.properties`. This setting is false by default.

Table Calculations

All configurable tables can provide calculations on columns within the table. This includes the sum of values or the average of all values in the column. You can also show the maximum, minimum, or median value of the column.

Calculations can be applied to any column in the table that contains numeric-only data except for the first column, which generally contains the object name. The calculation is performed only for the displayed data. For example:

- table page 1 has objects 1 through 10
- table page 2 has objects 11 through 20
- table page 3 has objects 21 through 25

If pagination is on and you are looking at page 2, then the calculation is applied to objects 11 through 20 only. If pagination is off, then the calculation is applied to all objects (1 through 25).

The following rules apply to page level calculations:

Rule	Description
Show on every page if paginated	If the listing is paginated, page calculations will appear on every page.
Disable if pagination is disabled	Page calculations will <i>not</i> appear if pagination is disabled since they would simply be redundant with the final calculations.
Disable if only one page	Page calculations will <i>not</i> be shown in a paginated view if there is only a single page to be displayed.

Final calculations use all the values for all pages. This section will always be shown if any calculation settings are enabled, and will appear on the last page of the listing.

For pages that have pagination enabled and contain more than one page, the following shows an example of a page layout that includes several page level calculations.

TechSpecification-5000: Microbiological Characteristics

Technical Specification | rev A | Draft

☐	Name	Microbiological Count	Microbiological Requirement	Absence Of	Total Count	
☐	 MicrobiologicalCharacteristic-5000	17	Spore Count per mm requirement	Splatter	17	
☐	 MicrobiologicalCharacteristic-6000	88	Chromatic Comparison requirement	0	88	
☐	 MicrobiologicalCharacteristic-7000	2	Polarized Cells requirement	100 a particles	2	
Page (3 items)						
Total		107			107	
Average		35				
Maximum		88				
Minimum		2				
Median		17				

For pages that have pagination enabled and contain more than one page, the following shows an example of both the page level calculations and the final calculations that will be shown at the end of the table.

TechSpecification-5000: Microbiological Characteristics

Technical Specification | rev A | Draft

☐	Name	Microbiological Count	Microbiological Requirement	Absence Of	Total Count	
☐	 MicrobiologicalCharacteristic-5000	17	Spore Count per mm requirement	Splatter	17	
☐	 MicrobiologicalCharacteristic-6000	88	Chromatic Comparison requirement	0	88	
☐	 MicrobiologicalCharacteristic-7000	2	Polarized Cells requirement	100 a particles	2	
Page (3 items)						
Total		107			107	
Average		35				
Maximum		88				
Minimum		2				
Median		17				
All (126 items)						
Total		2107			2107	
Average		35				
Maximum		88				
Minimum		2				
Median		35				

Page 10 of 10





Defining Calculation Settings

The following settings can be defined for any numerical column in the table. Labels will be shown in the first column of the listing that is not a selection checkbox, a selection radio button, or an icon image. The label column will span the remaining columns up to the first column that contains a calculation. The labels listed in the following table display by default with each of the calculation types.

Setting Name	Values	Description	Label
Calculate Sum	true false	A setting of true shows the total of the column values.	Sum
Calculate Average	true false	A setting of true shows the average of the column values. This is calculated using the sum of the values in the column divided by number of rows.	Average
Calculate Maximum	true false	A setting of true shows the largest number of the column values.	Max

Setting Name	Values	Description	Label
Calculate Minimum	true false	A setting of true shows the smallest number of the column values.	Min
Calculate Median	true false	A setting of true shows the middle number of the column values. If there is an even number of values, the arithmetic mean of the two middle values is shown.	Median

Improving Performance of Table Columns

This section explains how to improve the performance of a dynamic user interface table column whose Column Type setting is set to program or programHTMLOutput. Both these Column Type settings use a method in a Java Program Object (JPO) to obtain the column data.

The input to the JPO method that produces the column data must be a HashMap, which contains a MapList (list of object ids) and a HashMap (request parameters). Most table columns need to get a specific set of data for each object, do some processing on the data, and then add the processed data to the result set.

To accomplish this with the minimum number of calls to the database, use the ADK method `BusinessObject.getSelectedBusinessObjectData(...)`. Use this method, available in ADK 9.5.3.0 and higher, instead of the deprecated `context.getSelectedBusinessObjectData` method.

BusinessObject.getSelectedBusinessObjectData		
Parameters	context	the context for this request
	oidList	a StringList of OIDs
	selectStmts	a StringList of select statements (name, owner, etc.)
Returns	a complete set of data of type <code>BusinessObjectWithSelectlist</code> , for all the business objects you passed in as a list	

Parse this output data and process it as required. Then add it to the resulting column values before returning from the JPO. Using this method, you make only one database call to get the complete set of data required to process the entire column.

DO NOT use the following approach: Looping through the input MapList to get each object ID, then instantiating the business object for each objectID to get the set of data for that object and processing them to add to the column result list. This approach causes the program to access the database as many times as the number of rows in the column. For example, if there are 50 rows, the program makes 50 database calls to get the data required for this one column.

To see a sample JPO program that uses the `getSelectedBusinessObjectData` method, refer to the next section, [Sample JPO for Getting Table Column Values](#).

Sample JPO for Getting Table Column Values

The following is a sample Java Program Object (JPO) for getting the current vault for each object in the list in table rows. This is the kind of JPO you can use when the Column Type

setting for a table column is set to program or programHTMLOutput. The method “getVault” in this program processes the object list and returns the vault values.

For recommendations on improving the performance of programmatic table columns, see [Improving Performance of Table Columns](#).

The vault names can be obtained by configuring the table column as businessobject select expression “vault” instead of defining the column Type as “program”. The “vault” example is used for simplicity and to illustrate the steps involved in writing the JPO for a table column.

```
/*
 * emxTableProgram
 *
 * Copyright (c) 1992-2002 MatrixOne, Inc.
 *
 * All Rights Reserved.
 * This program contains proprietary and trade secret
information of
 * MatrixOne, Inc. Copyright notice is precautionary only and
does
 * not evidence any actual or intended publication of such
program.
 *
 * static const char RCSID[] = $Id: Exp $
 */

import matrix.db.*;
import matrix.util.*;
import java.io.*;
import java.util.*;
import com.matrixone.framework.beans.*;
import com.matrixone.framework.util.*;
import com.matrixone.framework.ui.*;

/**
 * @version AEF 9.5.0.0 - Copyright (c) 2002, MatrixOne, Inc.
 */
public class ${CLASSNAME}
{

    /**
     *
     * @param context the Matrix <code>Context</code> object
     * @param args holds no arguments
     * @throws Exception if the operation fails
     * @since AEF 9.5.0.0
     * @grade 0
     */
    public ${CLASSNAME} (Context context, String[] args)
        throws Exception
    {
```

```

        if (!context.isConnected())
            throw new Exception("not supported on desktop
client");
    }

    /**
     * This method is executed if a specific method is not
specified.
     *
     * @param context the Matrix <code>Context</code> object
     * @param args holds no arguments
     * @returns nothing
     * @throws Exception if the operation fails
     * @since AEF 9.5.0.0
     */
    public int mxMain(Context context, String[] args)
        throws Exception
    {
        if (!context.isConnected())
            throw new Exception("not supported on desktop
client");
        return 0;
    }

    /**
     * get Vault for the objects.
     *
     * @param context the Matrix <code>Context</code> object
     * @param args holds the following input arguments:
     *      0 - HashMap programMap
     * @returns vector of Vault names
     * @throws Exception if the operation fails
     * @since AEF 9.5.0.0
     */
    public static Vector getVault(Context context, String[]
args)
        throws Exception
    {
        HashMap programMap = (HashMap) JPO.unpackArgs(args);

        MapList relBusObjPageList =
(MapList)programMap.get("objectList");
        HashMap paramMap =
(HashMap)programMap.get("paramList");

        Vector vaultList = new Vector();
        StringList listSelect = new StringList(1);
        listSelect.addElement("vault");
    }

```

```

        String objIdArray[] = new
String[relBusObjPageList.size()];

        if ( relBusObjPageList != null)
        {
            for (int i = 0; i < relBusObjPageList.size(); i++)
                objIdArray[i] =
                (String) ((HashMap) relBusObjPageList.get(i)).get("id");

            // get the vault for the object
            BusinessObjectWithSelectList vaultSelectList = null;
            vaultSelectList =
            BusinessObject.getSelectBusinessObjectData(objIdArray,
            listSelect);

            for (int i = 0; i < relBusObjPageList.size(); i++)
            {
                String vaultName =
                vaultSelectList.getElement(i).getSelectData("vault");
                vaultList.add(vaultName);
            }
        }

        return vaultList;
    }
}

```

Parameters and Settings for Inquiry Objects

This table describes how to fill in the parameters for inquiry objects. For specific instructions on how to create inquiry objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Arguments	The arguments are used to replace the macros within the code. The symbolic names are translated into the actual name during substitution. ID is a reserved keyword for the inquiry objects used with the configurable table components. If the macro <code>{ID}</code> is used in the code section, the argument ID must be assigned to any dummy value. This value is replaced by the request parameter “objectId” at run time.	PART=type_Part EBOM_REL=relationship_EBOM ID = dummy
Code	The code section determines the output object list for a specific business requirement. The code is generally an MQL temp query or expand bus command that selects the found objects’ ids. The macro <code>{ID}</code> is substituted with the business object id during run time. This object Id is available to <code>emxTable.jsp</code>, passed in from the tree node or any other commands that operate based on a specific business object. For example, typically when <code>emxTable.jsp</code> is called from a tree, the “objectId” parameter is passed with the specific OID assigned to the current tree node and that OID will get substituted with <code>{ID}</code> at run time. <i>Note that this <code>{ID}</code> is not same as the RPE variable <code>{OID}</code>.</i> All the admin types like Type names (Part, Buyer Desk), attribute names, relationship names must be defined as macros and each macro will be assigned with its symbolic name in the arguments list. All other macros (like <code>{PART}</code>, <code>{EBOM_REL}</code>) defined within the code are to be assigned to the appropriate symbolic names in the Argument, if required. The macros are evaluated at run time by converting the symbolic name to actual name.	<pre>temp query bus \${PART} * * select id dump expand bus \${ID} from relationship \${EBOM_REL} select businessobject id select relationship id dump </pre>

URL Parameters Accepted by emxTable.jsp

This table lists the parameters that emxTable.jsp can use. You can add these parameters to the href parameter for the component that calls the table. For example, when you specify the emxTable.jsp to be called from a tree category, you can add these parameters to the href parameter for the command object.

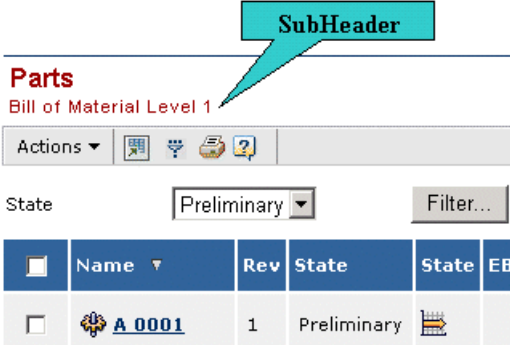
Parameter	Description	Accepted Input Values
CancelButton	Displays the Cancel link in the table footer frame. When a user clicks Cancel, the whole browser window gets closed. So the Cancel link can only be used for a table page that is opened in new window. The label for the Cancel link can be configured using the CancelLabel parameter. If CancelLabel is not passed in, the default label "Cancel" is used.	true—Shows the Cancel link. Use only for tables displayed in popup windows. false (Default)—Does not show the link.
CancelLabel	Defines the label for the Cancel link, if you want something other than "Cancel".	Any static text or string resource id. CancelLabel=emxEngineeringCentral.Common.Cancel
editLink	Use to display the Edit toolbar item on the table page. The Edit link is automatically configured to call the editable version of the current table using emxTableEdit.jsp.	true false (default)
Export	Shows or hides the Export  tool on the page toolbar. When users click the tool, the system exports the table data to a file in the user's preferred format in their preferred format: CSV, HTML, or Text. Users choose their preferred format using the Preferences tool in the global toolbar. For information on configuring the export preference, see Export Table Data Format . For instructions on using the preference page, click Help from the preference page. If the user chooses CSV or Text for their export format, the export does not include Icon column type and programHTMLOutput columns. In order to get the column values exported when the column type is "programHTMLOutput," you must also set "Export=true" for the column.	true (Default)—The Export tool displays, allowing users to export the table data. false—The tool does not display.
FilterFramePage	Specifies the name of the JSP that will be used in the custom filter frame, which is between the header and body. You can use the directory macros to refer the location of the page. The path of the page is relative from the ematrix/common directory. For details on implementing the custom filter, see Implementing a Custom Filter .	\${SUITE_DIR}/ emxTeamProjectTableFilter.jsp\$ \${ROOT_DIR}/ emxPartECOSatusFiletr.jsp\$ \${COMMON_DIR}/ emxCommonStateFilter.jsp\$ emxCommonStateFilter.jsp
FilterFrameSize	Controls the size of the filter frame in pixels. The value should be at least 40, which is the default.	40 80 160

Parameter	Description	Accepted Input Values
header	The content of the heading that appears at the top of the table page.	Any alphanumeric text or a string resource ID. header=Buyer Desk header=emxQuoteCentral.AssignedPackages.AssignedPackages
headerRepeat	Specifies how often the header row should be repeated. A repeating header row makes it easier for users to identify the content of each column when scrolling through a long table. If the parameter is not used, the header row repeats every 15 rows (or whatever number is assigned to the emxSystem.properties key emxNavigator.UITable.HeaderRepeat).	Integer For example, 11 would mean the header row is repeated every 11 rows. To turn off headerRepeat, pass in 0 as the value for the parameter.
HelpMarker	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help .	String The naming convention for help markers “emxhelp” followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
inquiry	Specifies the inquiry admin object that should be used to retrieve the business objects to be included in the table and any inquiries for the table page filter list. emxTable.jsp uses the first inquiry object to display the list in the table when the page is first loaded. It uses all inquiry objects for the table filter list in the header. If there is only one inquiry, the filter list in the header is not displayed. Currently, to build the filter list, you must use either JPOs or Inquiry objects. You cannot use both.	Names of inquiry admin object separated by a comma. inquiry=SCSBuyerDesk,SCSBuyerDeskAssigned inquiry=ENCAIParts, ENCREleasedParts
inquiryLabel	This parameter is used with the “inquiry” parameter. Specifies the labels for each inquiry in the filter list in the table page header. Each label is associated to the corresponding inquiry admin object name specified in the “inquiry” parameter. The label may have the actual text or the string resource id for internationalization.	One or more labels (equal to the number of objects specified in the “inquiry” parameter) for the inquiry admin object names separated by a comma. inquiryLabel=All,Assigned inquiryLabel=emxEngineeringCentral.Common.All, emxEngineeringCentral.Common.Released
massUpdate	Use to turn on or off mass update controls on the editable table page.	true (default)—Mass update is available for the page. false—Mass update not is available for the page.

Parameter	Description	Accepted Input Values
objectBased	Use to configure a table page so it lists items other than business objects. For example, the items such as collections and IconMail message are not business objects but can be listed in a table by passing the URL parameter objectBased=false. When this parameter is set to false, the JPO which gets the list should follow the same guidelines used for getting the object list. See Guidelines for Writing a JPO for Getting Object List. But the MapList containing the HashMaps with key name “id” should be assigned to the row identifier value and not the “objectId”. The value assigned to “id” is used as the row identifier for every row in the table. If the table has a checkbox column (named emxTableRowId), the checkbox value will be assigned with the value in “id”.	true (default)—The table page lists business objects. false—The table page lists items that are not business objects.
pagination	Specifies the number of rows to show per page. If the rows span more than one page, users can use the Left and Right Arrow buttons or the Page drop-down list to navigate to other pages. For more information on pagination controls, see How the Pagination Controls Work. If this parameter is not specified, the system uses the value set in the pagination property in emxSystem.properties. When installed, this property is set to 10 rows per page. If 0 is specified, all objects are listed on one page.	Any number. Use 0 if you don’t want to paginate. If 0 is specified, the pagination controls do not include the controls for navigation.
portalTable	Specifies the alternate table that can be used within a channel tab. Displaying table component listings in half page-width channels will typically cause the user to scroll horizontally to the view all columns in the table. For this situation, you can create an alternate table with fewer columns for display specifically within a channel tab. For example, if a standard ECO New Parts table contains 10 columns, you could define an alternate ECO New Parts table that contains just 4 columns.	Name of alternate table
PrinterFriendly	Specifies whether the table page should include the Printer Friendly tool. If this parameter is not included, the value is assumed to be True and the table page displays the Printer Friendly tool. You can have the system pass this parameter automatically by entering the Printer Friendly setting for the command object that calls the table page.	true false

Parameter	Description	Accepted Input Values
program	Used only when there is no “inquiry” parameter is passed in to emxTable.jsp. This parameter is used as an alternative approach to the inquiry object approach. The “program” approach uses a JPO program object to fetch the object list to be displayed in the table. The parameter value can have one or more sets of values separated by a comma “,”. emxTable.jsp uses the first set of values (JPO name and method name) to display the list in the table when the page is first loaded. It uses all the sets for the table filter list in the header. If there is only one program, the filter in the header is not displayed. Currently, to build the filter list, you must use either JPOs or Inquiry objects. You cannot use both.	This parameter is assigned to one or more set of values that will form a JPO program name and the method name. The format of the parameter value is: program=<JPO program name>:<JPO method name> The program name and the method name are separated by a colon “:”. program=emxTableBuyerDesk:getBuyerDesk,emxTableBuyerDesk:getAssignedBuyerDesk For guidelines on writing a JPO for getting a list of business objects, see Guidelines for Writing a JPO for Getting Object List. For a sample of a JPO that gets a list of business objects, see Sample JPO for Getting List of Objects.
programLabel	Used with the parameter “program” defined above. Specifies the labels for each program in the filter list in the table page header. The value may contain one or more labels (equal to number of “program” values) separated by a comma “,”. Each label is associated to the corresponding JPO object and method name specified in “program” parameter. The label may have the actual text or the string resource id for internationalization.	programLabel =All,Assigned programLabel = emxEngineeringCentral.Common.All, emxEngineeringCentral.Common.Released
rememberSelection	Controls whether the system remembers the items a user selects on one page when the user navigates to another page in the table. The emxSystem.properties key emxNavigator.UITable.Pagination.RememberSelection sets the default for table pages. Use this parameter to override the default for a single page. For more information, see How the Pagination Controls Work.	true—Selected items are remembered across the table’s pages. false—Selected items are not remembered across pages.

Parameter	Description	Accepted Input Values
selection	Controls whether the table page adds a column of check boxes or radio buttons in the left-most column of the table.	multiple—Users can select more than one row in the table. A check box is displayed in the left column of each row. There is no access restriction for the check boxes. If this parameter is passed, it overrides any check box column added to the table admin object. For more information, see Column Values as Check Boxes. single—Users can select one row in the table. A radio button is displayed in the left column of each row. none—A selection column is not added to the table page. (A check box column can still be displayed by adding the column to the table admin object.)
sortColumnName	Specifies the column by which the table should be sorted when the page is first loaded. If no column is specified, the rows are listed in the order they are retrieved from the database.	Name of table column.
sortDirection	The direction to sort the rows by.	ascending (default)—Sort a to z or 0 to n. descending—Sort z to a or n to 0.
Style	Determines the cascading style sheet to use for the table page.	list (default)—For table pages displayed in the content frame of emxNavigator.jsp. By default, the standard emxUIList.css style sheet is used. The style sheet to use is defined in this emxSystem.properties key: emxNavigator.UITable.Style.List = styles/emxUIList.css dialog—For table pages that display in popup windows, such as search result pages. By default, uses the emxUISearch.css style sheet. The style sheet to use is defined in: emxNavigator.UITable.Style.Dialog = styles/emxUISearch.css

Parameter	Description	Accepted Input Values
subHeader	Creates a subHeader below the main header in the table header frame.  The screenshot shows a table header with a subheader 'Parts' and a callout box labeled 'SubHeader' pointing to it. Below the subheader is a 'Bill of Material Level 1' section with an 'Actions' dropdown, a 'State' dropdown set to 'Preliminary', and a 'Filter...' button. The table has columns: Name, Rev, State, State, and EB. The first row shows a part with name 'A 0001', revision '1', and state 'Preliminary'.	The value can be any static text or a string resource id. For example: <pre>subHeader=emxEngineeringCentral.Common.BOMLevel</pre> <pre>subHeader=Bill of Material Level 1</pre> The value can also include macros such as <code>\$<type></code> <code>\$<revision></code>.
SubmitURL	Displays the Submit link in the table footer frame. The value can be any valid JSP page, which gets run upon clicking Submit. The label for the Submit link can be configured using the SubmitLabel parameter. If SubmitLabel is not passed in, the default label “Submit” is used.	Any JSP. The path can include any directory macro. <pre>SubmitURL=\${SUITE_DIR}/emxBlank.jsp</pre>
SubmitLabel	Defines the label for the Submit link.	Any static text or string resource id. <pre>SubmitLabel=emxEngineeringCentral.Button.Next</pre>
table	Specifies the table object to use for presenting this targeted page. The table object has all the information needed to display the table, including the columns to present.	Name of table admin object. For example: <pre>table=SCSBuyerDesk</pre> <pre>table=ENCParts</pre>
TipPage	Specifies whether the table page should include the Tip Page tool and call a specific html or jsp when a user clicks the tool. If this setting is not included, the Tip tool is not included on the table page.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the table.jsp: <pre>TipPage=../doc/customcentral/tippage.html</pre>
TransactionType	Controls whether the table query is run within an Update transaction or Read transaction. Update transaction can be used whenever the code needs to update the database while fetching the object list.	read (default)—The table query is not run within an Update transaction. update—The table query (inquiry or JPO) will be run within an Update transaction.

Guidelines for Writing a JPO for Getting Object List

As described earlier, a JPO can be written and configured to get the list of object to be displayed in the table page instead of using inquiry objects.

A JPO program object for getting an object list must follow these specific rules:

- The JPO program must define a method with the exact same name specified in the href parameter as program=<program name>:<method name>.
- The method takes an input argument as a HashMap containing details of the request parameters available to the JSP page. Every parameter will be available as a key/value pair in the HashMap. In addition, a parameter called “languageStr” will be available to get the language setting (such as en, fr, ja, etc).
- The method returns a MapList containing the list objects. This MapList will contain a list of HashMaps with the values of OIDs and RelIDs. Every HashMap will contain the value for OID and RelID belonging to one table row.

This is the structure of each HashMap belonging to a table row:

- Key “id” is assigned to the business object Id (OID) to be used in the table row.
- Key “id[connection]” is assigned to the relationship id (RelID) to be used (if required) for the table row.

For example...

MapList		
Object	Key	Value
HashMap 1	id	Value
	id[connection]	Value
HashMap 2	id	Value
	id[connection]	Value
...	id	Value
	id[connection]	Value
HashMap n	id	Value
	id[connection]	Value

Here is a JPO function template:

```
public static Object methodName(Context context, String[] args)
throws Exception
{
    HashMap paramMap = (HashMap)JPO.unpackArgs(args);
    MapList objectList = new MapList();

    // Get the required parameter values from "paramMap" - if
    // required
    String languageStr = (String)paramMap.get("languageStr");
    String objectId = (String)paramMap.get("objectId");
    String selectedFilter =
        (String)paramMap.get("selectedFilter");
```

```
// Get the object elements - OIDs and RELIDs using Beans or ADK
methods

// Build the MapList with List of HashMaps containg OIDs and
RelIDs

return objectList;
}
```

Sample JPO for Getting List of Objects

The following is a sample java code (JPO program object) for getting the list of objects that are to be displayed in a table.

```
/*
 * emxTableListJPO
 *
 * Copyright (c) 1992-2002 MatrixOne, Inc.
 *
 * All Rights Reserved.
 * This program contains proprietary and trade secret
information of
 * MatrixOne, Inc. Copyright notice is precautionary only and
does
 * not evidence any actual or intended publication of such
program.
 *
 * static const char RCSID[] = $Id: Exp $
 */

import matrix.db.*;
import matrix.util.*;
import java.io.*;
import java.util.*;
import com.matrixone.framework.beans.*;
import com.matrixone.framework.util.*;
import com.matrixone.framework.ui.*;

/**
 * @version AEF 9.5.0.0 - Copyright (c) 2002, MatrixOne, Inc.
 */
public class ${CLASSNAME}
{

    /**
     *
     * @param context the Matrix <code>Context</code> object
     * @param args holds no arguments

```

```

        * @throws Exception if the operation fails
        * @since AEF 9.5.0.0
        * @grade 0
        */
public ${CLASSNAME} (Context context, String[] args)
    throws Exception
{
    if (!context.isConnected())
        throw new Exception("not supported on desktop
client");
}

/**
 * This method is executed if a specific method is not
specified.
 *
 * @param context the Matrix <code>Context</code> object
 * @param args holds no arguments
 * @returns nothing
 * @throws Exception if the operation fails
 * @since AEF 9.5.0.0
 */
public int mxMain(Context context, String[] args)
    throws Exception
{
    if (!context.isConnected())
        throw new Exception("not supported on desktop
client");
    return 0;
}

/**
 * Get the list of objects.
 *
 * @param context the Matrix <code>Context</code> object
 * @param args holds the following input arguments:
 *      0 - objectList MapList
 * @returns Object of type MapList
 * @throws Exception if the operation fails
 * @since AEF 9.5.1.2
 */
public MapList getRTSList(Context context, String[] args)
    throws Exception
{
    HashMap paramMap = (HashMap)JPO.unpackArgs(args);
    String objectId = (String)paramMap.get("objectId");
    String selectedFilter =
(String)paramMap.get("selectedFilter");
    String programList = (String)paramMap.get("program");

```

```

        String filterName = "All";

        MapList relBusObjPageList = new MapList();
        SelectList selectListRTS = new SelectList(1);

        Person person = Person.getPerson(context);
        selectListRTS.add(person.SELECT_ID);
        relBusObjPageList = person.getOwnedRTSs(context,
        selectListRTS, true, null, null, false, filterName);

        return relBusObjPageList;
    }
}

```

Writing a JPO for Updating Column Values

This JPO approach is used for obtaining the column values, when the column is configured with the settings:

- Update Program=JPO name
- Update Function=JPO method name

For processing the data, the input parameter may be used by the method. The return type of this method is a Boolean, which will be true if the update is successful. Otherwise it will be false.

The JPO function template is provided below:

```

public static int methodName(Context context, String[] args) throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);

    // Get the required parameter values from "programMap" - as required
    String objectId = (String) programMap.get("objectId ");
    String relId = (String) programMap.get("relId ");
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    String languageStr = (String) programMap.get("languageStr");
    String newValue = (String) programMap.get("New Value");
    String oldValue = (String) programMap.get("Old Value");

    // Define and add selects if required

    // Process the information to set the field values for the current object
    ....

    return 0;
}

```

See [Sample JPO for Getting Table Column Values](#).

This graphic shows an example of a custom filter implemented on a table page.

RFQs

Actions   

State

		Name ▾	Round	State	Owner	Quote Requested By Date	Response Status	Package
☐		 RFQ-1000	1	Started	John Smith	Feb 28, 2003 9:00:00 AM	0/4	Package-2244
☐		 RFQ-2000	2	Sent	Rebecca George	Feb 28, 2003 9:00:00 AM	1/3	
☐		RFQ-3000	4	Response Completed	Alice Tenon	Feb 28, 2003 9:00:00 AM	0/2	
☐		 RFQ-4000	2	Started	Albert Swanson	Feb 28, 2003 9:00:00 AM	2/2	
☐		RFQ-5000	1	Sent	Thomas Wataki	Feb 28, 2003 9:00:00 AM	0/2	

To implement custom filtering

1. Create a custom JSP page that will access the table data and do the filtering based on the user input.
The custom JSP page will get an input request parameter "tableID." Read this parameter from the request object. This parameter will be the key to obtain the current table data.
2. Using the table ID key, call the table bean methods described below to obtain the table data.
3. The table data contains the complete list of objects and other input parameter details. Process the table data to get a filtered list and set the filtered objects list back in the table data using a bean method.
4. Refresh the table display page using a JavaScript API.

The following code is a template, to accomplish the above steps.

```

////////////////////// START Code Template ////////////////////////
//Declare the table bean with session scope
<jsp:useBean id="tableBean"
class="com.matrixone.framework.ui.UITable" scope="session"/>
// Start the JSP code
<%
    // Get the table ID from the request
    String tableID = Request.getParameter(request, "tableID");
    // Obtain the object list for the current table using the
    tableID
    MapList relBusObjList = tableBean.getObjectList(tableID);
    // Obtain the requestMap which contain all the request params to
    emxTable.jsp
    HashMap requestMap = tableBean.getRequestMap(tableID);
    ...

```

```
// Process the object List and filter it to create a new MapList
of filtered object List
MapList filteredObjPageList = ....
...
// Set the filtered object list for the current table using the
tableID
tableBean.setFilteredObjectList(tableID, filteredObjPageList);

// End JSP code and Start the Javascript code to call the
refresh method as below
%>
<script language="JavaScript">
    parent.refreshTableBody();
</script>
////////// END Code Template //////////
```

Sample JSPs for Custom Filtering

Two JSP pages may be required to implement the custom filtering within the configurable table:

1. The first page creates the user interface to initiate the filter.
2. The second page takes the input from the first page and processes the data. It then filters the data and updates the display. Only the first page is assigned as the URL parameter to emxTable.jsp.

Here are samples of both JSPs.

First JSP (for UI): emxTableFilterIncludeSample.jsp

```
<html>
<%@include file = "emxNavigatorInclude.jsp"%>
<jsp:useBean id="formEditBean"
class="com.matrixone.framework.ui.UIForm" scope="session"/>
<%
String tableID = Request.getParameter(request, "tableID");
%>
<head>
<link rel="stylesheet" href="styles/emxUIDefault.css"
type="text/css">
<link rel="stylesheet" href="styles/emxUIList.css" type="text/
css">
<script language="javascript">
    function filterData()
    {
        var filterProcessURL="emxTableFilterIncludeProcess.jsp";
        document.filterIncludeForm.action = filterProcessURL;
        document.filterIncludeForm.target = "listHidden";
        document.filterIncludeForm.submit();
    }
</script>
</head>
<body>
```

```

<form name="filterIncludeForm">
<table border="0" cellspacing="2" cellpadding="0" width="100%">
<tr>
<td width="99%">
<table border="0" cellspacing="0" cellpadding="0" width="100%">
<tr>
<td class="pageBorder"></td>
</tr>
</table>
</td></tr>
</table>
<table>
<tr>
<td width="100"><label>State</label></td>
<td class="inputField" ><select name="filterState" id="">
<option value="Preliminary">Preliminary</option>
<option value="Release">Release</option>
<option value="Create" selected >Create</option>
<option value="Obsolete">Obsolete</option>
</select>
</td>
<td width="50"> &nbsp; </td>
<td width="150"><input type="button" name="btnFilter"
value="Filter..." onclick="javascript:filterData()"></td>
</tr>
</table>
<input type="hidden" name="tableID" value="<%=tableID%>">
</form>
</body>
</html>

```

Second JSP (for processing and refreshing): emxTableFilterIncludeSample.jsp

```

<%@include file = "emxNavigatorInclude.jsp"%>
<jsp:useBean id="tableBean"
class="com.matrixone.framework.ui.UITable" scope="session"/>
<%
    String tableID = Request.getParameter(request, "tableID");
    String selectedState = Request.getParameter(request,
"filterState");
    // Get business object list from session level Table bean
    // HashMap tableData = tableBean.getTableData(tableID);
    MapList relBusObjList = tableBean.getObjectList(tableID);
    HashMap requestMap = tableBean.getRequestMap(tableID);
    MapList filteredObjPageList = new MapList();
    StringList busSelects = new StringList();
    BusinessObjectWithSelectList busObjwsl = null;
    if ( relBusObjList != null)
    {
        String objIdArray[] = new String[relBusObjList.size()];
        busSelects.add("current");
    }

```

```

        for (int i = 0; i < relBusObjList.size(); i++)
            objIdArray[i] =
                (String) ((HashMap) relBusObjList.get(i)).get("id");
        busObjwsl =
            BusinessObject.selectBusinessObjectData(context, objIdArray,
            busSelects);
        for (int i = 0; i < relBusObjList.size(); i++)
        {
            String currentStatus =
                (String) busObjwsl.getElement(i).getSelectData("current");
            if (currentStatus != null &&
                currentStatus.equals(selectedState) )
                filteredObjPageList.add(relBusObjList.get(i));
        }
        tableBean.setFilteredObjectList(tableID,
        filteredObjPageList);
    %>
<script language="JavaScript">
    parent.refreshTableBody();
</script>

```

Custom Sorting Programs

This section contains instructions and samples for a JPO that contains a custom algorithm for sorting a table based on the values in a column. To use such a program, the table column parameter sorttype must be set to other and the table column setting Sort Program must contain the name of the JPO.

- [Guidelines for Custom Sort JPO](#)
- [Sample Sorting Code](#)
- [Framework's Custom Sorting Programs](#)

Guidelines for Custom Sort JPO

A JPO defined as a Sort Program must follow these guidelines and definitions:

- JPO class must extend emxCommonBaseComparator JPO.
emxCommonBaseComparator JPO is installed by the framework. It contains the entire base implementations required to make custom sorting work.

```

public class ${CLASSNAME} extends
    ${CLASS:emxCommonBaseComparator}
{
}

```
- JPO class must define a default constructor as shown.

```

// default constructor
public ${CLASSNAME} ( )
{
}

```
- The class must define a method called compare(..)
The template for the compare method definition is:

```

public int compare(Object object1, Object object2)
{
    // Logic to compare the two objects and return the integer
    value
}

```

Input parameters and return values of the compare method are:

- param object1—the first object to compare.
- param object2—the second object to compare.
- Returns:
 - A negative integer if first argument is less than second argument.
 - A zero if the arguments are equal.
 - A positive integer if the first argument is greater than second argument.
- The object1 and object2 is of type Map.
- The information about sort and Map object passed in is stored in keys field of base comparator (emxCommonBaseComparator). The field “keys” is again a Map object containing the following keys.

Key Name	Description
Type	Specifies the type of sort and should always be set to “other.”
columnKey	Identifies the key name to get sort column values from Map object1 and object2.
direction	Direction of sort: ascending or descending.

Sample Sorting Code

The following sample code can be used to get column values from object1 and object2.

```

public int compare(Object object1, Object object2)
{
    // Logic to get column values to compare from object1 and
    object2
    // Get sort info keys
    Map sortKeys = getSortKeys ( );
    // Get column key name defined to get column values from
    object1 and object2
    String columnKey = (String) sortKeys.get("columnKey");
    // Get column values from object1 and object2
    Map m1 = (Map)object1;
    Map m2 = (Map)object2;
    String columnValue1 = (String)m1.get(columnKey);
    String columnValue2 = (String)m2.get(columnKey);

    // Code to custom compare columnValue1 and 2 goes here.
    .
    .
    .
}

```

Framework's Custom Sorting Programs

The framework provides two sample JPOs for custom sorting. They can be used in Engineering Central to sort Find Numbers on EBOM relationships.

JPO Name	Description
emxSortNumericAlpha Larger	This JPO comparator can be used to sort find numbers. The compare method does following. • If both objects are numbers, do a numeric compare. • If both objects are alphanumeric, do a string compare. • If one object is a number and the other is alphanumeric, then Alphanumeric > Number.
emxSortNumericAlpha Smaller	This JPO comparator can be used to sort find numbers. The compare method does following. • If both objects are numbers, do a numeric compare. • If both objects are alphanumeric, do a string compare. • If one object is a number and other alphanumeric, then Number > Alphanumeric.

Building Form Pages

Two Modes of the Form Page

The form page displays characteristics of a business object, for example, attributes, basics, and other properties. Each characteristic is listed in a row, also called a field, on the form page. The form page comes in two modes: View and Edit. The View form page displays information but users cannot edit the information. View form pages can contain an Actions menu that lets users execute actions related to the business object.

The dynamic user interface is designed so the View mode form page for a business object displays when a user views details for the object. The View form is in the right frame, called the content frame, and the navigation tree for the object is in the left frame.

Form page in View mode

Header

Page toolbar with Actions menu

Form rows or fields

Property: Mechanical Part CM-201010-01 1

Actions ▾

Type	Mechanical Part
Name	 CM-201010-01
Part Classification	Unassigned
Current State	Release
Find Number	1
Connected ECOs	 ECO-007000
Policy	EC Part
Date Originated	Jul 9, 2002 10:04:40 PM
Vault (basic)	eService Sample
Originator (Range program)	Test DesignEngineer
Unit of Measure	EA (each)
Description	Document Feeder Assembly
Vault	eService Sample
EBOM	 CM-202010-01
	 CM-202020-01
	

The Edit mode of the form displays when the user chooses the Edit toolbar item on the View form’s Actions menu. The Edit mode of the form page opens in a popup window. This graphic shows an example of the Edit mode for a form page.

Form page in Edit mode

Header

Form rows or fields

Property: Mechanical Part CM-201010-01 1

Fields in red italics are required ?

Type	Mechanical Part
<i>Name</i>	 CM-201010-01
Part Classification	Unassigned ▾
Current State	Release ▾
Find Number	1
Connected ECOs	 ECO-007000
Policy	EC Part
Date Originated	Jul 9, 2002 10:04:40 PM 
Vault (basic)	eService Sample ...
<i>Originator (Range program)</i>	Test DesignEngineer ▾
Unit of Measure	☐ LB (pound) ☐ IN (inch) ☐ GA (gallon) ☐ FT (feet) ☒ EA (each)

 Done  Cancel

The default mode for form pages is View mode. You make a form page display in Edit mode by adding the URL parameter `mode=edit` to `emxForm.jsp`. Even when a form displays in Edit mode, you can display the fields (form rows) as read only using the field setting “Editable”.

Forms can be configured to display in an editable mode. The edit link can be configured in two ways:

1. By passing in a URL parameter called `editLink=true`. This approach has no access control.
2. By configuring the toolbar command shown in the form with `href` assigned to `emxFormEdit.jsp`. This approach can be used when access control is needed for the edit command. It also needs to have the following associated settings:
 - `Submit=true`
 - `Popup Modal=true`
 - `Target Location=popup`

Links on Form Pages

Only the View mode can have a toolbar in the header. For instruction on configuring the toolbar menu, see [Building Toolbars](#).

The Edit form page always has a Done and Cancel button at the bottom. The Done button commits the changes to the database. Only the attributes that are modified are updated. In case of any exception, an error message is displayed as an alert message. Also, the transaction is aborted if any exception occurs during the process of update and all the updated values are rolled back to the original values.

The Cancel button aborts any changes made and closes the window.

Form Page HTML Components

The main JSP page in the configurable form component is called `emxForm.jsp`. This page is the top level frameset and mainly controls whether the displayed page is in View or Edit mode based on the input parameters passed with the URL.

This table lists the frames and forms that make up the View mode of the form page.

Frame Location	Frame Name	Form Name	Includes:
header	formViewHeader	—	Page label, Actions menu, page toolbar
body	formViewDisplay	frmFormView	All fields in web form object in read-only display
hidden	formViewHidden	—	—

This table lists the frames and forms that make up the Edit mode of the form page.

Frame Location	Frame Name	Form Name	Includes:
header	formEditHeader	—	Page label, help icon
body	formEditDisplay	editDataForm	All fields in web form object in editable display (if configured as editable) Names of HTML controls are the same as the fields names defined in the web form admin object.
footer	formEditFooter	editFooter	—
hidden	formEditHidden	—	—

Building a Form Page

Mechanisms that Define a Form Page

The configurable JSP `emxForm.jsp` creates form pages. This JSP accepts many parameters that define the content and behavior of the page. These parameters are passed to the JSP through the href URL.

The main components of the form page—the rows or rows of information and the Actions menu (Action menus are for read-only forms only)—are defined as admin objects in the

database. The names of these objects are passed to the emxForm.jsp through its parameters.

There should be a web form admin object for each unique set of fields. For example, suppose the details page for ECRs needs to show one set of attributes and the details page for parts needs to show another set. You would need a web form admin object for each of these forms, maybe called CustomECRBasics and CustomPartBasics.

Steps for Creating a Form Page and Linking it to an Application

Users typically access form pages by selecting an action from an Actions menu, a tree, or tree category. The component that calls the form page must specify the configurable JSP for form pages, emxForm.jsp, in the href parameter. The following procedure lists the main steps needed for creating a form page and connecting it to an application. The graphics illustrate the objects needed to create a form page that shows details for ECRs. The ECR details page is linked to the tree for ECRs.

ECR-008010

Approvals

Lifecycle

Obsolete Parts

Revised Parts

Revised Specs

Supporting Documents

Route

History

ECR-008010 rev -: Properties

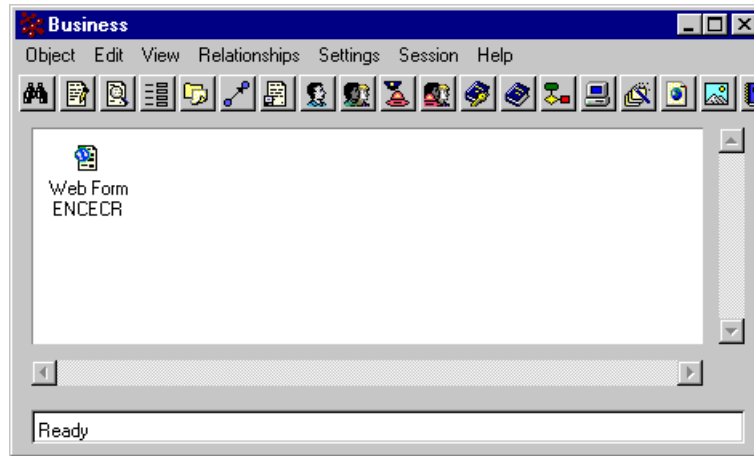
Actions

Type	ECR
Name	006000
Revision	-
Vault	eService Sample
Description	
Owner	Test DesignEngineer
General Description of Change	Move dimension in zone A-4
Reason for Change	To fix problem in over heating.
Reviewers Comments	
Category of Change	Drafting Error
Responsible Design Engineer	Test1 DesignEngineer
ECR Evaluator	Test1 DesignEngineer
Originator	Test DesignEngineer

To create a form page and connect it to an application

1. Create and configure a web form admin object and define the form fields to be displayed in the form page.

When you define form fields, you define the data that should be in the fields, the field label heading, the order in which the fields should appear, and who can access each field. For a description of how to fill in the parameters and settings for web forms and fields, see *Parameters for Web Form Objects*. For naming conventions, see *Naming Conventions for UI Administrative Objects*.



2. If the form page is View mode and you want to include an Action menu, configure the action menu and toolbar items using the instructions in [Building the Menus](#).

You can have the system add an Edit link to the top of a View mode form page automatically by passing the `editLink=true` parameter to `emxForm.jsp`. The link calls the same form page but displays it in Edit mode.

3. In the href parameter for the command object that should call the form page (or in the JSP if a JSP is calling the page), enter URL to display the form page. The URL should include these elements:
 - Since `emxForm.jsp` is in the `ematrix/common` directory, the URL should begin with the macro for the common directory: `${COMMON_DIR}`.
 - The URL should include `emxForm.jsp`.
 - The URL should include any parameters needed to display the page, particularly the mode parameter to specify whether it should be Edit or View mode. If the mode parameter is not included, the form defaults to View mode. For a description of the parameters accepted by `emxForm.jsp`, see [URL Parameters Accepted by emxForm.jsp](#).

For example, the following URL calls the form page in Edit mode:

```
${COMMON_DIR}/
emxForm.jsp?form=BuyerDesk&mode=edit&objectId=3243.32424.232&re
lId=3432.2342.2344&header=BuyerDesk&toolbar=SCSBuyerDesktoolbar
.
```

This URL calls the form in View mode:

```
${COMMON_DIR}/
emxForm.jsp?form=BuyerDesk&objectId=3243.32424.232&relId=3432.2
342.2344&header=BuyerDesk&toolbar=SCSBuyerDesktoolbar&editLink=
true
```

4. If you are working with the Web-based user interface as you are making changes and want to see your changes in the user interface, click the **Reload Cache** tool in the global toolbar and click the browser **Refresh** button.

The cache is refreshed automatically when the component age expires. This setting is in `emxSystem.properties`.

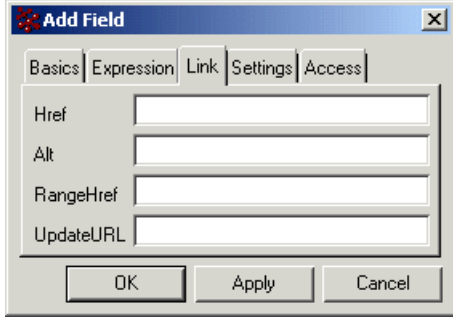
Only persons assigned to the Administration Manager role have access to the Reload Cache tool.

Parameters for Web Form Objects

This table describes how to fill in the parameters for web form objects. The definition of a web form consists primarily of definitions of the fields that make up the form. All the parameters listed below apply to fields within a form except those that are indicated as applying to the form itself. For specific instructions on how to create web form objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Access (user MQL command)	Use to specify the persons, roles, and groups who can access the field. To make the field available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access.	Names of group, role, person administrative objects. or All (default)
Alt	Use to define the ToolTip text that should display when users move their mouse pointer over fields configured to display images. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components . The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	emxFramework.Common.ProjectDetails Description
Applies To	Use to specify the item to apply the select expression to: the business object or relationship. Depending on which one is selected, the system passes a valid business object ID or relationship ID to emxForm.jsp. For example, emxForm.jsp?objectId=xxx or emxForm.jsp?relId=yyy. Dynamic UI table and tree components pass the objectId and relId automatically to emxForm.jsp, so the href link can be configured without passing IDs.	Business Object Relationship
Description	Use to give a brief description of how the form or field should be used. The web form admin object and each field within it has a description parameter.	Text string
Expression	Use to enter the select expression to get the field data. This expression is applied to either the relationship or business object, as specified in the Applies To options. If the expression is a business object basic or an attribute that will be editable, make sure you include the Field Type setting with either the basic or attribute value. Note that the Expression and Applies to options in Business are equivalent to the businessobject and relationship MQL commands. To see an example of a field configured to use an expression, see Field Values as Select Expressions .	For business objects: • type • name • current • \$<attribute[attribute_Originator].value> For relationships: • \$<attribute[attribute_FindNumber].value> • \$<attribute[attribute_Qty].value>

Parameter	Description	Accepted Values/Examples
Label	Use to enter the text that should appear as the label for the field. Make sure Custom Label is checked so the system gets the label you specify. If Custom Label is unchecked, the system uses the expression as the label. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components. The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	emxEngineeringCentral.common.N ame emxTeam.Common.ProjectName Description
href	Use to specify the URL that gets executed when the field data is clicked. This parameter is used only for the View mode. In Edit mode, it is ignored because fields in Edit mode are not shown with a hyperlink. When a user clicks the hyperlinked field data, the system passes the objectId parameter as part of the URL. By default, the value for the objectId parameter is the ID of the business object the form page applies to. Using the Alternate OID expression setting, you can configure the field to pass the ID returned from a different business object. The value for the href parameter should be a JSP and any associated parameters. You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Using Macros and Expressions in Dynamic UI Components in Chapter 8.	<code>\${COMMON_DIR}/emxTree.jsp</code> <code>\${SUITE_DIR}/</code> <code>emxpartEditPartDialog.jsp</code> To see an example of a field that uses an href, see Field Values as Hyperlink and Type Icon, Field Values with Hyperlinked Data Using an Alternate OID and Alternate Type Icon, and Field Values as Hyperlinked Image.
Name	Use to enter the name of the field used as identifier for the field within the web form object.	Name Revision Material Category
Name (of web form)	Use to enter the name of the web form admin object. For naming conventions, see Naming Conventions for UI Administrative Objects .	ENCPart

Parameter	Description	Accepted Values/Examples
RangeHref (range MQL command)	When configuring a form field in Business Modeler, the RangeHREF parameter is set on the Link tab.  Use to configure a textbox that has a Browse (...) button that calls a chooser or custom window from which users can select a value to populate the textbox. This Owner textbox is an example with a range helper.  In the RangeHref parameter, specify the href URL to display the window. For example, the href might call a custom selection page. Range helpers are used for form pages only in Edit mode. The only control that can be configured with a range helper is textbox. To specify the control type, set Input Type=textbox. To see an example of a field configured with a RangeHref, see Field with Popup Range Helper. Also see Implementing Range Helpers for Choosers or Custom Pages.	You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros. For example: <pre> \${COMMON_DIR}/ emxSelectVault.jsp \${SUITE_DIR}/ emxSelectUser.jsp emxTypeChooser.jsp?type peList=type_Part,type _Document ../common/ emxTypeChooser.jsp?&S electType=multiselect &SelectAbstractTypes= true&InclusionList=eS erviceEngineeringCent ral.Types&observeHidd en=true&ShowIcons=tru e </pre>
Settings	Additional settings that define the behavior and appearance of the column.	Name/value pairs, as defined in the table below.
Type (for web form)	The dynamic user interface components do not use the type parameter.	—
Update URL	The dynamic user interface components do not use this parameter. They use the Update Program and Update Function settings instead.	—

Settings for Web Form Objects

This table lists and describes the settings for web form objects. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
*Registered Suite	The application the field belongs to. The system looks for files related to the field in the registered directory for that application, which is specified in <code>emxSystem.properties</code>. Based on the application name, the system passes the following parameters in the href URL: • <code>suiteKey</code> • <code>emxSuiteDirectory</code> • <code>StringResourceFileId</code>	The value must be set without any spaces, for example, <code>EngineeringCentral</code> or <code>Framework</code>. The value must be set to the suite name as defined in the key <code>eServiceSuites.DisplayedSuites</code> within <code>emxSystem.properties</code>. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in <code>emxSystem.properties</code> is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”.
Access Expression Access Function Access Mask Access Program	All these settings can be used to control access to form fields just as they can be used to control access to other UI components. For details, see Controlling User Access to User Interface Components .	—
Admin Type	Use to translate fields whose values are names of admin objects or ranges of attributes. For example, suppose you are configuring a field that shows an object’s current state and you want the state name to be translated. You would add this setting and set the value to State. The translations for admin object names are stored in the <code>emxFrameworkStringResource.properties</code> files, as described in the chapter on internationalization.	These keywords get the field values translated for the appropriate type name: • Type • State • Role • Relationship • Policy • Group • Vault • Attribute (for translating the attribute name, not the range values) To translate an attribute value and range values, specify the symbolic name of the attribute, which starts with “attribute_” and is followed by the attribute name with no spaces. For example: <code>attribute_UnitOfMeasure</code> <code>attribute_PartClassification</code>
*Required Setting		

Setting	Description	Accepted Values/Examples
Allow Manual Edit	When true, users can manually edit the form row for this field. Applicable only when the range parameter is set to a URL or when the setting format is assigned to date. It is ignored in all other cases. When this setting is true, the Admin Type setting is ignored.	true false (default)
Alternate OID expression	By default, when a field's data is configured to show as a hyperlink using the href parameter, the system passes the ID for the business object the form page applies to. Using this setting, you can have the system pass the ID(s) for a different object, namely, the ID(s) for the object(s) returned from the expression specified in this setting.	<code><to[relationship_NewPartPartRevision].from.id></code> <code><to[relationship_EBOM].from.id></code> To see an example of a field that uses an Alternate OID expression and an Alternate Type expression, see Field Values with Hyperlinked Data Using an Alternate OID and Alternate Type Icon .
Alternate Type expression	When the Show Alternate Icon setting is true, this expression is used to obtain the object type. Based on the obtained type, the corresponding icon is displayed.	<code><to[relationship_NewPartPartRevision].from.type></code> <code><to[relationship_EBOM].from.type></code>
Cols	Used when the input type is set to textarea. This setting limits the length of the textarea on the form and specifies the visible width in average character widths. If not specified, it uses the HTML default, which is 25.	25 40 50
Default	If a field's value is empty or null and this setting is defined, the default value is displayed for the field.	The default value you want to display. This can be a string resource key or the actual characters you want to fill in as the default. The wildcard (*) can be used for search criteria fields. emxFramework.Common.default All *
Editable	Use to indicate whether the field is displayed as editable or read only. Only applies for Edit mode and View mode ignores the setting.	true (default)—Users can edit the field when shown on the Edit mode form. false—Users cannot edit the field when shown on the Edit mode form. The field looks just like it does in View mode except it is never hyperlinked.
Field Column Headers	Used in conjunction with the Field Type=Table Holder setting. Specifies the labels for the column headings. The number of labels should be the same as the value for the Field Table Columns setting and should be separated by a comma. See Fields Arranged in a Table .	Comma-separated list of column heading labels: Min,Max,Avg
*Required Setting		

Setting	Description	Accepted Values/Examples
Field Row Headers	Used in conjunction with the Field Type=Table Holder setting. Specifies the labels for the row headings. The number of labels should be the same as the value for the Field Table Rows setting and should be separated by a comma. See Fields Arranged in a Table .	Comma-separated list of row heading labels: Weight, Volume
Field Size	Determines the width of a textbox input type field. The width is given in pixels except when Input Type is textbox or not set. In that case, its value refers to the (integer) number of characters.	Number of pixels, for example: 30 20 is the default.
Field Table Columns	Used in conjunction with the Field Type=Table Holder setting. Defines the number of columns for the table. See Fields Arranged in a Table .	2, 3, ...
Field Table Rows	Used in conjunction with the Field Type=Table Holder setting. Defines the number of rows in the table. See Fields Arranged in a Table .	1, 2, ...
*Required Setting		

Setting	Description	Accepted Values/Examples
Field Type	This setting is used for several purposes: - To indicate that the field's data should be obtained from a program or image instead of an expression. - When the field data is obtained from an expression, if the data is basic information or an attribute. The system needs to know whether a field's data is basic information or an attribute in order to update the information correctly. Specifying whether the field is basic or an attribute is only required for fields that will be editable. - To indicate the field is a dummy field that defines fields to display in a table or group. For more information on specifying field data, see Defining Form Fields. The field type can be one of the following: - program—The values for this field are obtained from a program (JPO). With this setting, the program and function name are required as settings. - programHTMLOutput—Same as the “program” setting above, except the field value output is in HTML format. Field values are placed in the table cell between <td> and </td> tags. This setting ignores other field settings such as Show Type Icon, href, format, and Alternate OID expression. - image—The field's value is an image. To specify the image file, use the Image setting. - basic—The field displays basic information for the business object. Basic information includes name, type, originated, policy, etc. Specifying basic as the field type is only needed when the field is editable. The only editable basic information is: type, name, revision, current, policy, description, owner, vault. (Continued in column on the right.)	The field type can be one of the following (continued from column on left): - attribute—The field displays values for an attribute on the business object, such as Originator or Weight. - Section Header—Adds a new section heading between the form fields. The setting Section Level determines the heading level. See Field as Section Header and Separator. - Section Separator—Adds white space to separate fields and sections. - Group Holder—Groups the fields under the field in one row. Uses the Group Count setting to determine the number of fields to group. When a field has a Field Type of Group Holder, it serves as a dummy field to define the group and does not appear on the form. - Table Holder—Arranges the fields under the field in columns and rows. Table Holder fields serve as dummy fields to define the fields to display in a table. Also see these settings: Field Column Headers, Field Row Headers, Field Table Columns, Field Table Rows. - emxTable—Embeds a configurable table in the form. Used in conjunction with the table setting and either the inquiry or program setting. See Field that Embeds a Configurable Table in the Form.
*Required Setting		

Setting	Description	Accepted Values/Examples
format	Use to specify the type of data in the field. If the Editable setting is true and the field is on an Edit mode form, the system uses these format values to validate the field value. Validation takes place on the client side, before updating the object displayed in the form. For more information on validation, see Validating Form Field Data. To support the date compare logic, whenever the field format is “date”, an additional hidden parameter is added to the form with the name assigned to the web form field name suffixed by “_msvalue”. If there is a valid display value for the date, the hidden parameter is assigned with the value that is the equivalent of displayed date in milliseconds (calculated from midnight, January 1, 1970). This hidden parameter value can be used by the validation methods to compare two dates. Note: The hidden parameter gets updated only when the out of the box calendar component is used to change the date. So if the field type is changed to manual edit, the hidden parameter may not have the updated milliseconds value when the field is manually changed. In this case the validation method can simply ignore the date compare and must depend on the server side validation. The client side validation can be ignored by checking if the field is readonly.	- date—Uses the tag lib to format the displayed field value based on the browser locale setting. To see an example of a field configured as a date field and that has date validation, see Field Value with Dates. - currency - numeric—Use if the value must be validated as a number before updating the values. Applicable only in Edit mode. - email—Displays the column values as an email address. When a user clicks the email address, the email editor configured in the client is presented.
function	The name of the method to call within the JPO program specified in the program setting. This method within the JPO is used to get the field values if the setting “Field Type” is set to “program” or “programHTMLOutput”. For more information, see Field Values Obtained from Program.	The name of a function in the program JPO, such as: - getAssignedBuyerDesk - getPackageAccess - getParentPart
Group Count	Used in conjunction with the Field Type=Group Holder setting. Specifies the number of fields below the Group Holder field to place in the grouped row. See Fields that are Grouped.	2, 3, 4, ...
Help Marker	Specifies the name of the help marker to call for context-sensitive help. In the href URL called when the field data is clicked, the system passes a parameter called HelpMarker and includes the marker text specified for this setting.	The naming convention for help markers is the page title, as displayed at the top of the visual page, prefixed with “emxhelp”. The marker is all lowercase with no spaces.
Hide Label	Displays or hides the label for a particular row on a form.	True False
*Required Setting		

Setting	Description	Accepted Values/Examples
Image	Use to specify an image file when the field value should be an image only. This setting is required when the Field Type setting is set to image. This file must exist in the application server (not in the database). You can make the image a hyperlink by including a URL in the href parameter. To see an example of a field with an image, see Field Values as Hyperlinked Image.	images/newPart.gif images/EditItem.gif
Input Type	Only used for forms and table columns in edit mode. Specifies the type of HTML control to display for user input. The setting accepts the values listed below and those listed in the cell on the right. Although multiple choices can be displayed in a webform, only one selection can be saved during edit. If you want to save multiple selections, a custom JPO needs to be written which might delimit the choices in the attribute using the ADK. If you want to disable the ability to make multiple selections during view of webform, you need to use programHTMLoutput and the html tag that does that. textbox—This is the default. The field has a single-line box for typing text. This graphic shows a field configured as a Text Box with the Required setting equal to true. Weight 0.0 textarea—The field has a multi-line box for typing text. Description checkbox—The field has a check box next to each range value. For fields that allow multiple selections, use check boxes instead of a list box. Make Task Owner? ☐ See Sample JPO for Web Form with Custom Combobox. See the column on the right for more acceptable values.	listbox—The field has a list of range values. Although users can select more than one item in the list by holding the Ctrl key, it's best to use check boxes for multiple selection fields. radiobutton—There is a radio button next to each range value. To see an example of a field configured with radio buttons, see Field for Attribute with Choices in Radio Buttons. Unit of Measure ☐ LB (pound) ☐ IN (inch) ☐ GA (gallon) ☐ FT (feet) ☒ EA (each) combobox—The field has a drop-down list of options and users can only select one value. Use combo boxes for attributes that have defined ranges and for attributes whose ranges are determined with a Range Helper URL. To see an example of a field with a combo box, see Field for Attribute with Choices in Combo Box. ECR Evaluator TestECREvaluator
inquiry	Used only for fields defined with Field Type=emxTable. Either this setting or the program setting must be used to define the objects to retrieve. Specifies the inquiry admin object that should be used to retrieve the business objects to be included in the table.	Names of inquiry admin object separated by a comma. inquiry=SCSBuyerDesk,SCSBuyerDeskAss igned inquiry=ENCAIIParts, ENCREleasedParts
Maximum Length	Limits the number of characters that can be entered in a field with Input Type of textbox. If not specified, it uses the HTML default (unlimited).	Number of characters, for example: 30
*Required Setting		

Setting	Description	Accepted Values/Examples
Popup Modal	Use to configure whether any window that opens from a link is modal or non-modal. Only used when the setting Target Location is set to popup. If a form Component is opened in a dialog in edit mode then the dialog must open as a modal dialog. For example, if a table column is designed to pop up the Form Component in edit mode, then that column must have a setting Popup Modal=true to open the dialog as modal dialog.	true—the popup window is modal false—the popup window is non-modal If the setting is not specified, the window is modal.
Printer Friendly	Specifies whether the target page should include the Printer Friendly tool. In the href URL called when the field data is clicked, the system passes a parameter called PrinterFriendly and includes the value specified for this setting. If this setting is not included, the value is assumed to be false and the target page does not display the Printer Friendly tool.	True False
program	Use to specify the name of the JPO program to use to get the field's data. This program is used to get the field values when the Field Type setting is set to “program” or “programHTMLOutput”. Also can be used for fields defined with Field Type=emxTable to define the objects in the table. Alternatively, inquiry can be used. <i>Using a JPO to populate field data is recommended only when a select expression cannot be used to obtain the field value.</i> To see an example of a field that uses a program, see Field Values Obtained from Program.	The name of a JPO, such as: - SCSBuyerDeskForm - TMCPackagesForm - ENCPartForm - AEFUtilForm
Range Function	Use to specify the name of the method in the JPO specified in the Range Program setting. See the Range Program setting below for more information.	The name of a function in the Range Program JPO, such as: - getAssignedRange - getClassificationRange - getPartUOM
Range Program	Use to specify the name of a JPO that contains a method to get the field value ranges (choices). A range program is used only when the Input Type setting is combobox, radiobutton, or checkbox. If the choices need to be presented on a page in a popup window, for example in a chooser, use the RangeHref parameter instead. To see an example of a field configured with a range program, see Field for Attribute with Choices in Combo Box. Also see Writing a JPO for Getting Field Range Values (Choices).	The name of a JPO, such as: - SCSBuyerDeskForm - TMCPackagesForm - ENCPartForm - AEFUtilForm
*Required Setting		

Setting	Description	Accepted Values/Examples
Required	Use to indicate the value for this field is required. This setting is applicable for editable fields displayed on the Editable mode form.	true—When completing the form, the user must enter a value for the field. The field label appears in red italic text. If the user doesn't enter a value and clicks Done, a JavaScript message appears that prompts the user to enter a value. false (default)—When completing the Edit form, the user can leave this field blank.
Rows	Used when the input type is set to textarea. This setting limits the height of the textarea on the form and specifies the number of visible text lines. If not specified, it uses the HTML default, which is 5.	5 10 20
Section Level	Used in conjunction with the Field Type: Section Header setting to define the level of heading. To see an example, see Field as Section Header and Separator .	Two heading levels are available: 1 (default)—Font for heading label is large and a horizontal line is included above the label. 2—Font for heading label is smaller and the heading is in the same gray rectangle as standard fields.
Show Alternate Icon	If the field value needs to be displayed with an icon that is different from the current object type icon, set this setting to true. In order to get the right alternate icon, the Alternate Type expression setting must be defined with the expression to obtain the object type.	true false (default)
Show Clear Button	Adds a Clear hyperlink next to the field. This setting only applies to textarea/textbox input type fields. When the Clear link is clicked, it clears the content of the textbox/textarea input type field.	True False
Show Type Icon	If set to true, the field value is displayed along with the type icon for the business object the form page currently displays information for. A type's icon is defined in the emxFramework.smallIcon property in emxSystem.properties. The icon is displayed to the left of the field data. If no property is defined for the type's icon, the system looks for a property defined for the parent type, then grandparent type and so on. If no property is defined for any type in the hierarchy, the system uses the default icon specified in the emxFramework.smallIcon.defaultType property.	true false (default) To see an example of a field with a type icon, see Field Values as Hyperlink and Type Icon .
sortColumnName	Used only for fields defined with Field Type=emxTable. Specifies the column by which the table should be sorted when the page is first loaded. If no column is specified, the rows are listed in the order they are retrieved from the database.	Name of column in the table specified in the table=TABLE_NAME setting.
*Required Setting		

Setting	Description	Accepted Values/Examples
sortDirection	Used only for fields defined with Field Type=emxTable. The direction to sort the rows by.	ascending (default)—Sort a to z or 0 to <i>n</i> . descending—Sort z to a or <i>n</i> to 0.
table	Specifies the name of a table admin object to embed in the form. Used only for fields defined with Field Type=emxTable. See Field that Embeds a Configurable Table in the Form .	Name of table admin object. For example: table=SCSBuyerDesk table=ENCParts
Target Location	Controls where the page specified in the href parameter appears when a user clicks the hyperlinked data.	popup—Page appears in a new window. The window modality can be set with the Popup Modal setting. Popup is the default value and is used if the setting is not included or if the named frame cannot be found. content—The page replaces the content frame. mainFrame—Page appears in the frame that includes the content and menu frames. _top—Page replaces the entire body of the browser window. This value can be set to any valid frame name that is available to the form body frame.
Tip Page	Specifies whether the target page should include the Tip Page tool (light bulb icon). In the href URL called when the field data is clicked, the system passes a parameter called TipPage and includes the URL specified for this setting. If this setting is not included, the value is assumed to be false and the target page does not display the Tip Page tool.	Name of a custom URL page.
Update Function	Specifies a method name in the JPO given in the Update Program setting. See Update Program for more information.	The name of a function in the Update Program JPO, such as: • setAssignedBuyerDesk • setPackage Access • setPartClassification
Update Program	Specifies a JPO that contains a method to set the field value when the field is displayed on an Edit mode form and when the user clicks Done. This program is used only when the Field Type setting is program or programHTMLOutput. <i>Using an update program is recommended only when the Field Type is not attribute or basic.</i> See Writing a JPO for Updating Field Values .	The name of a JPO, such as: • SCSBuyerDeskForm • TMCPackagesForm • ENCPartForm • AEFUtilForm
*Required Setting		

Setting	Description	Accepted Values/Examples
Validate Type	This setting is used to validate any field value with any specific characters defined in emxSystem.properties.	Basic or Restricted—The field values are validated against the value of emxFramework.Javascript.BadChars. Name—The field values are validated against the value of emxFramework.Javascript.NameBadChars.
Window Height	Use to define the height of the popup chooser window or window opened from any href link in a form field.	Number of pixels, such as 700. The default is 600.
Window Width	Use to define the width of the popup chooser window or window opened from any href link in a form field.	Number of pixels, such as 700. The default is 600.
*Required Setting		

URL Parameters Accepted by emxForm.jsp

This table lists the parameters that emxForm.jsp can use. You can add these parameters to the href parameter for the component that calls the form. For example, when you specify the emxForm.jsp to be called from a tree category, you can add these parameters to the href parameter for the menu object.

Parameter	Description	Accepted Input Values
toolbar	Specifies the menu admin object used in the header frame.	Name of menu admin object that represents an action menu.
editLink	Use to have the system include a default Edit toolbar item in the View Form header.	true false (default) emxForm.jsp?form+ENCPart&editLink=true
*form	Specifies the web form admin object used for presenting the form page.	Name of web form admin object. emxForm.jsp?form=ENCPart
formHeader	Use to define the content of the heading that appears at the top of the form page, in the header frame. The value can be either a string resource id or the label itself. The text in the label and string resource ID can contain any valid select expression.	emxForm.jsp?form=ENCPart&header=Properties emxForm.jsp?form=ENCPart&header=Properties:\$<type>\$<name> emxForm.jsp?form=ENCPart&header=emxQuoteCentral.AssignedPackages.AssignedPackages
HelpMarker	Specifies the name of the help marker to call for context-sensitive help. The Help link always displays for form pages.	The naming convention for help markers is the page title, as displayed at the top of the visual page, prefixed with “emxhelp”. The marker is all lowercase with no spaces.
*Required		

Parameter	Description	Accepted Input Values
mode	Specifies whether the form page should be view or edit mode.	view (default)—form is read only edit—form is editable emxForm.jsp?form=ENCPart &mode=edit
objectId	Use to specify the business object for which the properties need to be displayed.	Valid business object ID. emxForm.jsp?objectId=3243.3242.232
PrinterFriendly	Specifies whether the form page should include the Printer Friendly tool.	true (default) false
RegisteredDirectory	The directory where the help files are located.	Directory name within ematrix.
relId	Use to specify the relationship used to get the relationship attribute values for relationship fields.	Valid relationship ID. emxForm.jsp?relId=3243.32424.232
TipPage	Specifies the Web page (html or jsp) that should be launched when the user clicks one of the page toolbar buttons in the form header frame.	Name of a custom URL page. emxForm.jsp?form=ENCPart &TipPage=../myapplication/showMyTipPage.jsp?
*Required		

Sample JPO for Web Form with Custom Combobox

```
import matrix.db.*;
import matrix.util.*;
import java.io.*;
import java.util.*;

public class ${CLASSNAME}
{
    public ${CLASSNAME} (Context context, String[] args) throws Exception
    {
        // ...

    }

    public String getChoicesHTML(Context context, String[] args) throws Exception
    {
        // check String[] args passed in to tell if view or edit?

        System.out.println("Array length = " + args.length);

        HashMap myMap = (HashMap) JPO.unpackArgs(args);
    }
}
```

```

HashMap requestMap = (HashMap) myMap.get("requestMap");

/*
System.out.println("\ntags within HashMap.....: " + requestMap + "\n");
*/

//Edit mode:

// this should return "edit" if "edit" then build combobox else just display value for
view mode
String edit = (String) requestMap.get("mode");

String sReturn = "";

if (edit == null ) // display value for view mode in webform
{
    sReturn ="3";
    System.out.println("mode not edit");
}
else // display list of values in combo box for edit mode
{
    System.out.println("mode IS edit");
    sReturn = "<SELECT name=\"test\"> <OPTION>1 <OPTION>2 <OPTION SELECTED>3 <OPTION>3
<OPTION>4 <OPTION>5 </SELECT>";
}

// Your webform field needs these settings

// Field Type=programHTMLOutput
// function=JPOMethodName
// program=JPOName

// Watch the spelling and case of the above settings!

// To actually save the value selected you would have to add the following settings with
a new save method
// Update Function=JPOMethodName
// Update Porgram=JPOName

// the data to be save will be in the requestMap similar to above

// this returns the String needed to display the custom HTML combo with 3 SELECTED as
default
return sReturn;
}

public static Object getChoices(Context context, String[] args) throws Exception
{

```

```

System.out.println("inside getChoices2 method");

HashMap programMap = (HashMap) JPO.unpackArgs(args);
StringList fieldRangeValues = new StringList();

// Get the required parameter values from "programMap" - as required

String objectId = (String) programMap.get("objectId ");
String relId = (String) programMap.get("relId ");

String requestQueryString = (String) programMap.get("requestQueryString");
String languageStr = (String) programMap.get("languageStr");

fieldRangeValues.addElement("test2_1");
fieldRangeValues.addElement("test2_2");
fieldRangeValues.addElement("test2_3");
fieldRangeValues.addElement("test2_4");
fieldRangeValues.addElement("test2_5");

System.out.println("done adding elements");
return fieldRangeValues;
}

}

```

Using Action Menus with a Configurable Form

Action menus can only be used with the View mode of the form page. You add an action menu to a View form page by passing the action menu name through the URL parameter to `emxForm.jsp`.

The following example URL shows how the action menu is passed in for the View mode form. Because the default mode is View, the mode parameter does not have to be passed.

```

${COMMON_DIR}/
emxForm.jsp?form=BuyerDesk&objectId=3243.32424.232&relId=3432.2
342.2344&header=BuyerDesk&toolbar=SCSBuyerDeskToolbar

```

The parameter `actionMenuName` can be assigned to any valid menu object that is configured as an Action Menu. These menus contain the command objects that represent the toolbar items. This section describes some specific uses of action menus with the configurable form. For instructions on configuring an action menu and toolbar items, see [Building the Menus](#).

Action Menu Link for Editing the Form Using Default Edit Link

The View mode form page supports a default Edit link that displays when the parameter “`editLink=true`” is passed in to `emxForm.jsp`. Here is an example of the default Edit link.

Property: Mechanical Part CM-201010-01 1

Actions ▼	   
Type	Mechanical Part
Name	 CM-201010-01

When a user clicks this link, the same form page opens, with the same web form object name, but in Edit mode.

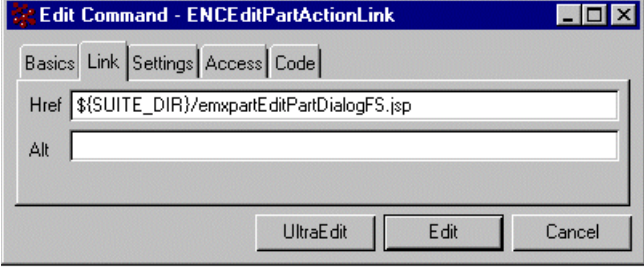
Action Menu Link for Editing a Form using Custom Page and Refresh

You can configure an action menu link so it edits the object displayed in the View mode form page. Upon successful update, it then refreshes the View mode form to reflect the changes.

In order to implement this kind of toolbar item, the action menu link command object must have these settings:

- Row Select=single
- Submit=true
- Target Location=popup

Here is an example of a command configured with these settings.



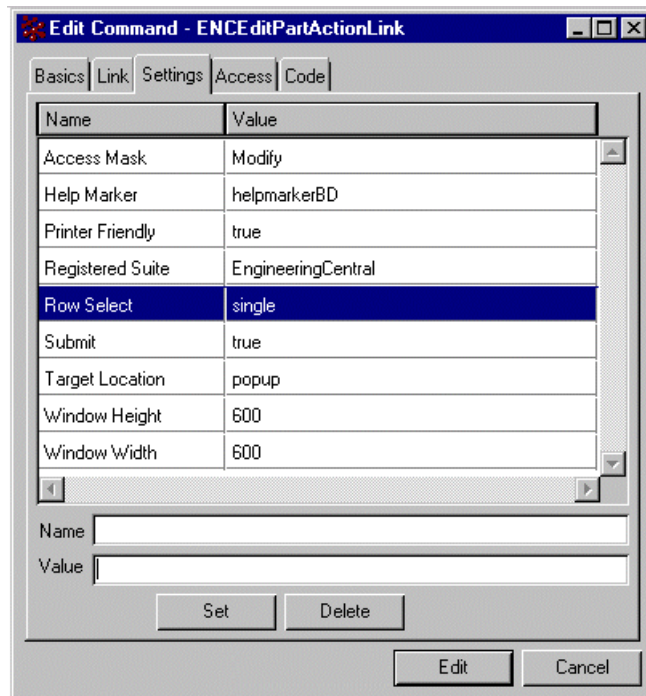
Dialog box: Edit Command - ENCEditPartActionLink

Tabs: Basics | Link | Settings | Access | Code

Href:

Alt:

Buttons: UltraEdit | Edit | Cancel



After editing the selected item, to refresh the View mode form page, an appropriate JavaScript API method should be called, such as:

```
//do Form refresh
top.opener.top.refreshViewFormPage();
top.close();
```

There is one special case when using a modal window to edit an item and then refresh the Form display. If the toolbar item setting “Popup Modal” is set to true, an additional JavaScript method “releaseMouse()” must be called in order to release the mouse. The following JavaScript method shows how to implement the form refresh from the modal window.

```
function doDone() {
//check for page errors here, if there are no errors,

// Submit the data to edit/create the item

//then continue on to do the following:
top.opener.top.modalDialog.releaseMouse();

//do any page redirects/refreshes here - (Form refresh)
top.opener.top.refreshViewFormPage();
top.close();
}
```

If the Cancel button in modal window refreshes the page underneath it, then the above code should also be used. If the Cancel button simply closes the window (top.close()), then it shouldn't be needed.

Decisions and Options Available for Form Fields

This section describes some of the decisions you should make before configuring form fields, lists the main options available for different kinds of fields, and lists the parameters and settings to implement the various options. For descriptions of the parameters and settings referenced in this section, see [Parameters for Web Form Objects](#). For use case examples of the implementing features, see [Defining Form Fields](#).

What is the best mechanism for obtaining the field values?

- Can the values be retrieved using a select expression applied to the current business object or relationship?
If yes, this is the preferred mechanism. Use the Expression parameter and Applies to options for the field.
- Is the value an image file?
If yes, then use the Field Type=image and Image=FILENAME settings.
- Can the values be obtained using a JPO?
If the values can't be obtained using an expression or image file, then a JPO can be used. Use the Field Type=program or Field Type=programHTMLOutput setting and the program and function settings to specify the program and function names.

Who should have access to the field?

Define the users, groups, and roles who can access the field. Form fields have the same access checks as the other UI components. See [Controlling User Access to User Interface Components](#).

Does the field need to be translated?

If yes, the labels should be translated in the StringResource.properties file for the application as described in the chapter on internationalization. If the field's value is the name of an admin object, policy state names, vault name, or ranges for an attribute, use the Admin Type setting.

When the Allow Manual Edit setting is true, admin type is ignored.

What should the field look like when displayed on a View mode form?

- Should it be hyperlinked?
If yes, then enter the URL to call when the field value is clicked in the href parameter. By default, the system passes the object ID for the current object to the called page.
 - Should the hyperlink pass the object ID for an alternate object?
If yes, use the Alternate OID expression setting to specify the expression that gets the object(s) to pass.
 - Where should the href page be displayed?
Use the Target Location setting to specify whether the page displays in a popup window or in a specific frame of the current window. Use Popup Modal to control the modality of the popup window and the Window Height and Window Width settings for the window size.
- Should it have a type icon next to it?
If yes and the icon should reflect the current business object's type, use the Show Type Icon=true setting.
If yes and the icon should reflect the type for an alternate business object, use the Show Alternate Icon=true and Alternate Type expression=EXPRESSION settings.

Should the field be editable when displayed on an Edit mode form?

If yes, add the Editable=true setting or don't specify the setting (it defaults to true). Then see the options listed below for editable fields.

If no, add the Editable=false setting.

What characteristics should the editable field have when displayed on an Edit mode form?

- What type of control should be displayed so users can enter or choose a value?
Use the Input Type setting to specify the type of control, such as textbox, combobox, radiobuttons, etc. The default is textbox.
- If the control is a text box, should it have a Browse (...) button that calls a range helper window?
If yes, use the RangeHref parameter to specify the page to appear in the window. This page can be a common chooser page included with the framework or a custom page. To define the height and width of the window, use the Range Helper Window Height and Range Helper Window Width settings.
- Should the editable field be required?
If yes, add the Required=true setting. The field's label will be red and italic.
- Should the field values be validated?
If yes and you want to use standard Javascript validation, use the format setting and specify the type of data, such as date, real, integer, text.
If yes and you want to use a custom function to validate the data, use the Validate setting.
- Is the field value an attribute or basic?
If yes, use the Field Type=attribute or Field Type=basic setting so the system updates the database according to the type of value.

Defining Form Fields

This section describes how to use the parameters and settings for a web form admin object to define the form's fields. Each field within the web form object is configured to define how the field's values are obtained and how they are displayed. If there is more than one value for a field, the form displays all the values when in View mode. In Edit mode with the setting Editable=true, the field only displays the first value.

Each section lists and briefly describes the specific parameters and settings that implement the field characteristics illustrated in that section. For a complete description of the parameters and settings, see [Parameters for Table Objects](#).

Field Values as Select Expressions

One way to get field data is based on a select expression applied to a businessobject or relationship. The select expression might return one or more values.

For example, to get data for a Current State field, you can use a select expression applied to the current business object. To get the value for the Find Number attribute, which is an attribute on a relationship, you can use a select expression applied to the relationship whose ID is passed to the form page. This graphic shows examples of these two fields on a form in View mode.

Current State	Release
Find Number	55

The fields would look like this when the web form admin object is displayed on a form page in Edit mode.

Current State	Release
Find Number	55

To configure a field so the data is obtained from a select expression, use these parameters and settings

- Expression parameter—Enter the select expression that should be applied to a business object or relationship to get the field values.
- Applies to—Choose what the expression should be applied to: Business Object or Relationship.
- Field Type setting—When the expression is a business object basic or an attribute and the field is editable, you must include the Field Type setting. This lets the system correctly update the basic or attribute property.

The below graphics show how the Current State and Find Number fields shown previously are configured in Business Modeler. In addition to the parameters that get the data using a select expression, the fields have settings that define the application associated with the field and that specify the admin type so the value can be translated.

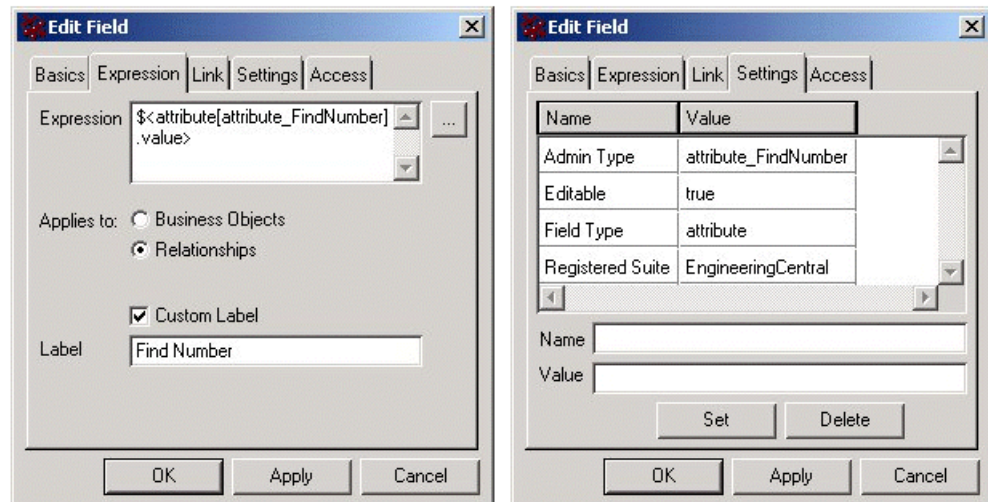
Configuring a field to get data from an expression applied to the current business object

The left screenshot shows the 'Edit Field' dialog box with the 'Expression' tab selected. The 'Expression' field contains 'current'. Under 'Applies to', 'Business Objects' is selected. The 'Label' field contains 'Current State'. The right screenshot shows the 'Edit Field' dialog box with the 'Settings' tab selected. It displays a table with the following data:

Name	Value
Admin Type	State
Editable	false
Registered Suite	EngineeringCentral

Below the table, there are fields for 'Name' and 'Value', and buttons for 'Set', 'Delete', 'OK', 'Apply', and 'Cancel'.

Configuring a field to get data from an expression applied to the current relationship



Field Values Obtained from Program

Field values can also be obtained from a method in a JPO program. The method is defined to obtain the values based on specific business logic.

To get field values using a method in a JPO, use these parameters and settings

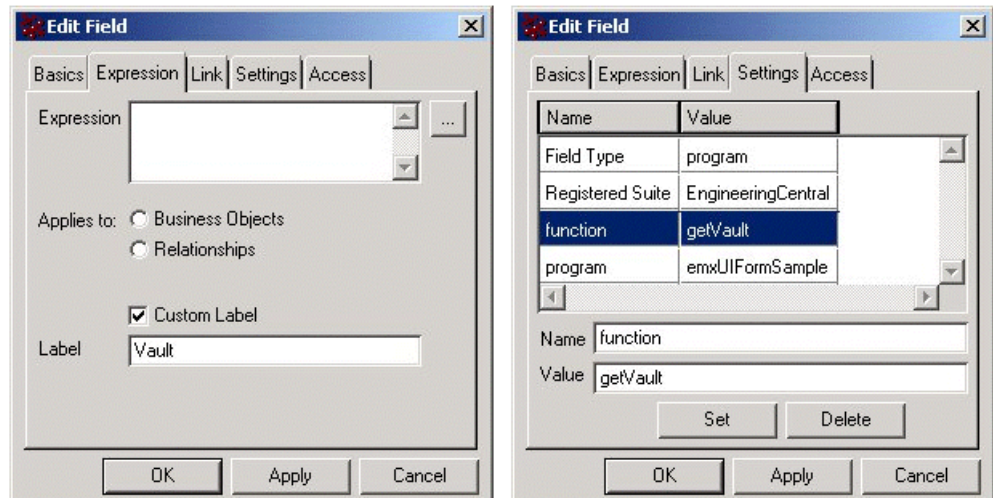
- Expression parameter—Leave blank.
- Field Type setting—Set to program when program results contain only field values. Set to programHTMLOutput when program results contain complete HTML tag (including the field value) to be displayed in the form row.
- program setting—Enter the name of the JPO object to be used for getting the field values.
- function setting—Enter the name of the method in the JPO that should be implemented to return the field values.

To see a sample JPO that includes a method to get field values, see [Sample JPO for Getting Table Column Values](#).

The below graphics show how a Vault field could be configured in Business Modeler to get data from a JPO.

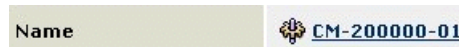
For simplicity, these example screen shots show a field that gets the current vault using a JPO program, even though the vault can be obtained more directly using the select expression "vault".

Configuring a field to get data from a JPO program



Field Values as Hyperlink and Type Icon

Fields can be configured to display data as a hyperlink and with a type icon to the left of the hyperlinked text data. For example, suppose you want a field to display the name of the current business object with the object's type icon displayed to the left of the name. In View mode, the name is hyperlinked. When a user clicks the link, the system passes to the href URL the ID for the business object the page applies to. The ID is passed using the objectId parameter. The View mode for such a field is shown below.



When this field is displayed on a form in Edit mode, the data is not hyperlinked because Edit mode doesn't display hyperlinks. The field would look like this.



To configure a field so the data is hyperlinked when shown on a form in View mode, use these parameters and settings

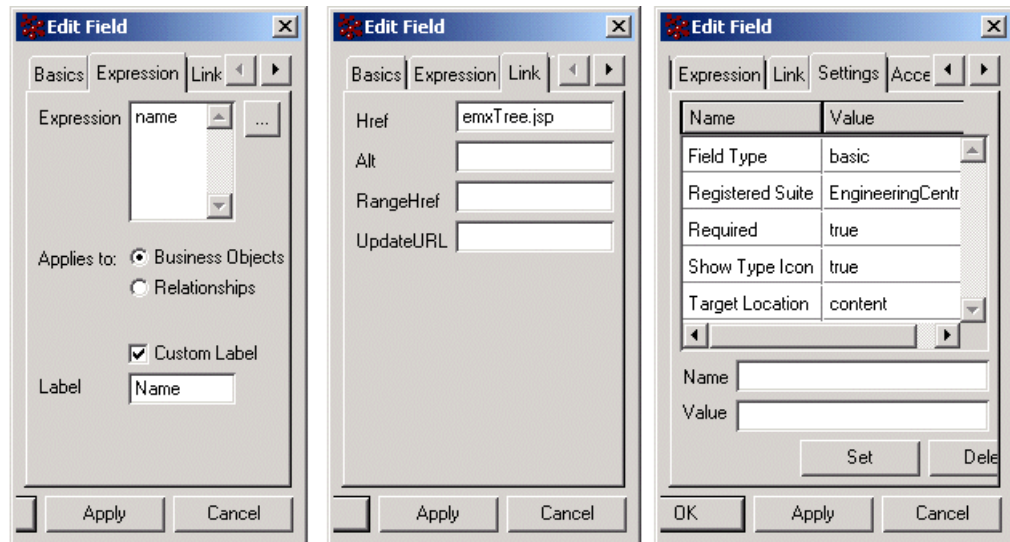
- href parameter—Enter the JSP that should be called when the user clicks the hyperlinked data. The JSP can include parameters accepted by the JSP and path information, including directory macros.
- Target Location setting—Use this setting to specify where the href JSP should be displayed: popup, content, _top, or any valid frame name.

To configure a field to include the type icon for the current object, use these parameters and settings

- Show Type Icon setting—Set this setting to true.

The below graphics show how the Name field shown previously is configured in Business Modeler. In addition to the parameters and settings that make the data hyperlinked and that add the type icon, the field has settings that make the field required when the form is in Edit mode.

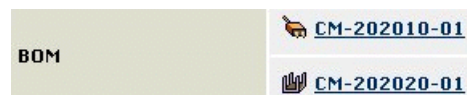
Configuring a field with hyperlinked data and type icon



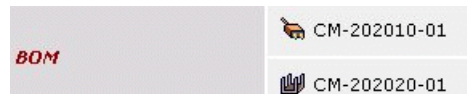
Field Values with Hyperlinked Data Using an Alternate OID and Alternate Type Icon

When a user clicks a hyperlinked field, the field passes an object ID to the href URL using the objectId parameter. By default, ID is for the object the page applies to. Another way to configure a hyperlinked field is to have the field pass the ID for a different business object. The alternate object IDs are obtained from the expression assigned to the setting Alternate OID expression. The href parameter and therefore this setting are applicable only for the View mode form and the Edit mode form ignores it. You can also configure fields to show a type icon other than the type icon for the object the form page applies to.

This graphic shows a field in View mode that is configured to show hyperlinked data with an alternate OID and an alternate type icon.



This graphic shows the same field on a form in Edit mode.



To configure a hyperlinked field to pass an alternate object ID, use these settings and parameters

- href parameter—Enter the JSP that should be called when the user clicks the hyperlinked data. The JSP can include parameters accepted by the JSP and path information, including directory macros.
- Alternative OID expression—Enter a valid select expression that returns one or more business objects. The system passes the IDs for the objects in the href using the objectId parameter.
- Target Location setting—Use this setting to specify where the href JSP should be displayed: popup, content, _top, or any valid frame name.

To configure a field to display a type icon for an object different from the object the form applies to, use these parameters and settings

- Show Alternate Icon—Set to true.
- Alternate Type expression—Enter a valid select expression that returns a business object. The system displays the type icon associated with this business object.

The below graphics show how the BOM field shown previously is configured in Business Modeler. The Link tab, not shown, would have emxTree.jsp for the href parameter.

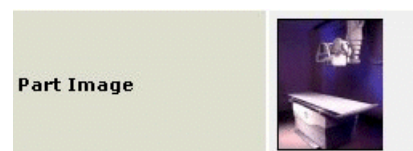
Configuring a hyperlinked field to pass alternative OID and alternate type icon

The left screenshot shows the 'Edit Field' dialog box with the 'Expression' tab selected. The 'Expression' field contains '\$<from[relationship_EBOM].to.name>'. The 'Applies to' section has 'Business Objects' selected. The 'Custom Label' checkbox is checked, and the 'Label' field contains 'BOM'.

The right screenshot shows the 'Edit Field' dialog box with the 'Settings' tab selected. The 'Name' field is empty, and the 'Value' field is empty. The 'Alternate OID expression' field contains '\$<from[relationship_EBOM].to.id>'. The 'Alternate Type expression' field contains '\$<from[relationship_EBOM].to.type>'. The 'Editable' field is set to 'false'. The 'Registered Suite' field is set to 'EngineeringCentral'. The 'Show Alternate Icon' field is set to 'true'. The 'Target Location' field is set to 'popup'.

Field Values as Hyperlinked Image

A field can be configured to display an image only and this image can be hyperlinked when displayed on a form in View mode. This graphic shows an example of this type of field.

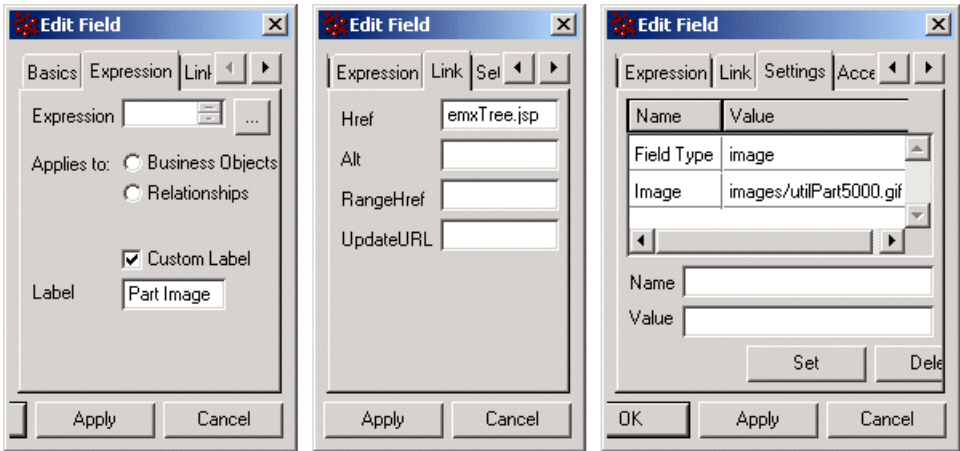


To configure a field as an image that is hyperlinked in View mode, use these parameters and settings

- Field Type setting—Set this setting to image.
- Image setting—Specify the image file name. This file must exist in the application server (not in the database).
- href parameter—Enter the JSP that should be called when the user clicks the hyperlinked image. The JSP can include parameters accepted by the JSP and path information, including directory macros.
- Target Location setting—Use this setting to specify where the href JSP should be displayed: popup, content, _top, or any valid frame name.

The below graphics show how the Part Image field shown previously is configured in Business Modeler.

**Configuring a
hyperlinked image**



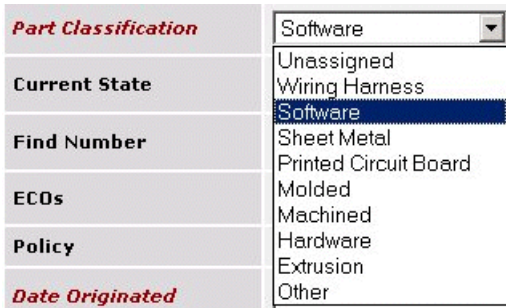
Field for Attribute with Choices in Combo Box

A form field can be configured to display an attribute (for a business object or relationship) that has choices (ranges). Users can select one of the available choices using a combo box when the field is shown on a form in Edit mode.

For example, this graphic shows a field for a Part Classification attribute that has been configured to show the choices in a combo box.



This graphic shows the same attribute with the combo box expanded to show all the ranges. The range Software is currently selected.



When in View mode, the currently selected attribute value is shown as read only text, as with this Part Classification attribute.



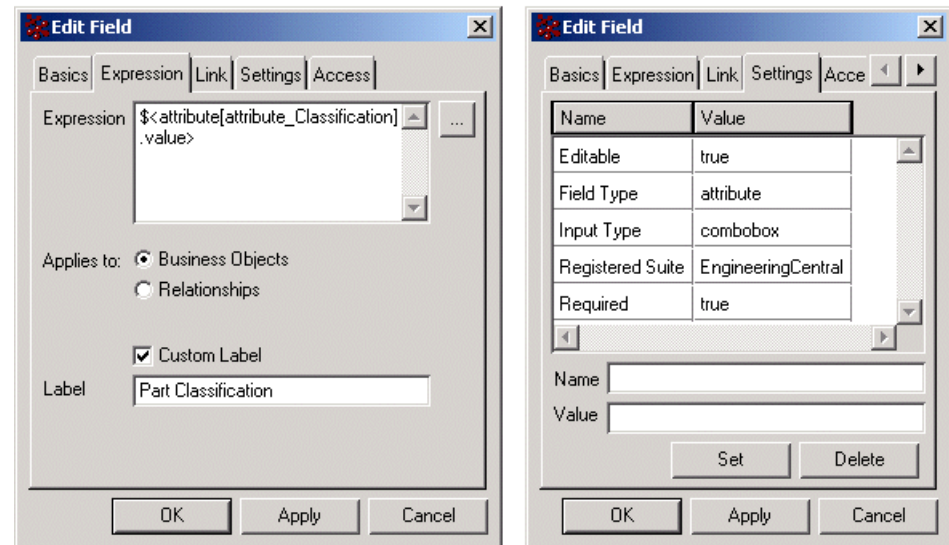
There are two ways to configure this kind of field depending on how you get the choices that should populate the combo box. One method is for when the choices are defined as ranges for the attribute in Business Modeler, as with the Part Classification attribute, and the other is for when the ranges must be obtained from a JPO program. Both methods are described below.

To configure a field to display attribute ranges in a combo box when the ranges are obtained from the attribute admin object, use these parameters and settings

- Input Type setting—Set to combobox.
- Field Type setting—Set to attribute. This tells the system to obtain the ranges from the admin object.
- Editable setting—Set to true (although it is true by default so you don't have to add it explicitly).

The below graphics show how the Part Classification field shown previously is configured in Business Modeler.

Configuring a field to display attribute ranges in combo box, with ranges obtained from attribute admin object



For some attributes, it may not be practical to define the ranges in the admin object, as with the Originator field shown below. In this case, the field can be configured to obtain the data from a JPO.



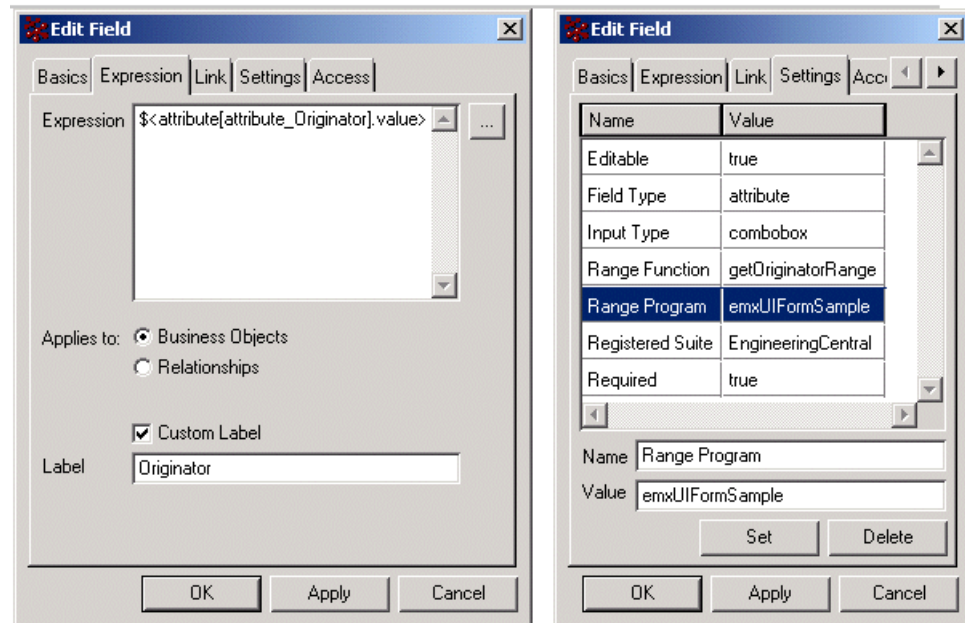
To configure a field to display attribute ranges in a combo box when the ranges are obtained from a program, use these parameters and settings

- Input Type setting—Set to combobox.
- Range Program setting—Enter the JPO name that has the method to get the range values.
- Range Function setting—Enter the JPO method name to get the range values. The method should return the choices in the object type StringList.

For details of how to write the JPO for fetching the field choices, see [Writing a JPO for Getting Field Range Values \(Choices\)](#).

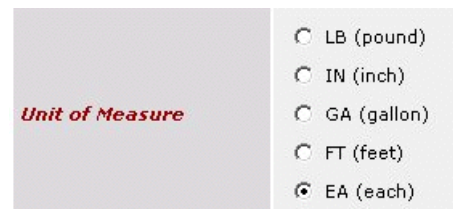
The below graphics show how the Originator field shown previously is configured in Business Modeler.

Configuring a field to display attribute ranges in combo box, with ranges obtained from program

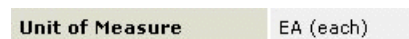


Field for Attribute with Choices in Radio Buttons

Field can be configured to display an attribute (for a businessobject or relationship) that has choices defined for it in a radio button control. The radio button controls are only shown when the field is displayed on the form in Edit mode. For example, this Unit of Measure field has been configured with radio buttons.



When the field is displayed in View mode, only the currently-selected choice is displayed.

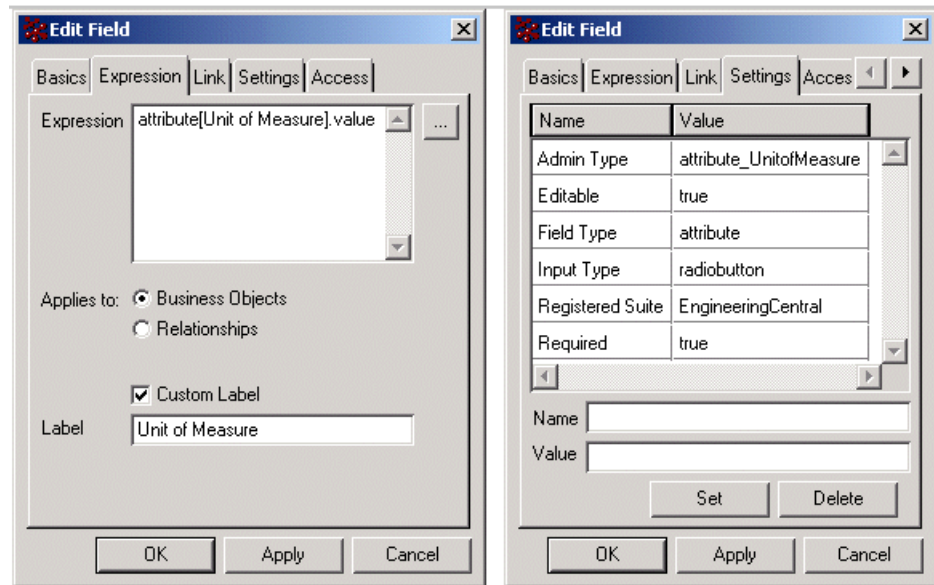


To configure a field to show attribute ranges as radio buttons, use these parameters and settings

- Input Type setting—Set to radiobutton.
- Field Type setting—Set to attribute. This setting and value causes the system to get the values from the database so the choices must be defined as ranges for the attribute in Business Modeler.

The below graphics show how the Originator field shown previously is configured in Business Modeler.

Configuring a field to display attribute ranges as radio buttons, with ranges obtained from program



Field Value with Dates

Fields can be configured to display the field value in date format. The value must be a valid date string. The date value is displayed based on the system and browser setting using the “!zDate” taglib. Here is an example of a date field in View mode.

Date Originated Jul 9, 2002 10:04:40 PM

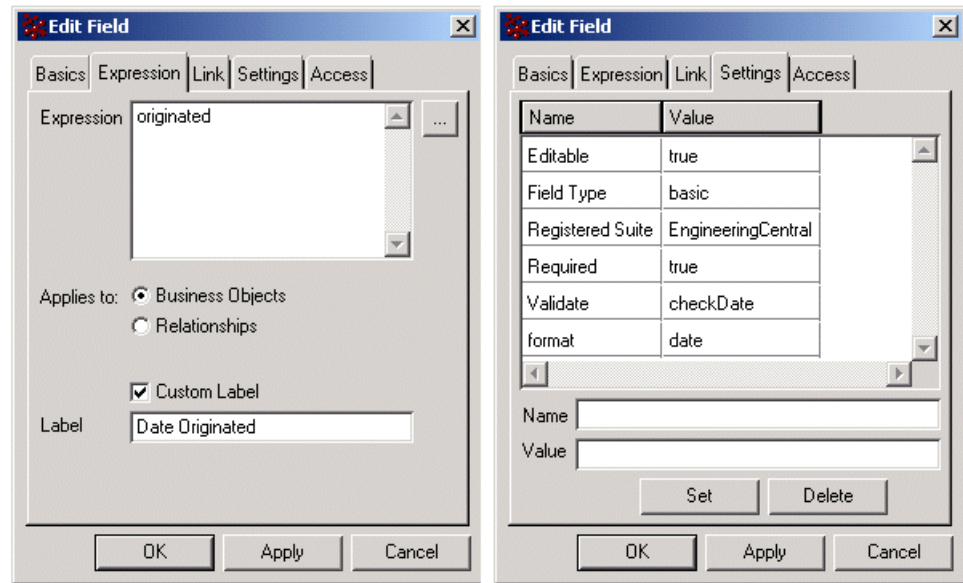
This is the same date field in Edit mode. The field includes the Calendar icon which users click to select a date from a Calendar popup.

Date Originated Jul 9, 2002 10:04:40 PM 

To configure a field to display a date and allow date input with the Calendar popup, use these parameters and settings

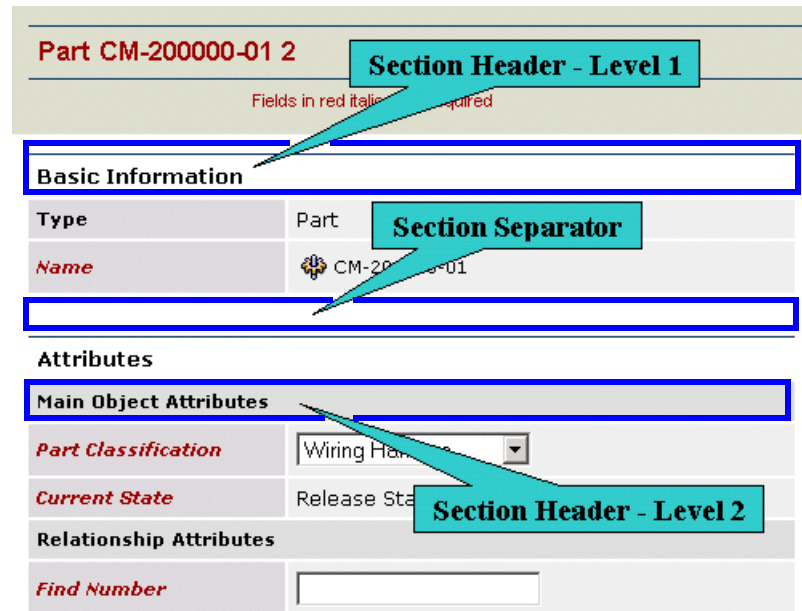
- format setting—Set to date.
- Editable setting—Set to true.

Configuring a field to display a date



Field as Section Header and Separator

Fields can be configured as section headers and separators (white space) to group similar kinds of attributes and information. There are two levels of section headers. Level 1 includes a line above the heading label.



To configure a field to display a section header, use these parameters and settings

- Field Type = Section Header
- Section Level = 1 or 2
- label=<Header Label>

Configuring a field to display a section header

The 'Edit Field' dialog box has tabs for Basics, Expression, Link, Settings, and Access. The Basics tab is active. It contains a table with the following data:

Name	Value
Field Type	Section Header
Registered Suite	EngineeringCentral
Section Level	1

Below the table are input fields for Name and Value, and buttons for Set, Delete, OK, Apply, and Cancel.

To configure a field to display a section separator, use these parameters and settings

- Field Type = Section Separator

Configuring a field to display a section separator

The 'Edit Field' dialog box has tabs for Basics, Expression, Link, Settings, and Access. The Basics tab is active. It contains a table with the following data:

Name	Value
Field Type	Section Separator
Registered Suite	EngineeringCentral

Below the table are input fields for Name and Value, and buttons for Set, Delete, OK, Apply, and Cancel.

Fields that are Grouped

A form can be configured to display a group of fields in one row. To group fields, you add to the web form a dummy field above the fields you want to group. The dummy field is defined using Field Type=Group Holder and does not actually appear on the form. Using the Group Count setting, you specify the number of fields to group. For example, if you specify Group Count=3, the system groups the three fields below the Group Holder field.

Part CM-200000-01

Actions ▼ [Icons]

Basic Information

Type	Part	Name	CM-200000-01	Revision	3
Description	Copy Machine				
Owner	Test Everything	Originator	Test DesignEngineer		
Originated	Jan 24, 2003 11:58:18 PM	Modified	Jan 27, 2003 6:28:08 PM		

Three fields grouped by Group Holder field

To configure a form to group multiple fields in one row, use these settings

- Field Type = Group Holder
- Group Count = 1, 2, 3, ...

Configuring a field to server as a group holder

Edit Field

Basics | Expression | Link | Settings | Access

Name	Value
Field Type	Group Holder
Group Count	3

Name

Value

Set Delete

OK Apply Cancel

Fields Arranged in a Table

A form can be configured to display fields in a tabular format. To arrange fields in a table, you add to the web form a dummy field above the fields you want to include in the table. The dummy field is defined using Field Type=Table Holder and does not actually appear on the form. Using the Field Table Columns and Field Table Rows settings, you specify the number of columns and rows and their headings.

The system uses the first fields under the Table Holder field to populate the table. For example, if the settings specify 3 columns and 2 rows, the table contains 6 cells. Therefore, the system uses the first 6 fields under the Table Holder to populate the cells, working from upper left to lower right (the first field under the Table Holder is placed in the upper left cell, the second field is in the upper middle cell, and so on). After adding the Table Holder field, make sure the fields under it in the web form are the ones you want in the table and are in the correct order. The fields in a table do not need labels defined for them because the column and row headings server as labels.

Attributes

Main Attributes	
Part Classification	Unassigned
Current State	Approved

Table Holder defined with 3 columns and 2 rows

Column Header

Cost and Weight Info

	Estimated	Target	Lead Time
Cost	0.0	0.0	Unassigned
Code	Make	Buy	Plastic

Row Header

To configure a form to arrange fields in columns and rows, define a Table Holder field using these settings

- Field Type = Table Holder
- Field Column Headers = Label for each column header, separated by a comma
- Field Row Headers = Label for each row header, separated by a comma
- Field Table Columns = 2, 3, ...
- Field Table Rows = 1, 2, 3, ...

Configuring a field to server as a table holder

The 'Edit Field' dialog box is shown with the 'Settings' tab selected. It contains a table with the following data:

Name	Value
Field Column Headers	Estimated,Target,Lead Time
Field Row Headers	Cost,Code
Field Table Columns	3
Field Table Rows	2
Field Type	Table Holder
Registered Suite	EngineeringCentral

Below the table, there are input fields for 'Name' and 'Value', and buttons for 'Set', 'Delete', 'OK', 'Apply', and 'Cancel'.

Make sure the fields under the Table Holder in the web form are the ones you want in the table

Configuring a web form to display fields in a tabular format

Section Header

Table Holder field

6 fields to display in table
(no labels needed because column and row headings serve as labels)

Field that Embeds a Configurable Table in the Form

A form can be configured to display a configurable table. To display a configurable table in a web form, add a dummy field to define the table using Field Type=emxTable.

Bill of Materials					
F/N	Name	Revision	State		
2	CM-201020-01	1	Approved		
4	CM-201040-01	1	Approved		
5	CM-201050-01	1	Approved		
1	CM-201010-01	1	Approved		
3	CM-201030-01	3	Approved		

emxTable Field

To configure a form to display a configurable, use these settings

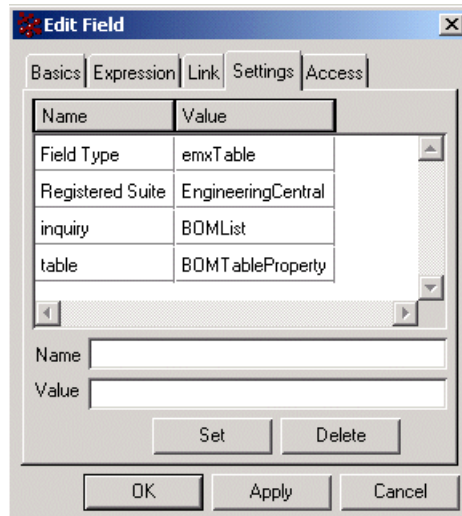
- Field Type = emxTable
- table = <name of the admin table>
- sortColumnName = <name of the column in table>
- sortDirection = <ascending or descending>

One of the following to get the list of objects:

- inquiry = <admin inquiry name>

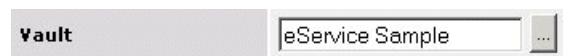
Or:

- program = <JPO program name:JPO method name>

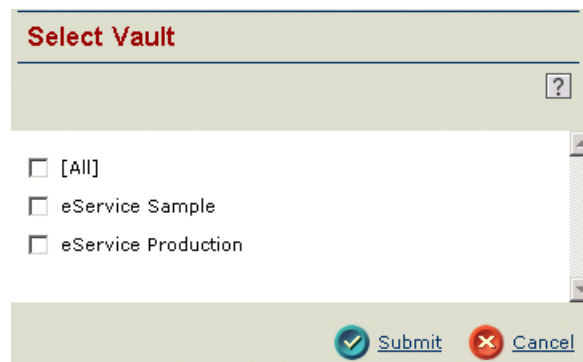


Field with Popup Range Helper

Form fields or table columns can be configured so the Edit mode field displays the field value in a text box. Users select the range value from a user defined/custom popup window or from one of the common chooser components available in the framework, such as the Type Chooser. This Vault field is an example of a field with a popup range helper.



Users access the popup window using the Browse (...) button to the right of the text box. For example, the Vault field can be configured to display a Select Vault page as shown below.

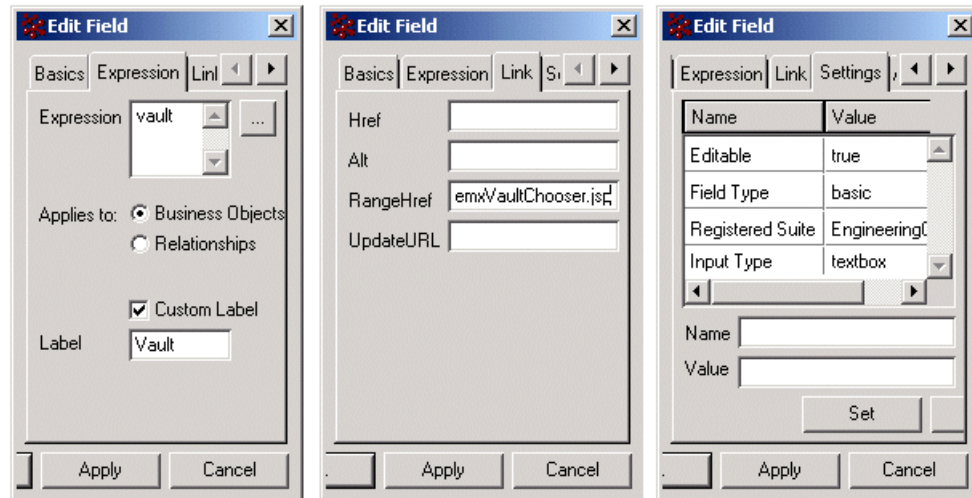


To configure a field so, when in Edit mode, users can choose the value from a page in a popup window, use these parameters and settings

- **RangeHref parameter**—Enter the page that should appear in the popup window plus any parameters needed by the page. This page can either be a common chooser page included with the framework, such as the Vault chooser, or a custom selection page. The page must accept two parameters that the Edit mode form page appends to the URL automatically. For details on implementing the Range Helper page, see [Implementing Range Helpers for Choosers or Custom Pages](#).
- **Input Type setting**—Set to textbox. Since textbox is the default input type, you do not have to specify the setting if you want a text box.

Configuring a field so users can select a range value from a popup window

- Editable setting—Set to true.



Implementing Range Helpers for Choosers or Custom Pages

Fields can be configured so when the form or table is in Edit mode, users can click a Browse (...) next to the field and then choose a range value from a page in a chooser or custom window. This page is specified in the RangeHref parameter for the field. See [Field with Popup Range Helper](#) for details on configuring this type of field. The page can be a common chooser page installed with the framework or can be a custom page. Implementation details for both types of range helpers are described below.

Using a Common Framework Chooser as a Range Helper Page

The framework includes chooser pages, which can be used as range helper pages and specified as the RangeHref for a field. For example, the Type Chooser component can be used by entering the following in the RangeHref parameter:

```
${COMMON_DIR}/
emxTypeChooser.jsp?fieldNameActual=PartType&fieldNameDisplay=PartTypeDisplay&formName=PartList&SelectType=multiselect&InclusionList=eServiceEngineeringCentral.Types&ObserveHidden=False&SuiteKey=eServiceSuiteEngineeringCentral&ShowIcons=False
```

For a current list of the common chooser pages, see [Standard Commands and Pages](#) in Chapter 8.

The common chooser pages require the following two parameters to be passed in. The Edit mode of the form or table page automatically appends these two parameters to the RangeHref, so you do not need to add these parameters to the RangeHref parameter.

- `fieldNameDisplay`—the form or table element name to display the field display value (translated).
- `fieldNameActual`—the hidden form or table element to store actual field value (to be stored in database).

The chooser pages update the Edit form field when the user selects a new value.

Using a Custom, User-Defined Page as a Range Helper

To specify a custom select or chooser page as the range helper page, enter the page name plus any needed parameters in the RangeHref parameter for the field. For example:

```
customRangeHelper.jsp?parameter1=value1
```

The custom JSP page must handle these two parameters:

- **fieldNameDisplay**—the form or table element name to display the field display value (translated).
- **fieldNameActual**—the hidden form or table element to store actual field value (to be stored in database).

The Edit mode of the form or table page automatically appends these two parameters to the URL. So if the RangeHref parameter contains

customRangeHelper.jsp?parameter1=value1, the resulting URL would be:

```
customRangeHelper.jsp?parameter1=value1&
fieldNameDisplay=fieldDisplay&fieldNameActual=fieldName
```

When the user selects a new entry, the custom JSP must set these two elements. Use the following JavaScript to update the Edit form field:

```
var setString =
("top.opener.document.frames[1].document.forms[0]['"+formFieldName+
"'].value='"+newValue+"';");
eval(setString);
```

Validating Form Field Data

You can configure form fields so, when in Edit mode, data is validated on the client side before the data is submitted to the server where the changes are committed. For every form field, you can use either standard JavaScript validation or custom (user-defined) JavaScript validation. Both methods are described below.

Standard JavaScript Validation

Using the field setting “format”, you can have field data validated when it is displayed in Edit mode and the Editable setting is true. For example, if a field has the setting “format=integer”, when the user clicks Done on the Edit Form page, this field's value is validated to make sure it is an integer value. The valid values for the format setting are:

- date
- real
- integer
- alpha
- numeric

The field value is also validated when the field includes the setting “Required=true”. An appropriate JavaScript message appears in case of invalid entries.

User Defined / Custom JavaScript Validation

You can use a custom JavaScript method to validate any field when it is displayed in Edit mode and the Editable setting is true. The method name is configured for the field with the setting “Validate=methodName”. The method can refer to the control with the keyword “this”.

For example, for a field with the setting “Validate=checkLength”, a JavaScript function called checkLenth() is defined as below. The function must return true for successful validation.

```
// JavaScript validation method
function checkLength()
{
    if (this.value.length < 20)
        alert("Invalid Field Length..");

    return (this.value.length > 20);
}
```

The above function must be added to any JavaScript file or any JSP file and the name of the file must be specified in emxSystem.properties file with the key SUITEKEY.UIForm.ValidationFile. One or more files separated by a comma can be assigned as the property value.

AEF Validation File

The AEF validation file is defined using the following property:

```
eServiceSuiteFramework.UIForm.ValidationFile = scripts/
emxUIFormValidation.js
```

Or for backward compatibility:

```
emxFramework.UIForm.ValidationFile = scripts/
emxUIFormValidation.js
```

emxUIFormValidation.js is common and is available to all the web form pages. It is distributed with the AEF and gets overwritten during the AEF upgrade. Currently, the file is empty and serves as a placeholder for future validation methods that may be needed for all applications. Do not change this file since it will get overwritten in future upgrades.

To add any common custom files to be available across all web forms used by different applications, add the custom files to this property. For example:

```
eServiceSuiteFramework.UIForm.ValidationFile =scripts/
emxUIFormValidation.js, \
scripts/abcCustomValidation1.js
```

In this case scripts/abcCustomValidation1.js will not get overwritten by AEF upgrades.

Application-Specific Validation File

Application specific custom validation files are defined using the property based on the Registered Suite (suite key) in emxSystem.properties. For example, if the Registered Suite is “EngineeringCentral”, the corresponding key is defined in the system property file as shown below:

```
EngineeringCentral.UIForm.ValidationFile = scripts/
emxEngrValidation.js
```

Or:

```
eServiceSuiteEngineeringCentral.UIForm.ValidationFile =  
scripts/emxEngrValidation.js
```

One or more files separated by a comma can be assigned as the property value to define multiple files.

Committing Changes Made on Edit Form

Clicking Done on the Edit mode form page commits all changes to the database. Clicking Cancel cancels all changes.

When Done is clicked, the form page executes these processes in the following sequence.

1. All the field values are validated for the format specified in the field setting “format”. The presence of values in fields with the setting Required=true is also checked. For more information, see [Validating Form Field Data](#).
2. All the field values are validated using any user defined/custom JavaScript methods configured in the field setting “Validate”. For more information, see [Validating Form Field Data](#).
3. The form is submitted for processing and checked to see if there are any changes made to the data.
4. If there are changes, the fields are updated by grouping them as business object basics, attributes, relationship attributes and program fields.

Preventing Multiple Submits

In the Edit mode of the form page, clicking Done submits the form to process the data. When the user clicks Done multiple times, multiple submissions are not sent.

Refreshing the View Mode Form after Editing Object Properties

After the Edit mode of the form page commits the changes to the database successfully, the opener frame (typically it would be the View mode of the form page) is refreshed to display the updated data.

It is important to note that when the object’s vault is changed, the object Id changes and therefore the object Id on the View mode of the form page is updated automatically to refresh the data.

When the form is updated while the context tree is present, and the change caused an object id change, then the tree is updated with the new object id. The form component automatically senses the change in object id and updates the context tree as required.

JPO Interface for Form Fields

When you cannot accomplish a task or business requirement using a select expression or other feature supported by the configurable form component, you can use a JPO. The configurable form supports JPOs in three areas:

- Getting the field values to be displayed in a form field.
- Getting the range values (choices) to be populated for a form row.
- Updating field values after updating the form.

To use a JPO, you must specify parameters and settings in the field of the web form admin object as described in [Parameters for Table Objects](#). The JPOs must follow the specific signatures described in the following section.

To see a sample JPO, go to [Sample JPO for Getting Table Column Values](#).

JPO Method Signature and Input Argument

The JPO class must have the mxMain method implemented. The JPO methods are defined with the following signature.

```
public static Object methodName(matrix.db.Context context,
String[] args)
throws Exception
{
}
}
```

The input arguments are:

- Context—passed in always.
- String[] args—will have one element of type HashMap.

The input argument “arg” is a HashMap, which contains the details of the object to be processed. The data structure of this input parameter (HashMap) is defined below:

Key Name	Data Type	This key is assigned to the:
languageStr	String	Language string for the current browser setting. The program can use this key for any internationalization purpose.
New Value	String	Applicable only for use with Update Program and Update Function. Key “New Value” is assigned to the new value changed by the user.
objectId	String	Business object Id (OID) to be used in the current form page.
Old Value	String	Applicable only for use with Update Program and Update Function. Key “Old Value” is assigned to the actual value for the field before the user changed it.
relId	String	Relationship id (RelID) to be used for the form page.
requestMap	HashMap	Key “requestMap” is assigned to a HashMap, which contains a set of key/value pairs as available in the request object. The keys are of type String (parameter names) and the values are of type String (parameter value).

The JPO method can extract the information from the input argument for processing the data.

```

public static Object methodName(matrix.db.Context context,
String[] args) throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    HashMap paramMap = (HashMap) programMap.get("paramMap");

    String objectId = (String) requestMap.get("objectId");
    String relId = (String) paramMap.get("relId");
    String languageStr = (String) requestMap.get("languageStr");
    ...

    return object
}

```

The method returns an object of appropriate type depending on the where it is used. The details of return types are explained in the following sections.

Writing a JPO for Getting Field Values

When a JPO is used to get field values, the form field must be configured with these settings:

- Field Type=program
- program=JPO name
- function=JPO method name

For processing the data, the input parameter can be used by the method. The return type of this method is a `StringList`, which contains one or more field value to be displayed.

Here is a template for this kind of JPO method:

```

public static Object methodName(Context context, String[] args)
throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    HashMap paramMap = (HashMap) programMap.get("paramMap");

    String objectId = (String) requestMap.get("objectId");
    String relId = (String) paramMap.get("relId");
    String languageStr = (String) requestMap.get("languageStr");

    // initialize the return variable
    StringList fieldValues = new StringList();

    // define and add selects if required

    // Process the information to obtain the values
    if ( objectId != null)
    {

```

```

        //add the field values
        fieldValues.addElement(...);

    }

    return fieldValues;
}

```

Writing a JPO for Getting Field Range Values (Choices)

When a JPO is used to get field value ranges, the form field must be configured with these settings:

- Input Type=combobox / radiobutton / checkbox
- Range Program=JPO name
- Range Function=JPO method name

For processing the data, the input parameter can be used by the method. The return type of this method is a `StringList`, which contains one or more field range values (choices) to be displayed.

Here is a template for this kind of JPO function:

```

public static Object methodName(Context context, String[] args)
throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    HashMap paramMap = (HashMap) programMap.get("paramMap");

    String objectId = (String) requestMap.get("objectId");
    String relId = (String) paramMap.get("relId");
    String languageStr = (String) requestMap.get("languageStr");

    // initialize the return variable
    StringList fieldRangeValues = new StringList();

    // define and add selects if required

    // Process the information to obtain the range values and add
    them to fieldRangeValues

    fieldRangeValues.addElement(...);

    ...

    return fieldRangeValues;
}

```

Writing a JPO for Updating Field Values

When a JPO is used to update field values, the form field must be configured with these settings:

- Input Type=comboBox / radioButton / checkBox
- Range Program=JPO name
- Range Function=JPO method name

For processing the data, the input parameter can be used by the method. The return type of this method is a Boolean, which is true if the update is successful. Otherwise it is false.

Here is a template for this kind of JPO function:

```
public static Object methodName(Context context, String[] args)
throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    HashMap paramMap = (HashMap) programMap.get("paramMap");

    String objectId = (String) requestMap.get("objectId");
    String relId = (String) paramMap.get("relId");
    String languageStr = (String) requestMap.get("languageStr");

    // initialize the return variable
    Boolean bUpdate = false;

    // Define and add selects if required

    // Process the information to set the field values for the
    current object
    ....

    // On successful update initialize the Boolean object bUpdate
    bUpdate = true;

    return bUpdate;
}
```

Sample JPO for Getting Field Values

The following is a sample Java Program Object (JPO) for getting the current vault for the object the form page currently pertains to. This is the kind of JPO you can use when the Field Type setting for a form field is set to program or programHTMLOutput. The method “getVault” in this program processes the object and returns the vault value.

The vault name can be obtained by configuring the form field as a businessobject select expression “vault” instead of defining the Field Type as “program”. The “vault” example is used for simplicity and to illustrate the steps involved in writing the JPO for a form field.

```

/*
 * emxUIFormSample
 *
 * Copyright (c) 1992-2003 MatrixOne, Inc.
 *
 * All Rights Reserved.
 * This program contains proprietary and trade secret
information of
 * MatrixOne, Inc. Copyright notice is precautionary only and
does
 * not evidence any actual or intended publication of such
program.
 *
 * static const char RCSID[] = $Id: Exp $
 */

import matrix.db.*;
import matrix.util.*;
import java.io.*;
import java.util.*;
import com.matrixone.framework.beans.*;
import com.matrixone.framework.util.*;
import com.matrixone.framework.ui.*;

/**
 * @version AEF 9.5.0.0 - Copyright (c) 2002, MatrixOne, Inc.
 */
public class ${CLASSNAME}
{

    /**
     *
     * @param context the eMatrix <code>Context</code> object
     * @param args holds no arguments
     * @throws Exception if the operation fails
     * @since AEF 9.5.0.0
     * @grade 0
     */
    public ${CLASSNAME} (Context context, String[] args)
        throws Exception
    {
        if (!context.isConnected())
            throw new Exception("not supported on desktop
client");
    }

    /**
     * This method is executed if a specific method is not
specified.
     *

```

```

    * @param context the eMatrix <code>Context</code> object
    * @param args holds no arguments
    * @returns nothing
    * @throws Exception if the operation fails
    * @since AEF 9.5.0.0
    */
    public int mxMain(Context context, String[] args)
        throws Exception
    {
        if (!context.isConnected())
            throw new Exception("not supported on desktop
client");
        return 0;
    }

    /**
    * get Vault for the object.
    *
    * @param context the eMatrix <code>Context</code> object
    * @param args holds the following input arguments:
    *      0 - HashMap programMap
    * @returns StringList containing Vault name
    * @throws Exception if the operation fails
    * @since AEF 9.5.0.0
    */
    public static Object getVault(Context context, String[]
args)
        throws Exception
    {
        HashMap programMap = (HashMap) JPO.unpackArgs(args);

        String objectId = (String)programMap.get("objectId");
        StringList vaultList = new StringList();
        StringList listSelect = new StringList(1);
        listSelect.addElement("vault");

        if ( objectId != null)
        {

            BusinessObject bo = new BusinessObject(objectId);
            bo.open(context);
            String vault = bo.getVault();
            bo.close(context);
            vaultList.addElement(vault);
        }

        return vaultList;
    }

```



```

/**
 * set Vault for the objects.
 *
 * @param context the eMatrix <code>Context</code> object
 * @param args holds the following input arguments:
 *      0 - HashMap programMap
 * @returns vector of Vault names
 * @throws Exception if the operation fails
 * @since AEF 9.5.0.0
 */
public static int setVault(Context context, String[] args)
    throws Exception
{
    HashMap requestMap = (HashMap) programMap.get("requestMap");
    String objectId = (String) requestMap.get("objectId");
    String newValue = (String) requestMap.get("New Value");

    if ( objectId != null && newValue != null)
    {
        // Vault newVault = new Vault(newValue);
        BusinessObject busObj = new BusinessObject(objectId);
        busObj.open(context);
        // busObj.setVault(context, newVault);
        BusinessObject newBusObj = busObj.change(context,
busObj.getTypeName(), busObj.getName(), busObj.getRevision(),
newValue, busObj.getPolicy().toString());
        busObj.close(context);
    }

    return (0);
}

/**
 * get getOriginatorRange for the field.
 *
 * @param context the eMatrix <code>Context</code> object
 * @param args holds the following input arguments:
 *      0 - HashMap programMap
 * @returns StringList of Range values
 * @throws Exception if the operation fails
 * @since AEF 9.5.0.0
 */

public static Object getOriginatorRange(Context context,
String[] args)
    throws Exception
{
    HashMap programMap = (HashMap) JPO.unpackArgs(args);

```

```
String objectId = (String)programMap.get("objectId");
// HashMap paramMap = (HashMap)programMap.get("paramList");
    System.out.println("objectId << " + objectId + " >>");

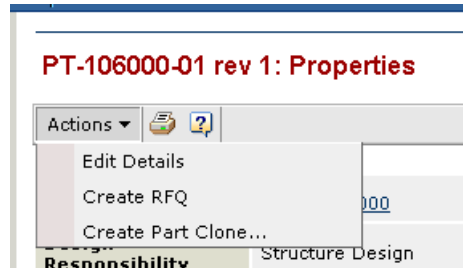
    StringList rangeList = new StringList();
    rangeList.addElement("Test SeniorDesignEngineer");
    rangeList.addElement("Test Everything");
    rangeList.addElement("Test DesignEngineer");
    rangeList.addElement("Test Buyer");

    System.out.println("rangeList << " + rangeList + " >>");

    return rangeList;
}
}
```

Building Toolbars

The configurable toolbar displays a drop-down menu of the actions that can be performed on a table page, form page, portal page, or a list of objects.



Overview of the Configurable Toolbar

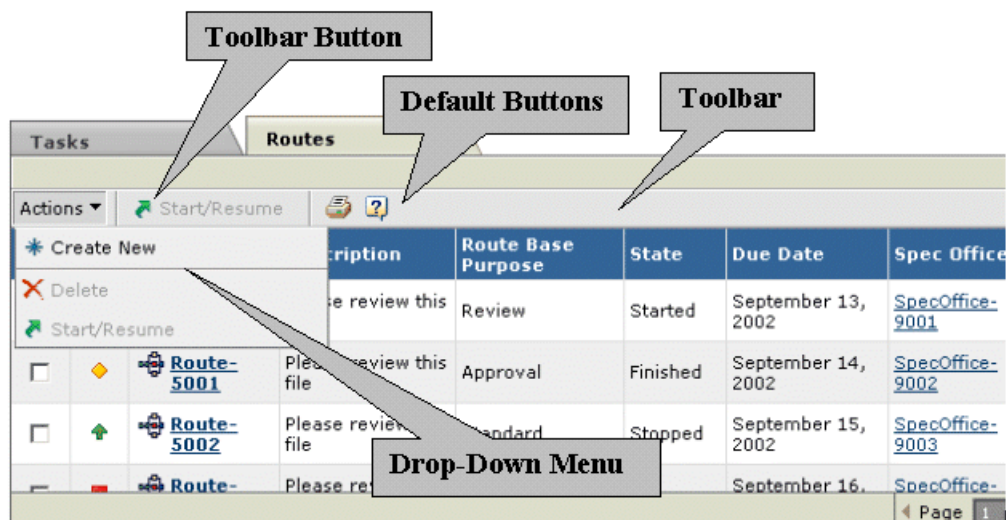
This section explains how to build and configure toolbars.

- [Overview of Table Pages](#)
- [Drop-Down Menus Can Display to the Right](#)
- [Building a Table Page](#)
- [Parameters for Table Objects](#)
- [Parameters for Tree Category Command Objects](#)
- [URL Parameters Accepted by emxTable.jsp](#)
- [Implementing a Toolbar in a JSP](#)

The configurable toolbar contains two main components:

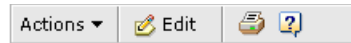
- the [Toolbar](#) itself, which is the horizontal grey bar that contains default buttons such as Help and Printer-Friendly, links, and drop-down menus with additional links
- [Drop-Down Menus](#), which expand when click to list links configured for the menu

The following figures shows the configurable toolbar within the context of a table page.



Toolbar

This graphic shows a toolbar that contains all the main components that a toolbar can have.

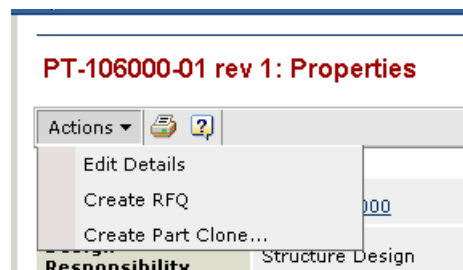


The toolbar can contain:

- Default links such as the Help and Printer-Friendly links shown above.
The toolbar component relies on querystring parameters to determine whether to display the Printer-Friendly link, the Export to Excel links, and the TipPage. By default, these buttons display, unless they are explicitly turned-off.
The Help link always displays. If no valid help marker is passed in, it defaults to the main help page for the specific application.
- Links configured for the toolbar, such as the Edit link shown above.
When the user clicks such a link, the configured page for it is called.
- Top-level menu items that contain a list of drop-down actions, such as the Actions menu item shown above.
When the user clicks the menu item, a list of drop-down actions displays.
- Vertical separators that separate links on the toolbar.
The above graphic has 2 vertical dividers. The dividers can be placed as needed. For example, you can configure the toolbar to have a divider for every two buttons. Or, a divider can be added to create a logical grouping of links, much like the grouping of new, open, save buttons on the Windows toolbar.

Drop-Down Menus

The drop-down contains the list of actions that can be performed from the current page. This graphic shows a drop-down menu.



Drop-down menus can contain:

- Top-level menu buttons that activate the drop-down menu. This is the Actions button in the graphic shown above.
The title of the top-level menu button is configurable.
The key corresponding to the display name is found in the system.properties file by the name: emxFramework.UIActionbarMenu.Label. This properties key in turn points to a key in the frameworkstringresource.properties file. By default the value is set to emxNavigator.UIMenuBar.Shortcuts. To view or change the value for the display actions name, open the emxFrameworkStringResource.properties file and look for the emxNavigator.UIMenuBar.Shortcuts key and change its value to the desired string.
- Links in the drop-down menu.

When users click the links, the page configured for that link is called.

Both the drop-down menu links and the toolbar links can be configured with the following characteristics:

- Icons/Text/Both—For example, the Edit link shown above is configured as an icon with text and the Printer-Friendly and Help links are shown above as icons only.
- Activation—Toolbar buttons and drop-down menus can be enabled and disabled. For example, if the action requires that at least one item is selected on the list page, then the action is disabled until at least one item is selected.
- Form Submission—Support for form submission through toolbar buttons/drop-down options.
- Horizontal separators.

From historical perspective, the divider represents the separation of the top and bottom toolbar items used in UI Level 3.

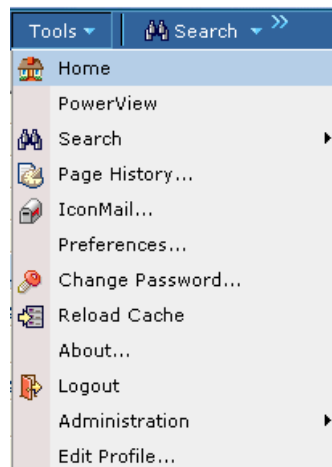
The configurable toolbar replaces the top and bottom action bars. When top and bottom action bars are used, they are each configured separately and must both be passed in as URL parameters to the configurable page. With the configurable toolbar, only a toolbar menu is required and it is passed to the configurable page. The top actions now display on top of the drop-down menu and the actions that would previously appear on the bottom actionbar appear below the horizontal separator.

For both the toolbar and the drop-down menus, there are no set limits for the number of items in the toolbar or the drop-down menus. This new behavior replaces the pagination behavior currently used with the actionbar in UI Level 3.

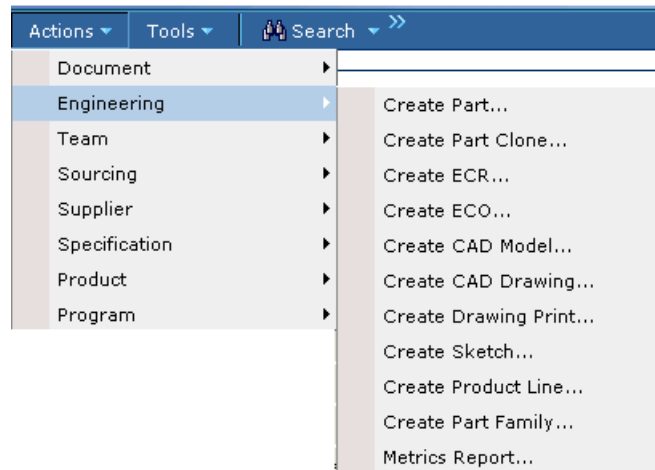
Drop-Down Menus Can Display to the Right

Using a setting for the menu object that represents the drop-down menu, you can configure a drop-down menu so the links in it display directly under the menu or to the right of it. This option to pull right is available only for the second-level drop-down menus. By default, the second-level drop-down menus pull to the right.

Here is an example of a drop-down menu that is not pulled to the right.



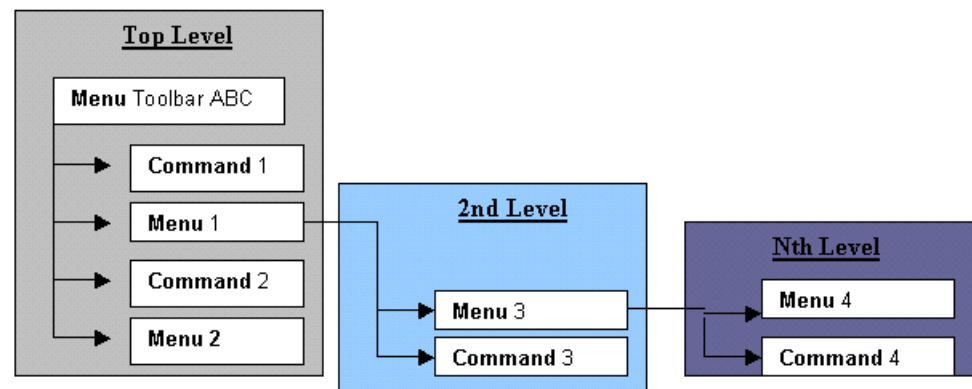
Here is an example of a menu that is pulled to the right.



Building a Configurable Toolbar

Mechanisms that Define a Configurable Toolbar

The configurable toolbar is configured using menus and commands. The toolbar supports an unlimited number of levels, but it is recommended that the number of levels be kept at 2 or 3. The following image depicts one possible structure.



The Top-Level section contains the toolbar menu and its connected menus and commands. The top-level menu, Menu Toolbar ABC in the above graphic, is an anchor for the toolbar structure. As such, it contains no settings. The connected menus and commands display as toolbar buttons on the top-level toolbar as shown below.

Command1	Menu 1	Command 2	Menu 2	

So to display a standard link in the toolbar, one that does not contain a drop-down menu, you connect the command for the link to the top-level menu object. To display a drop-down menu on the toolbar, you connect the menu object for the drop-down menu, such as Menu 1, to the top-level menu object. Then connect the commands for the links within the drop-down menu to Menu 1.

How the Toolbar is Linked to an Application

To use the configurable toolbar on a table, form, or portal page, the URL that calls the page needs to add the following parameter:

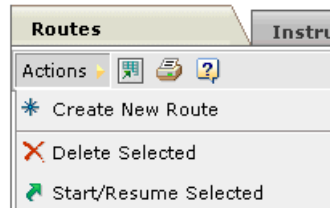
```
toolbar=TOOLBARMENU
```

Where TOOLBARMENU is the name of the toolbar menu. Below is an example URL that includes a toolbar on a table page:

```
${COMMON_DIR}/  
emxTable.jsp?program=emxAEFCollection:getObjects&table=AEFCollectionItems&toolbar=AEFCollectionsToolBar&selection=multiple
```

Steps for Building a Toolbar and Linking it to an Application

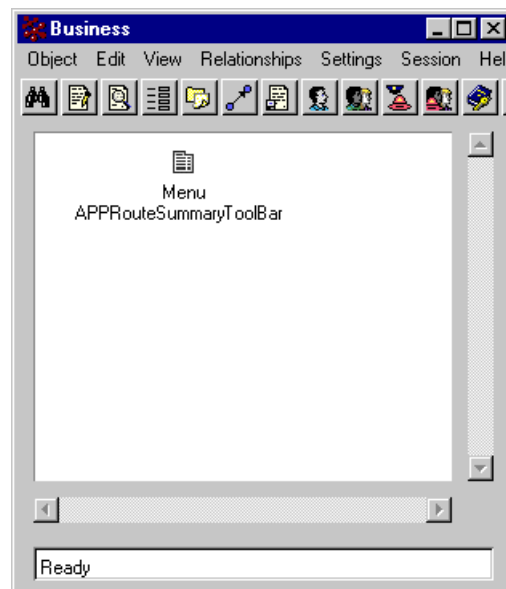
The following procedure lists the main steps needed for creating a toolbar and including it on a page. The graphics included in the procedure represent the administrative objects needed to build a toolbar for a page that lists routes, as shown in this graphic. This sample toolbar has 1 drop-down menu and 3 default links. It has no links that display on the toolbar and that are added using command objects. The drop-down menu contains 3 links plus a separator.



To create a toolbar and add it to a page in an application

1. Create a menu object to represent the top-level menu.

There are no settings needed for the top-level menu. For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

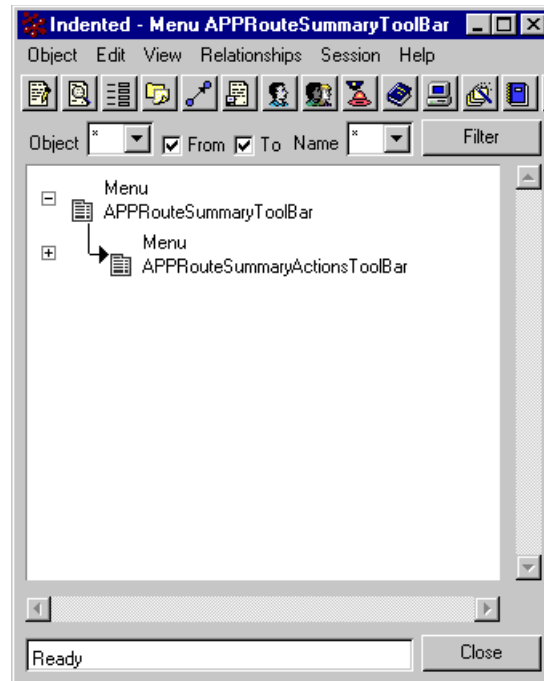


2. Create a command object for each link that users should be able to access for the page, including those that should be within a drop-down menu.

For a description of how to fill in parameters and settings for toolbar link command objects, see [Parameters for Tree Category Command Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

3. Create lower-level menu objects for each drop-down menu and assign the lower-level menu objects to the top-level menu object. For help understanding the structure, see [Mechanisms that Define a Configurable Toolbar](#).

For a description of how to fill in parameters and settings for toolbar menu objects, see [Parameters for Table Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

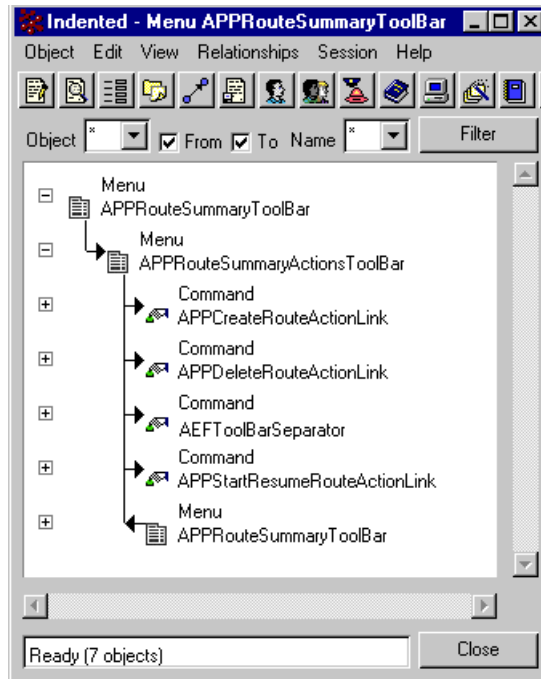


4. For each link that you want to appear on the toolbar, and not in a drop-down menu, assign the command to the top-level menu object that you created in Step 1. (The sample route toolbar shown before this procedure does not toolbar links except the default links. So no commands will be connected to the top-level menu for the sample toolbar.)

Remember to assign commands to represent separators as needed. For each separator that you want in the toolbar, assign the command `AEFToolBarSeparator`, which is installed by default with the Framework, in the appropriate order for the top-level menu. When a separator command is added to the toolbar, it appears as a vertical line.

5. For each link that you want to appear in a drop-down menu, assign it to the appropriate menu. A link can appear in both the toolbar and in a drop-down menu.

Remember to assign commands to represent separators as needed. For each separator that you want in the drop-down menu, assign the command `AEFToolBarSeparator`, which is installed by default with the Framework, in the appropriate order for the menu. When a separator command is added to a drop-down menu, it appears as a horizontal line.



6. In the URL that calls the table page, form page, or portal page that you want to include the toolbar on, include the parameter:

`toolbar=TOOLBARMENU`

Where TOOLBARMENU is the name of the toolbar menu. Below is an example URL that includes a toolbar on a table page:

```
${COMMON_DIR}/  
emxTable.jsp?program=emxAEFCollection:getObjects&table=AEFCollectionItems&toolbar=AEFToolBarMenu&selection=multiple
```

Make sure you pass in the parameters as needed for the default links. The Help link always displays. If you don't pass in any parameters, the Printer Friendly and Export links display by default and the Tip and Conversion links do not. You can pass in parameters to have these link display or not display.

Printer Friendly=false/true

export=false/true

Tip Page=false/true

Currency Converter=false/true

7. If you are working with the Web-based user interface as you are making changes and want to see your changes in the user interface, click the **Reload Cache** tool in the toolbar and click the browser **Refresh** button.

The cache is refreshed automatically when the component age expires. This setting is in `emxSystem.properties`.

Only persons assigned to the Administration Manager role have access to the Reload Cache tool.

Parameters for Toolbar Menu Objects

This table describes how to fill in the parameters for menu objects that represent the top-level menu for a toolbar and the lower-level menus. For specific instructions on how to create menu objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Alt	The text that should display in a Tooltip when the user mouses over the toolbar.	—
Commands (specified in the Items tab in Business Modeler)	The command objects that represent the links in the toolbar for the top-level menu and that represent links in the drop-down menu for lower-level menus. The order of the commands is the order the links appear on the toolbar and in the drop-down menu.	Names of command objects, such as: ENCCreateRevisionToolBarActionLink TMCEditProfileToolBarActionLink
href	The URL to call when a user clicks the label or image that represents the menu object. An href is not applicable to menu objects for My Desk, Actions, or Toolbar menus or submenus. The Toolbar menu cannot be clicked (only the individual tools can be clicked). The submenu labels for My Desk and Actions menus can be clicked but doing so only expands and contracts the menu and does not call another page.	—
Icon	Icon for the object within the Matrix core. The dynamic user interface does not display images for each application.	The name of an image file, such as Part.gif.
Label	Not applicable for top-level menus. The text that should be displayed on the toolbar menu. This is the text that appears on the “button” that users click to access the drop-down menu, such as Action. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components . The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	Create New Part emxFramework.common.Actions If the label text is not found and no image is defined, the toolbar component displays an error icon to indicate that a configuration error has occurred.
Menus (specified in the Items tab in Business Modeler)	The menu objects that represent drop-down menus. The order of the menus is the order they appear in the toolbar or drop-down menu.	Names of menu objects, such as: ECBOMListToolBar TMCToolBar
Settings	Additional settings that define the behavior and appearance of the toolbar menu objects.	Name/value pairs, as defined in the table below.

Settings for Toolbar Menu Objects

This table lists and describes the settings for menu objects that represent drop-down toolbar menus. There are no settings for the top-level menu. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
Image	Use to specify the image used for the menu. This is optional if a label has been defined but required if there is no label.	<code>\${COMMON_DIR}/iconSmallOrganization.gif</code>
Maximum Length	Limit the number of characters that are displayed on drop-down menu label. The Title property of the button is used as the Alt text to display the full label.	Any number of characters, such as 25 If this setting is not set, the toolbar displays the entire label.
*Registered Suite	The application the toolbar menu belongs to. The system looks for files related to the menu in the registered directory for that application, which is specified in <code>emxSystem.properties</code>. Based on the application name, the system passes the following parameters in the href URL: • <code>suiteKey</code> • <code>SuiteDirectory</code> • <code>StringResourceFileId</code>	The value must be set without any spaces, for example, <code>EngineeringCentral</code> or <code>Framework</code>. The value must be set to the suite name as defined in the key <code>eServiceSuites.DisplayedSuites</code> within <code>emxSystem.properties</code>. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in <code>emxSystem.properties</code> is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the menu is clicked, the system passes a parameter called “suiteKey”. The value for the parameter is the property name from <code>emxSystem.properties</code> that maps to the setting’s value.
Pull Right	Determines whether the drop-down links are displayed in the same level as the menu or pulled right to display the links in the next level. Applies only to second-level drop-down menus. For examples, see Drop-Down Menus Can Display to the Right .	True (default) False
Access Expression Access Function Access Mask Access Program	All these settings can be used to control access to tabs just as they can be used to control access to other UI components. For details, see Controlling User Access to User Interface Components .	—
*Required Setting		

Parameters for Toolbar Link Command Objects

This table describes how to fill in the parameters for command objects used for toolbar links. For specific instructions on how to create command objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/ Examples
Access	The persons, roles, and groups who can access the link. To make the link available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access. If a user doesn't have access to any links in a menu, the menu is not displayed. Also see Controlling User Access to User Interface Components .	Names of group, role, person administrative objects. or All (default)
Alt	The Alt text to appear over the link when the user mouses over it.	—
href	The URL to call when a user clicks the link.	<code>\${COMPONENT_DIR}/ emxComponentsPage.jsp, \${SUITE_DIR}/ emxTeamWorkspaceDetailsFS. jsp</code>
Icon	Icon for the object within the Matrix core. This parameter is NOT the image that displays on the dynamic user interface (for example, on a toolbar button). The Image setting is used to specify the UI image for components that have images associated with them. Many components, such as submenus, don't have images associated with them.	The name of an image file, such as Part.gif.
Label	The text that should be displayed on the toolbar link. You can configure a link to have a text label, an image, or both. If you don't specify an image using the Image setting, you must specify a label. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components . The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	Create New Part emxFramework.common.Actions If the label text is not found and no image is defined, the toolbar component displays an error icon to indicate that a configuration error has occurred.
Settings	Additional settings that define the behavior and appearance of the links. For a list of the accepted settings, see the table below.	Name/value pairs, as defined in the table below.

Settings for Toolbar Link Command Objects

This table lists and describes the settings for command objects used for toolbars. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
Access Expression Access Function Access Mask Access Program	All these settings can be used to control access to toolbars just as they can be used to control access to other UI components. For details, see Controlling User Access to User Interface Components .	—
Action Type	Tells the configurable toolbar that the command is a separator instead of a link. If the setting is not included, the toolbar treats the command as a standard link.	Separator—A line that separates one or more links. If the command is assigned to the top-level menu and is therefore on the toolbar itself, it is considered a vertical separator. Otherwise, it is considered a horizontal separator.
Confirm Message	Provides a JavaScript confirmation message when users click on the link. For example, a custom delete message can be configured for the onClick event to display a JavaScript confirm message to the user. The Confirmation dialog has the “OK” and “Cancel” button. Clicking OK proceeds with the processing and Cancel cancels the operation. You can include the number of selected table rows in the message using a macro. See Other Macros .	The actual text to display in the confirmation message or a string resource property key. To internationalize the message, a string resource key must be used: Are you sure you want to delete this object? emxFramework.common.alertMsg
Maximum Length	Limit the number of characters that are displayed on toolbar link. The Title property of the button is used as the Alt text to display the full label.	Any number of characters, such as 25 If this setting is not set, the toolbar displays the entire label.
Image	The image used for the link. This is optional if a label has been defined but required if there is no label.	\${COMMON_DIR}/ iconSmallOrganization.gif
*Required Setting		

Setting	Description	Accepted Values/Examples
*Registered Suite	The application the command belongs to. The system looks for files related to the link in the registered directory for that application, which is specified in emxSystem.properties. Based on the application name, the system passes the following parameters in the href URL: • suiteKey • emxSuiteDirectory • StringResourceFileId	The value must be set without any spaces, for example, EngineeringCentral or Framework. The value must be set to the suite name as defined in the key eServiceSuites.DisplayedSuites within emxSystem.properties. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in emxSystem.properties is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the tree category is clicked, the system passes a parameter called “suiteKey”. The value for the parameter is the property name from emxSystem.properties that maps to the setting’s value.
Popup Modal	If the setting Target Location is set to popup, the window can be configured as modal or non-modal.	true—the popup window is modal false—the popup window is non-modal If the setting is not specified, the window is modal.
*Required Setting		

Setting	Description	Accepted Values/Examples
Row Select	Used only for links on table pages. Specifies whether the JSP specified for the href expects one, at least one, or no rows in the table (Middle Frame) to be selected. If the Row Select setting is set to “single” or “multi”, the setting Submit must be set to “true” in order to get the details of the selected item(s) from the table. If the setting is not specified, it is assumed to be none. A JavaScript alert message is displayed on error conditions when the onClick event occurs. The error conditions are explained in the column to the right.	single—Appropriate only for links that appear on table pages that have radio buttons or check boxes for each row. The JSP specified for the href expects exactly one row in the table (middle frame) to be selected. If more than one item is selected, a JavaScript alert message is displayed. The default message is set to the value of the following key in emxFrameworkStringResource.properties: emxFramework.Common.PleaseSelectOneItemOnly. Currently, the message is set to “Please Select One Item Only”. If no item is selected, then the user will see the following message: “Please select an Item”. This message corresponds to this key value in the emxFrameworkStringResource.properties file: emxFramework.Common.PleaseSelectitem multi—Appropriate only for links that appear on table pages that have check boxes or radio buttons for each row. The JSP specified for the href expects at least one row in the table (middle frame) to be checked. If no item is selected, then the user will see the following message: “Please select an Item”. This message corresponds to the following key value in the emxFrameworkStringResource.properties file: emxFramework.Common.PleaseSelectitem none (default)—The JSP specified for the href expects no row in the table to be selected. If a row is selected, the JSP does not use it.
Selectable in Preferences	Only for commands in My Desk submenus. Controls whether the link is available as a preferred Home page. If not set, the system assumes true and the command is listed as a Home page preference. Commands that call pages to display in a popup window, such as an object search command, should not be available as a Home page preference.	True False
*Required Setting		

Setting	Description	Accepted Values/Examples
Submit	Specifies whether the system should send the object ID(s) (and relationship ID, if applicable) for the current page to the JSP specified in the href parameter. For form pages, the submitted data is just the object ID for the business object the page is about. For table pages, the submitted data is the ID for each selected item in the table, which can include object and relationships. In either case, the IDs are sent using the relBusIdList parameter. If the setting is not specified, it is assumed to be false.	true—Submits the IDs to the URL specified in the href parameter. false—The URL specified in the href parameter is directly called from the link and no IDs from the current page are used. Use this value for links that do not require the business object IDs, such as links for creating a new object.
Target Location	Controls where the page specified in the href parameter appears.	content—The page replaces the content frame. popup—Page appears in new window. The modality of the window is set using the Popup Modal setting. _top—Page replaces the entire body of the browser window. listHidden—Submits the table frame and carries out any background processing. hiddenFrame—Targets the Navigator hidden frame to perform any background processing. This frame should not be used within the context of the configurable table. Use listHidden instead. fromViewHidden—Used within the context of the configurable form. Any valid frame name that is available to the current frame.
Window Height	The window height for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	400 600 (default) 800
Window Width	The window width for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	400 600 (default) 800
*Required Setting		

URL Parameters Accepted by emxPortal.jsp

This table lists the parameters that emxPortal.jsp can use. You can add these parameters to the href parameter for the component that calls the portal page. For example, when you

specify the emxPortal.jsp to be called from a tree category, you can add these parameters to the href parameter for the command object.

Parameter	Description	Accepted Input Values
portal	Specifies the menu admin object that represents the portal page.	Name of menu admin object defined to represent a portal page.
toolbar	Specifies the menu admin object that represents the Actions menu, which appears in the page header.	Name of menu admin object defined to represent the Actions menu.
header	The content of the heading that appears at the top of the portal page.	Any alphanumeric text or a string resource ID. header=SummaryView header=emxSpecificationCentral.S COSummaryView.SummaryView
HelpMarker	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help .	String The naming convention for help markers “emxhelp” followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
TipPage	Specifies whether the page should include the Tip Page tool and call a specific html or jsp when a user clicks the tool. If this setting is not included, the Tip tool is not included on the page.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the portal.jsp: TipPage=../doc/customcentral/tippage.html

Implementing a Toolbar in a JSP

To implement the toolbar with a JSP page

1. Configure the required toolbar (admin menu object) and toolbar items (admin command objects) and connect them appropriately as described in [Building a Table Page](#).
2. Include these style sheets in the JSP page:
emxUIDefault.css
emxUIMenu.css
emxUIToolbar.css
3. Include these JavaScript files in the JSP page:
emxUIConstants.js
emxUICore.js

emxUICoreMenu.js

emxUIToolbar.js

4. Insert the following section of JSP code into the JSP page.

```
//*****
<%
// String sToolBarName = "<Name of the Toolbar Bar - menu
object>";
String sToolBarName = "ENCRoutesToolBar";
if ( (sToolBarName!= null))
{
%>
<jsp:include page = "emxToolBar.jsp" flush="false">
    <jsp:param name="toolbar" value="<%=sToolBarName%>"/>
    <jsp:param name="objectId" value="<%= objectId %>"/>
.....
</jsp:include>
<%
}
%>
//
*****
```

In the above code, the main parameters to be assigned are:

- sToolBarName: This is JSP page variable assigned to the name of the menu object, which is configured for the toolbar.
- toolbar: This is jsp:include parameter, assigned to the name of the Toolbar menu object.

Building PowerView Pages

A PowerView page displays information from different locations in an application on a single page, allowing users to access and update commonly-used information quickly.

This section explains how to build and configure PowerView pages.

- *Two Kinds of PowerView Pages*
- *Overview of Table Pages*
- *Building a Table Page*
- *Parameters for Table Objects*
- *Parameters for Tree Category Command Objects*
- *URL Parameters Accepted by `emxTable.jsp`*

Two Kinds of PowerView Pages

There are two kinds of PowerView pages:

- Desktop PowerView

Users can access their PowerView page using the Tools menu in the global toolbar. If the user's Home page preference is set to the PowerView page, the user can also access it using the Home tool. The PowerView lets users directly access items that are important to their daily use of the applications.

The screenshot displays the Matrix10 PowerView interface. At the top, a navigation bar includes 'My Desk', 'Tools', 'Shortcuts', 'Search', 'Home', and 'Log Out'. The 'Matrix10' logo is in the top right corner. Below the navigation bar, the 'PowerView' title is shown. The main content area is divided into two sections: 'My Collections' and 'Tasks'.

My Collections

Actions ▾

My Collections Technical Speci... Specification Tem... SCOs

Name	Count
☐ Collection-5000	11
☐ My Workspace	24
☐ Parts 2000	200
☐ Parts 3000	8
☐ Parts and ECRs for Alpha Project	8

Page 1 of 10

Tasks

Actions ▾

Name	Action	Instructions	Due Date/Time	Route
☐ Task-5000	Approve	Please review this task and determine if this task is acceptable. Provide comments on your analysis and set the action as to whether you approve or reject the item associated with this task.	Sep 2, 2002 4:00 PM EST	Route-5000
☐ Task-5011	Approve	Please review this task and determine if this task meets the company standards for this type of product. Provide comments and promote or demote the object associated with this task.	Jul 5, 2002 12:30 PM EST	Route-5000
☐ Task-5011	Approve	Please perform the assigned task by reviewing this document and note when you have completed your review. Provide additional comments that may be useful to other readers of this document.	Jul 5, 2002 10:00 AM EST	Route-5000
☐ Task-5011	Approve	Please perform the assigned task per the standard procedures applicable to this task.	Jul 5, 2002 9:00 AM EST	Route-5002

Page 1 of 10

- Object PowerView for object types

An object type can have a PowerView page defined for it that aggregates information about that object onto a single page. The PowerView can be set by the Business Administrator as the default page to display when users initially access objects of that type. The Context Navigator displays in the left frame and the PowerView page displays in the content frame on the right.

Matrix10 - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Matrix10

My Desk Tools Shortcuts Search Home Log Out User: Barry Smith

SCO-5000

- Properties
- Affected Specs/Templates
- Approvals
- History
- Lifecycle
- Routes

SCO-5000: Summary View
SCO | Draft

Actions ?

Instructions

Please complete the required information in the 'Summary' tab and click 'Done' to promote the specification change order for Peer Review.

Or, click 'Edit Details' to make other changes to the SCO.

Summary

Fields in red *italics* are required

Description Revise tolerances and measurements per contract requirements.

Reason for Change To comply with contractual requirements and meet the industry standard ISO 9001 rating.

Reviewers List RL-1123 Clear

Approvers List AL-2123 Clear

Done

Affected Specs/Templates Routes

Actions ?

Name	Rev	Type	Description	State	Business Unit	Region	Spec Office	Owner	Is ATS	Has ATS	SCO
TechSpecification-5000	3	Raw Material	Pulp Comminution (Fluff Pulp) 406MM	Draft	BusinessUnit: 5000	North America	NA - IT	Smith, Michael		Yes	
TechSpecification-			VTO TYC		BusinessUnit:		NA -	Smith			

Overview of PowerView Pages

The following graphic shows the main components of a PowerView page. This is an example of a PowerView page for an SCO. The Context Navigator, which displays to the left of the PowerView, is not shown. The components of the Desktop PowerView are similar except it is not specific to an object type and does not include a Context Navigator on the left.

SCO-5000: Summary View Page Header

SCO | Draft

Page Toolbar

Actions ▾

Instructions

Please complete the required information in the 'Summary' tab and click 'Done' to promote the specification change order for Peer Review.

Or, click 'Edit Details' to make other changes to the SCO.

Summary

Fields in red italics are required

Description	Revise tolerances and measurements per contract requirements.
Reason for Change	To comply with contractual requirements and meet the industry standard ISO 9001 rating.
Reviewers List	RL-1123 ... Clear
Approvers List	AL-2123 ... Clear

[Done](#)

Affected Specs/Te... Channel with 1 tab

Routes

Channel with 2 tabs

Actions ▾

	Name	Rev	Type	Description	State	Business Unit	Region	Spec Office	Owner	Is ATS	Has ATS	SCO
☐	TechSpecification-5000	<u>3</u>	Raw Material	Pulp Comminution (Fluff Pulp) 406MM	Draft	BusinessUnit-5000	North America	NA - IT	Smith, Michael		Yes	
☐	TechSpecification-5001	<u>2</u>	Assembly	VTO TXC RMS 102501	Issued	BusinessUnit-5000	Europe	NA - IT	Smith, Michael		Yes	

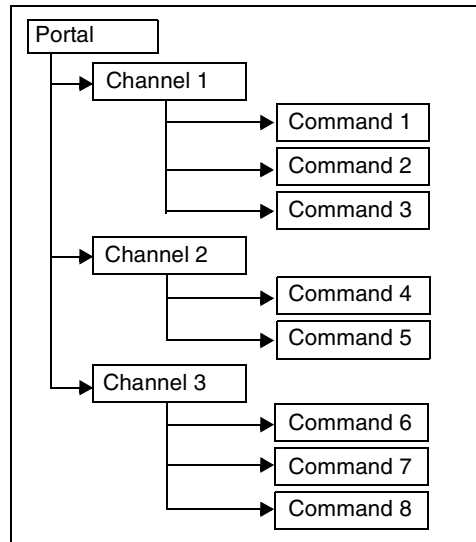
The channel tabs contain the actual content for users to view or work on. The tabs can contain any information that is normally displayed on an application page, such as an Actions menu, a configurable table with pagination controls and filter list, a read-only or editable configurable form, a custom JSP, or any Web page. For example, the channel in the upper right of the PowerView shown above contains an editable form page, and the selected tab on the bottom channel contains a table page, both of which were built with configurable components.

Clicking object links in an object PowerView page behaves the same as clicking a link displayed on any application page. The object is displayed in the main content area of the window and added to the category list under the appropriate category.

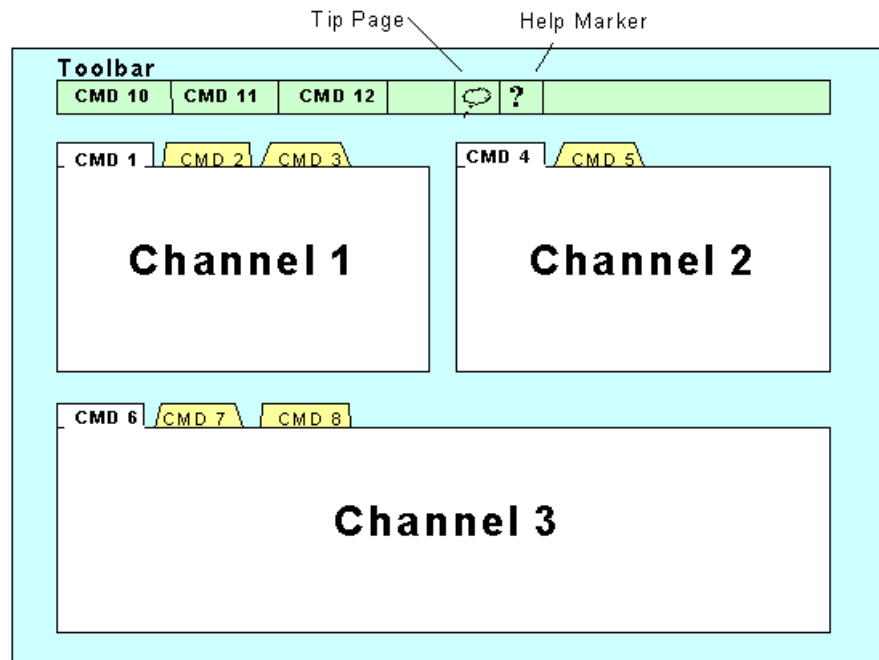
Mechanisms that Define a PowerView Page

The configurable JSP `emxPortal.jsp` creates PowerView pages. This JSP accepts parameters that define the content and behavior of the page. These parameters are passed to the JSP through the href URL.

This graphic shows the admin objects needed to define a PowerView page. The main components of the PowerView page are defined as portal, channel, and command admin objects. The names of the top-level PowerView portal object and the toolbar menu object (not shown below) are passed to the `emxPortal.jsp` through its URL parameters.



The above admin objects would create a PowerView page like the one shown below. Note that the page toolbar is also defined using menu and command objects but these objects are not shown in the above graphic. For information on page toolbars, see [Building Toolbars](#).



There should be a portal admin object for each unique PowerView page and one or more channel admin objects for each unique portal. Channel admin objects can be assigned to more than one portal object.

How the Desktop PowerView Page is Linked to the Application

The Framework installs the portal admin object called `AEFPowerView`, and the first channel of the Desktop PowerView page, called `AEFPowerViewChannel`. We also add a command to that channel called `AEFCollections`. Each installed MatrixOne application may add channel objects to this object to define channels appropriate for that application. For example, Specification Central might add a channel that contains tabs for collections, specifications, and SCOs. You can configure the channels and tabs assigned to `AEFPowerView` as needed but this object must be used for the Desktop PowerView.

The order in which the channel objects are assigned to the portal object determines where the channel displays on the PowerView page. Therefore, the order in which the applications are installed determines the default order the channels display in. For example, the channels added for the first MatrixOne application installed will be listed first on the Desktop PowerView page, the channels for the second application installed will be listed next and so on. You can change the order by changing the order in which they are assigned to the portal object.

Users typically access the Desktop PowerView page by clicking **Tools > PowerView** from the global toolbar or by clicking the **Home** tool from the global toolbar when the Home page preference is set to PowerView. The component that calls the Desktop PowerView page must specify the configurable JSP for PowerView pages, `emxPortal.jsp`, in the href parameter. The href must include a parameter that specifies `AEFPowerView`, which is the object installed by the Framework that represents the PowerView. The href should include any other parameters needed to configure the page, such as a toolbar menu object, as described in [URL Parameters Accepted by `emxPortal.jsp`](#).

For example, when entered in the href parameter for an appropriate command admin object, the following URL creates a Desktop PowerView page:

```
${COMMON_DIR}/  
emxPortal.jsp?portal=AEFPowerView&toolbar=AEFPowerViewToolbar&H  
elpMarker=HelpMarker&header=PowerView
```

How Object PowerView Pages Are Linked to MatrixOne Applications

Users typically access the PowerView page for an object type by clicking an object name from a table or by searching for a specific object. For example, in Engineering Central, clicking the name of an ECO from a table or search results displays the ECO PowerView page for that object.

The component that calls the object PowerView page must specify the configurable JSP for PowerView pages, `emxPortal.jsp`, in the href parameter. The href must include a parameter that specifies the portal object defined for the PowerView page. The href should include any other parameters needed to configure the page, as described in [URL Parameters Accepted by `emxPortal.jsp`](#).

For example, when entered in the href parameter for an appropriate command admin object, the following URL creates a PowerView page:

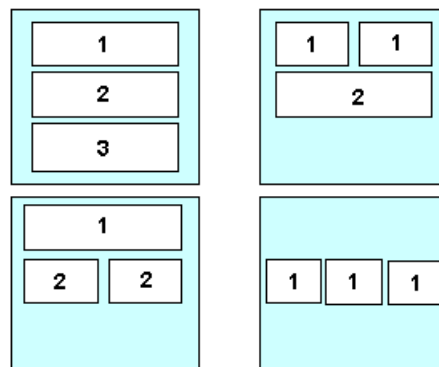
```
${COMMON_DIR}/  
emxPortal.jsp?portal=SPCSCOPortal&toolbar=SPCSCOToolbar&header=  
SummaryView&HelpMarker=emxhelpscosummary
```

Controlling the Layout of Channels

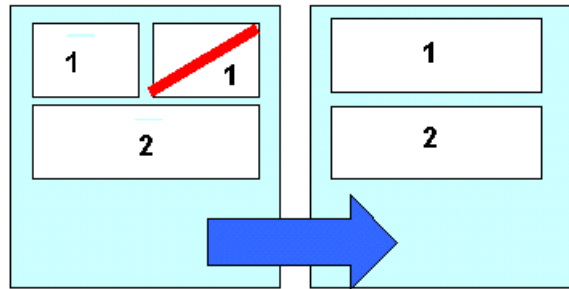
A channel can contain one or more tabs. There is no limit for the number of channels that can be displayed in a row or on a single PowerView page but displaying more than 2 channels in a row or more than 3 rows would make the page cumbersome.

The row a channel appears in is determined by how it is assigned to the portal object. If you are using Business Modeler to create portal objects, channels can be arranged in the Items tab to reflect how they will appear on the page. In MQL, use the Channel clause of the Create Portal or Modify Portal command to arrange the channels. See the *Matrix Business Modeler* or *MQL Guide* for details.

For example, channels can appear one below the other, or they can be grouped to appear side by side.



The PowerView page automatically compensates if the user does not have access to any of the tabs in the channel by hiding the channel and rearranging the display of channels. For example, if the user does not have access to the second channel in the first row, as shown in the left of the figure below, the layout shown on the right displays instead.



Displaying table component listings in half page-width channels will typically cause the user to scroll horizontally to the view all columns in the table. For this situation, you can create an alternate table with fewer columns for display specifically within a channel tab. See [URL Parameters Accepted by emxTable.jsp](#).

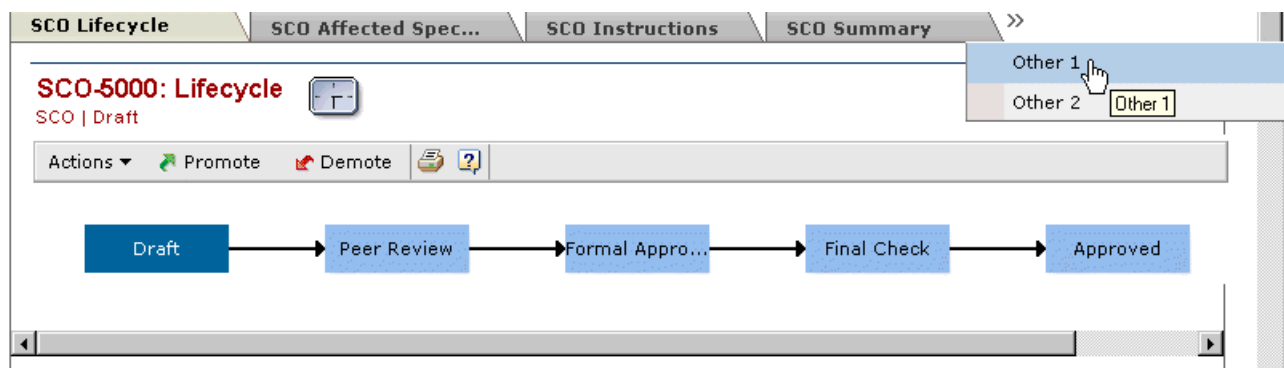
Controlling the Layout of Tabs within a Channel

The tabs within channels contain the actual content to be viewed or worked on. There is no limit to the number of tabs that can be assigned to a channel, but more than 4 or 5 tabs would be cumbersome.

The order in which tabs appear within a channel is determined by the order the tab commands are added to the channel object.

Up to 4 tabs can be displayed on a full, page-width channel and 2 tabs for half page-width channels. For channels that have more tabs, a small arrow tab is displayed. This tab provides a drop-down menu of the remaining tabs. When the user selects one of the items in the drop-down menu, the currently-selected tab is replaced by the newly-selected tab. The tab it replaces is then placed in the drop-down menu.

For example, this graphic shows a page-width channel that has 6 tabs. The 5th and 6th tabs are accessed using the >> drop-down menu. When the 5th tab (Other 1) is selected...



...it replaces the previously-selected tab (SCO Lifecycle). The previously-selected tab is then listed in the >> drop-down menu.

Other 1

SCO Affected Spec...

SCO Instructions

SCO Summary

SCO Lifecycle

Other 2

Edit Details

Fields in red italics are required

Description	Revise tolerances and measurements per contract requirements.		
Reason for Change	To comply with contractual requirements and meet the industry standard ISO 9001 rating.		
Reviewers List	RL-1123	...	Clear
Approvers List	AL-2123	...	Clear

Steps for Building and Linking a PowerView Page

The following procedure lists the main steps needed for creating a PowerView page and connecting it to an application. The procedure explains how to create all the components needed for a PowerView page from scratch. You would use this procedure if you wanted to create a PowerView page for an object type that doesn't have one by default. Since the PowerView installs by default with the Framework and each installed application adds to it, you wouldn't build a PowerView page for the Desktop PowerView from scratch. To configure the PowerView portal page or an existing object PowerView page, you add and remove portal, channel, menu and command objects and change settings for the objects.

The graphics included in the procedure represent the administrative objects needed to build an object PowerView page for technical specifications, as shown in this graphic. This sample PowerView page has 3 channels, each positioned in its own row. The first and third channels have 2 tabs (commands) each and the second has 4 tabs.

Comp1 AssemblyTechnica

Spec View

Approvals

Bill Of Specifications

Content

Rendered Files

History

Lifecycle

Products

Reference Documents

Related Specifications

Revisions

Routes

Subscriptions

Suppliers

Where Used

Comp1 AssemblyTechnicalSpecificationATS11 rev A: SummaryView

Actions

Routes

Instructions

Actions

	Name	Description	Route Base Purpose	Status	Due Date	Owner	
☐	R-0000100	Review of spec	Review	Started	Aug 29, 2003	Comp1 SpecificationManager1	

Page 1 of 1

Lifecycle

Affected Specs/Templates

Instructions

Summary

Comp1 SCO13 rev -: Lifecycle

Actions

Draft

Peer Review

Formal Appro...

Final Check

Approved

Reference Documents

Bill Of Specifications

Actions

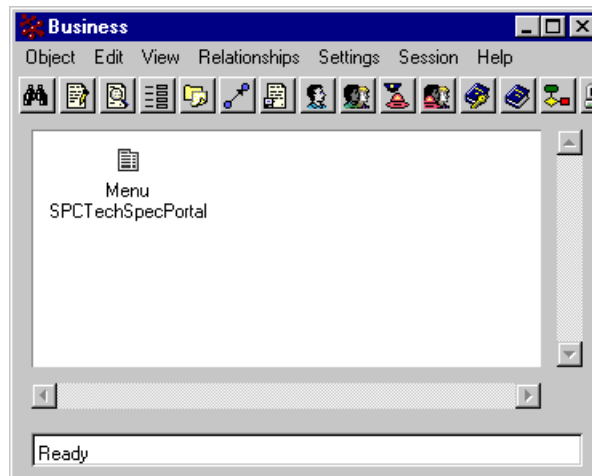
Chapter 8: Configuring the User Interface

703

To create a PowerView page and add it to an application

1. Create a portal object to represent the PowerView page.

For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

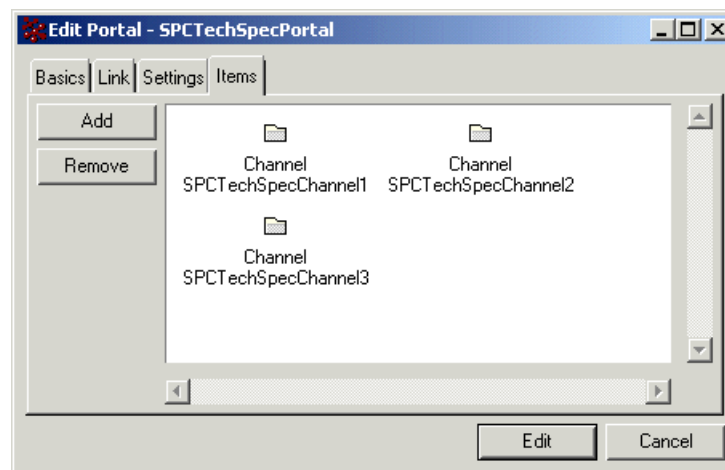


2. Create a channel object for each channel that should be on the PowerView page.

When you create a channel object, you define whether it appears alone in a row on the page or in a row with other channels. For a description of how to fill in parameters and settings for channel objects, see [Parameters and Settings for Channel Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

3. Assign the channel objects to the portal object that you created in Step 1.

You can drag the channel objects up and down or right and left within the Items tab to rearrange them in the order you want them displayed on the PowerView page. For details, see [Controlling the Layout of Channels](#).

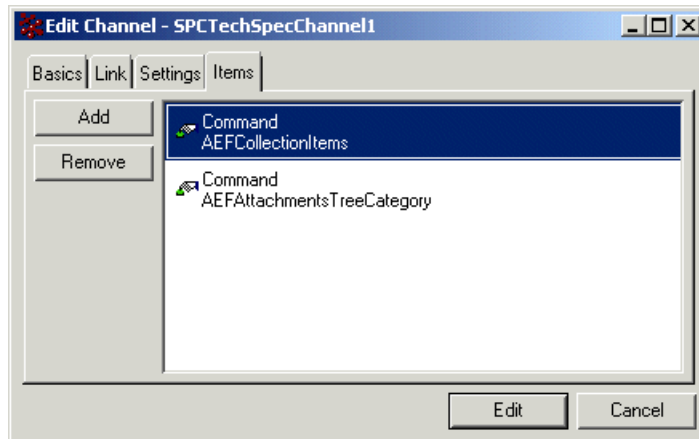


4. Create a command object for each tab that should be included for each channel.

For a description of how to fill in parameters and settings for tab command objects, see [Parameters for Tab Command Objects](#) and [Settings for Tab Command Objects](#). For naming conventions, see [Naming Conventions for UI Administrative Objects](#).

5. Assign the tab command objects to the channel that it belongs to.

The order you add the tab commands to the channel object determines the order they display on the channel. You can drag the command objects up and down within the Channel Items tab to rearrange them in the order you want them displayed.



6. Make sure the admin objects are created for the menus, toolbars, table pages, and form pages needed for the PowerView page and any of its channels/tabs.
7. In the href parameter for the menu or command object that should call the table page (or in the JSP if a JSP is calling the page), enter `emxPortal.jsp` and specify the parameters as needed to display the page. Since `emxPortal.jsp` is in the `ematrix/common` folder, the first part of the URL should usually be: `${COMMON_DIR}/emxPortal.jsp`. Here is an example of an href value for an object PowerView page. For a description of the parameters, see [URL Parameters Accepted by emxPortal.jsp](#).
`${COMMON_DIR}/emxPortal.jsp?portal=SPCSCOPortal&toolbar=SPCSCOToolbar&header=SummaryView&HelpMarker=emxhelpscosummary`
8. If you are working with the Web-based user interface as you are making changes and want to see your changes in the user interface, click the **Reload Cache** tool in the toolbar and click the browser **Refresh** button.

The cache is refreshed automatically when the component age expires. This setting is in `emxSystem.properties`.

Only persons assigned to the Administration Manager role have access to the Reload Cache tool.

Using frameset pages inside the PowerView page

In order to use the FS pages (frameset object) within any PowerView page, the following changes must be made in the FS page.

- Change the `framesetObject` constructor from...
`framesetObject fs = new framesetObject()`
change to...
`framesetObject fs = new framesetObject(request)`
- Add the following method call to tell the `framesetObject` whether the page is a table page or a form page.
If the page is a table page, use:
`fs.setPageType("table");`

Or

If the page is a form page like the Properties page:

```
fs.setPageType("form");
```

Parameters and Settings for Portal Objects

This table describes how to fill in the parameters for portal, channel, and command objects that represent a PowerView page. For specific instructions on how to create objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Name (specified in the channel Items tab in Business Modeler)	Name of the portal object. For naming conventions, see Naming Conventions for UI Administrative Objects in Chapter 8.	—
Channel	The channels associated with the portal	Names of channel objects, such as: AEFCollections AEFPowerViewChannel SPCSCOChannel

Parameters and Settings for Channel Objects

This table describes how to fill in the parameters for channel objects that represent a PowerView page. For specific instructions on how to create objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/Examples
Name	Name of the channel.	Name of the channel
Height	The height of the channel.	The default is 260.
Commands (specified in the channel Items tab in Business Modeler)	The command objects that represent the tabs in the channel. The order of the commands is the order the tabs appear in the channels in the user interface.	Names of command objects, such as: SPCSCOInstructions SPCRoutesTreeCategory

Parameters for Tab Command Objects

This table describes how to fill in the parameters for command objects used for tabs that appear within channels on PowerView pages. For specific instructions on how to create

command objects using Business Modeler or MQL, refer to the *Business Modeler Guide* or *MQL Guide*.

Parameter	Description	Accepted Values/ Examples
Access	The persons, roles, and groups who can access the tab. To make the tab available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access. If a user doesn't have access to any tabs in a channel, the channel is not displayed. Also see Controlling User Access to User Interface Components .	Names of group, role, person administrative objects. <i>Or</i> All (default)
Alt	The Alt text to appear over the tab label when the user mouses over the label.	WBSTasks emxEngineeringCentral.common.ECRs
Href	The URL to display when a user selects the tab. Tabs can display content from the following sources: - configurable table - configurable form - JSP page - URL If the specified URL contains a header parameter, it is not used because the header would be redundant with the tab label. For example, if the URL contains emxTable.jsp with a header specified, the table header is not used.	emxTable.jsp?xxxxx emxForm.jsp?xxxxx Sample.jsp?xxxxx http://www.matrixone.com
*Label	The text that should display on the tab. The label can be made up of: - Plain text, such as "ECRs". - A string resource ID, such as "emxEngineeringCentral.common.portal". To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components . The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text and/or output from the select expression. The label can be up to 17 characters in length. If it exceeds 17 characters, it is truncated to 17 characters and an ellipsis ("...") is added to the label. This default can be changed in emxsystem.properties with the property emxFramework.PowerView.Channel.Label.MaximumLength. The HTML "Title" attribute is defined so that when the user mouses over the tab, the full label displays. 	Properties EBOM Lifecycle emxEngineeringCentral.common.portal
Settings	Additional settings that define the behavior and appearance of the tab command. For a list of the accepted settings, see the table below.	Name/value pairs, as defined in the table below.

Settings for Tab Command Objects

This table lists and describes the settings for command objects used for portals. Note that the name and value for each setting are case sensitive.

Setting	Description	Accepted Values/Examples
Access Expression Access Function Access Mask Access Program	All these settings can be used to control access to tabs just as they can be used to control access to other UI components. For details, see Controlling User Access to User Interface Components .	—
*Registered Suite	The application the command belongs to. The system looks for files related to the command in the registered directory for that application, which is specified in emxSystem.properties.	The value must be set without any spaces, for example, EngineeringCentral or Framework. The value must be set to the suite name as defined in the key eServiceSuites.DisplayedSuites within emxSystem.properties. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in emxSystem.properties is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the tab is clicked, the system passes a parameter called “suiteKey”. The value for the parameter is the property name from emxSystem.properties that maps to the setting’s value.
*Required Setting		

URL Parameters Accepted by emxPortal.jsp

This table lists the parameters that emxPortal.jsp can use. You can add these parameters to the href parameter for the component that calls the portal.jsp. For example, when you specify the emxPortal.jsp to be called from a tree category, you can add these parameters to the href parameter for the menu object.

Parameter	Description	Accepted Input Values
portal	Specifies the portal admin object that represents the top-level menu, which defines the channels.	Name of portal object
toolbar	Specifies the menu admin object that represents the toolbar, which appears in the page header.	Name of menu admin object.

Parameter	Description	Accepted Input Values
HelpMarker	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help .	String The naming convention for help markers “emxhelp” followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
TipPage	Specifies whether the page should include the Tip Page tool and call a specific html or JSP when a user clicks the tool. If this setting is not included, the Tip tool is not included on the page.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the table.jsp: TipPage=../doc/customcentral/tippage.html
header	The content of the header that appears at the top of the PowerView page.	Any alphanumeric text or a string resource ID. header=PowerView header=“ECR Summary”
subHeader	Creates a subHeader below the main header in the PowerView header frame.	The value can be any static text or a string resource id. The value can also include macros such as \$<type> \$<revision>.

Portal.jsp passes a URL parameter portalMode=true to all JSP pages configured within the PowerView page so that the pages can differentiate between normal display and portal display.

URL Parameters Accepted by emxTable.jsp

Displaying table component listings in half page-width channels will typically cause the user to scroll horizontally to the view all columns in the table. For this situation, you can create an alternate table with fewer columns for display specifically within a channel tab. For example, if a standard ECO New Parts table contains 10 columns, you could define an alternate ECO New Parts table that contains just 4 columns. The portalTable parameter can be used to pass in to emxTable.jsp the name of the alternate table to be displayed within a channel tab.

You can add this parameter to the href parameter for the component that calls the table.

Parameter	Description	Accepted Input Values
portalTable	Specifies the alternate table that can be used within a channel tab.	Name of alternate table

Configuring Preference Pages

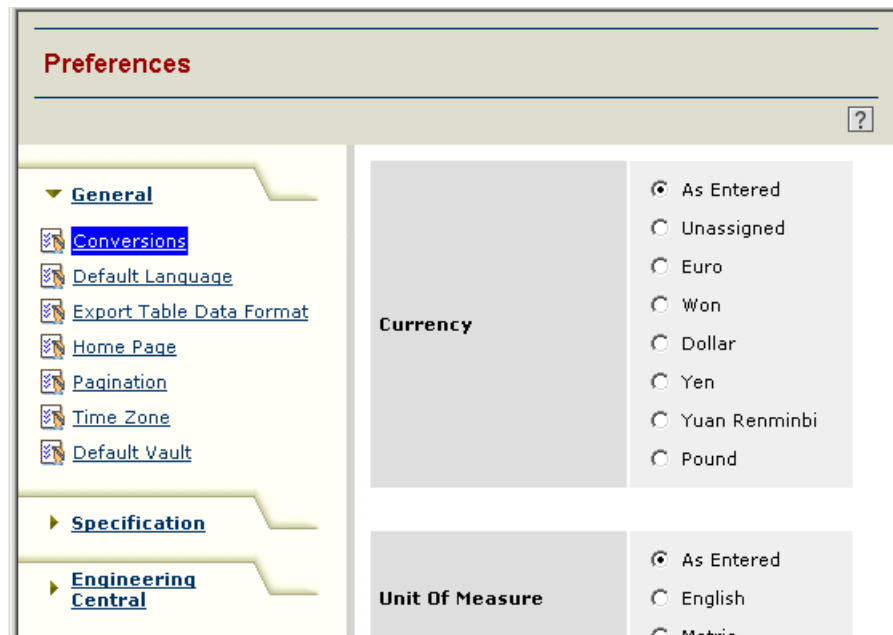
This section describes how to configure and build preference pages that let users specify general preferences for all applications and preferences specific to an application.

- [Overview of Preferences](#)
- [Building Preference Pages](#)
- [Configuring General Framework Preferences](#)

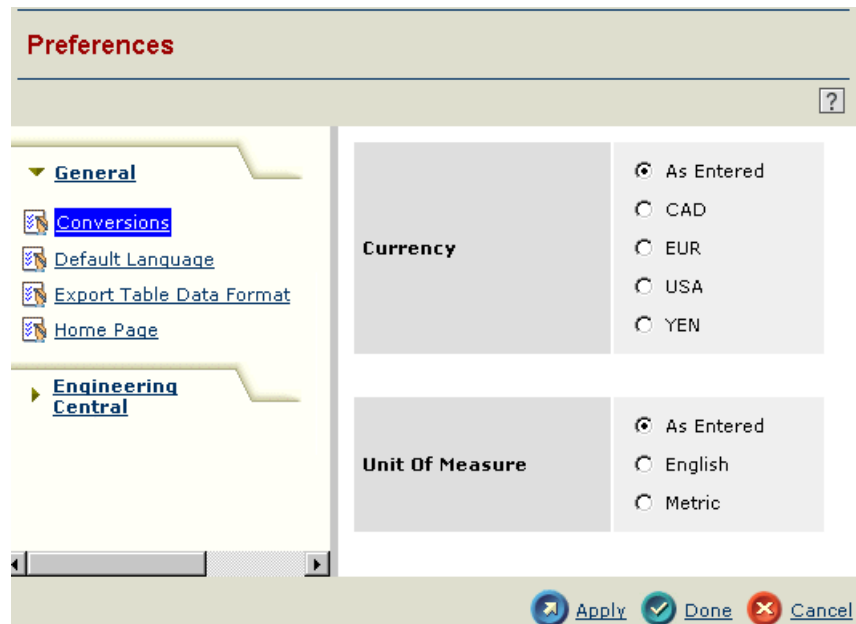
Overview of Preferences

The Tools menu on the global toolbar contains a Preferences link.

When a user clicks the Preferences link, the Preferences page appears in a popup window.



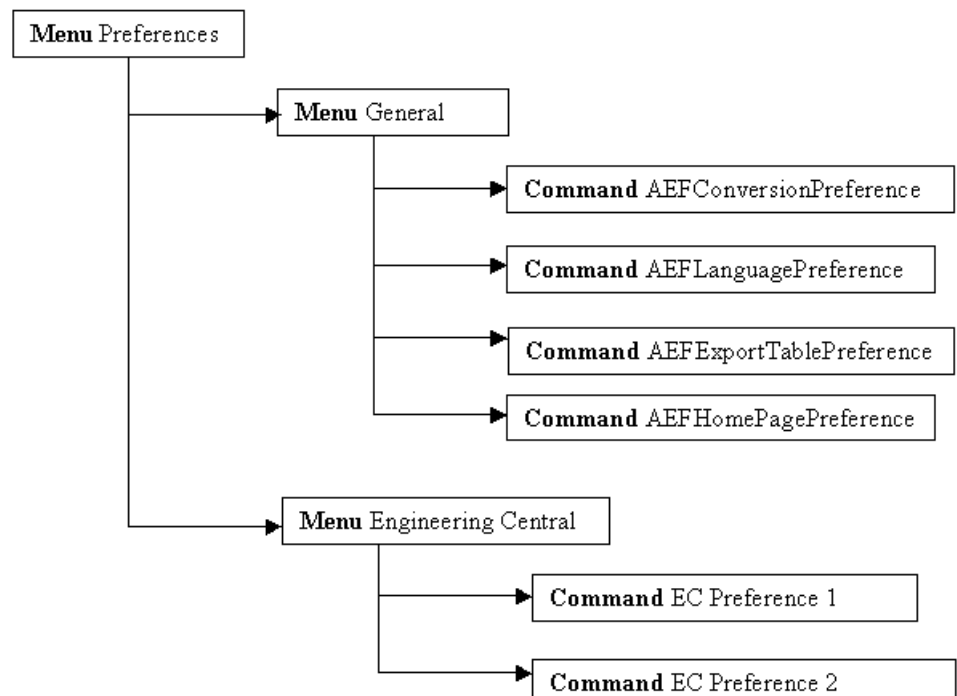
Links for specific preferences can be categorized based on whether they apply to multiple applications, General, or to a specific application. The framework installs with a number of general preferences and specific MatrixOne applications may install with additional general or application-specific preferences. All these preferences can be configured and custom general and application specific preferences can be added. For example, here is a Preferences page with preferences added for Engineering Central.



Building Preference Pages

Mechanisms that Define Preferences

The framework installs an administrative menu object, called Preference, which represents the Preferences page. Categories of preferences, such as General and Engineering Central, are defined using menu objects that are assigned to the Preferences menu. For example, to define the General category for preferences, the framework installs with a menu object called General, which is assigned to the Preferences menu. The specific preferences within a category are defined using command objects which are in turn connected to the menu object for the category. For example, to define the link for the Home Page Preference page, the framework installs with a command called AEFHomePagePreference. This command is assigned to the General menu. This graphic shows a sample admin object structure needed for an installation that uses the general preferences installed with the framework and two additional preferences for Engineering Central. All the objects shown here are installed with the framework except the menu and two command objects for Engineering Central. These objects might be installed with Engineering Central or could be added as custom preferences.



Steps for Adding a Preference Page

The procedure below lists the main steps for adding a preference page and adding a new category to place the new page under, if needed. The graphics in the procedure illustrate the steps needed to create a preference called Default Templates for the Custom Central application, as shown in this graphic:

Preferences

?

▼ General

Conversions

Default Language

Export Table Data Format

Home Page

▼ Custom Central Preferences

Default Templates

Currency

☒ As Entered

☐ CAD

☐ EUR

☐ USA

☐ YEN

Unit Of Measure

☒ As Entered

☐ English

☐ Metric

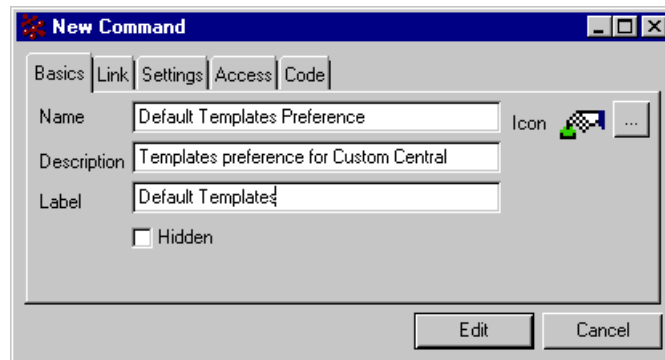
Apply

Done

Cancel

To add a preference page

1. Create a command object for the preference and fill in the parameters and settings for the command as follows.



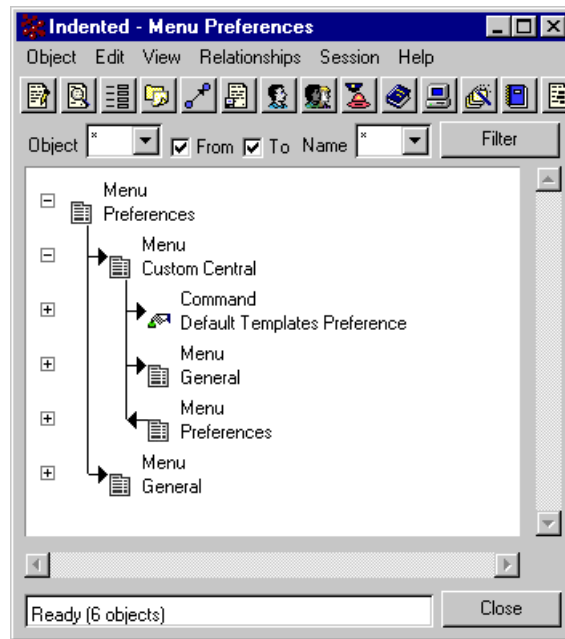
Parameter/ Setting	Description	Accepted Values/ Examples
Label	The text that should be displayed on the link for the preference. The link appears under the category name the command is assigned to. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components. The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	Home Page Date/Time Formal emxFramework.Prefere nces.Conversions
Name	Name of preference.	Home Page Default Language
href	The URL that gets executed when the link for the preference is clicked. This URL is displayed in the right frame of the Preferences page. The value for the href parameter should be a JavaScript function or JSP and any associated parameters. You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros.	<code>\${COMMON_DIR}/ emxPrefConversions.jsp</code> <code>\${SUITE_DIR}/ emxENCPrefVault.jsp</code>
Access	The persons, roles and groups who can access the preference. To make the link available to all users, regardless of role/group assignments, choose All.	Names of group, role, person administrative objects. or All (default)

Parameter/ Setting	Description	Accepted Values/ Examples
Setting: Image	Icon to be displayed in the Preference window.	COMMON_DIR/ buttonToolbarPreferences.gif
Setting: Registered Suite	The name of a suite that registered this command	Framework TeamCentral

- If you need to create a new category for the preference, create a menu object for the category and fill in the parameters and settings as follows. Assign the menu object to the Preferences menu. If the preference fits under an existing category, skip to Step 3.

Parameter/ Setting	Description	Accepted Values/ Examples
Label	The text that should be displayed for the category on the Preferences page. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components. The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	Engineering Central Custom Central
Name	Name of category.	Engineering Central Custom Central
Setting: Registered Suite	The name of a suite that registered this command	Framework TeamCentral

- Assign the command you created in Step 1 to the appropriate category menu object. This is the menu object you created in Step 2 or an existing menu object.



Configuring General Framework Preferences

This section describes configuration options for the general preferences installed by the framework.

For instructions on how to use a preference page, click the  Help button on the page.

Currency Conversion

This preference lets users choose the currency in which they want to view monetary attributes, such as unit price. Currently, only Sourcing Central uses currency and unit of measure conversions.

When at least one column in a configurable table is defined with the setting `format=currency`, the table page includes a Conversion  tool in the page toolbar. (The tool also displays if a column is configured with `format=UOM`, for unit of measure conversion.) Applications may also include the Conversion tool in the page toolbar for pages that are not built using the configurable table. Like the configurable Conversion tool, these non-configurable instances of the Conversion tool operate using the currency selected in preferences, as described below.

When a user opens a page that has column data configured as `format=currency`, all currency data is listed as the data was entered. When the user clicks the  tool, the page opens in a new window and all currency columns are converted to the currency chosen on the Currency and Unit of Measure Preferences page, shown above (unit of measure columns are also converted). If the default selection of `As Entered` is selected as the preferred currency, no conversion is made.

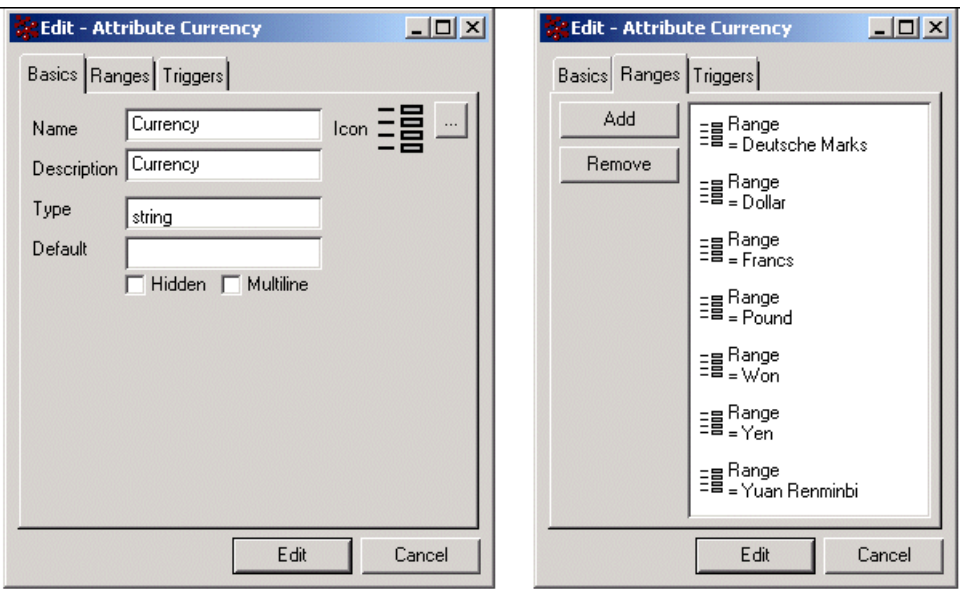
This table summarizes the configuration options for currency conversion preferences.

Preference	Available choices defined by:	Default value defined by:	User's selection stored:
Currency conversion	Range values on the Currency attribute	Specifying a default range value for the Currency attribute	for global use in current session

To implement currency conversions

1. Define which currencies users can convert to, for example, you could choose to let users convert to Yen, Francs, and Dollars. In Business Modeler or MQL, add and remove range values for the Currency attribute so it includes all currencies that you want to convert from and to. Use the standard abbreviations for each currency.

Range values on Currency attribute determines currency options in preferences



- 2. Within Supplier Admin in Sourcing Central, define currency conversions for each company using the Currency Exchange Rates category in the company’s category list. For instructions, see the Sourcing Central help or user guide.
- 3. In a configurable table, identify the currency columns that users should be able to convert by adding the format=currency setting for the column.

To display the table with the converted currency data in the new window, the table component uses the ConvertCurrencyTag TagLib. The TagLib uses these parameters to configure the displayed converted data.

TagLib Parameter	Column Setting	Description
from	Currency Expression	The name of the currency to convert from. Setting “Currency Expression” is assigned to a select clause, which provides the value for the Currency attribute for a specific object or relationship. This should be the currency the user entered the data in (the As Entered currency). For example, to convert data for an RFQ Quotation, the following select clause returns the currency format. to[Supplier Line Item].attribute[Currency]
to		The name of the currency to convert to. The value for this “to” parameter is obtained from the session and is the preferred currency selected by the user. If not defined, no conversion is applied.
value		Actual value to be displayed in the column (currency).

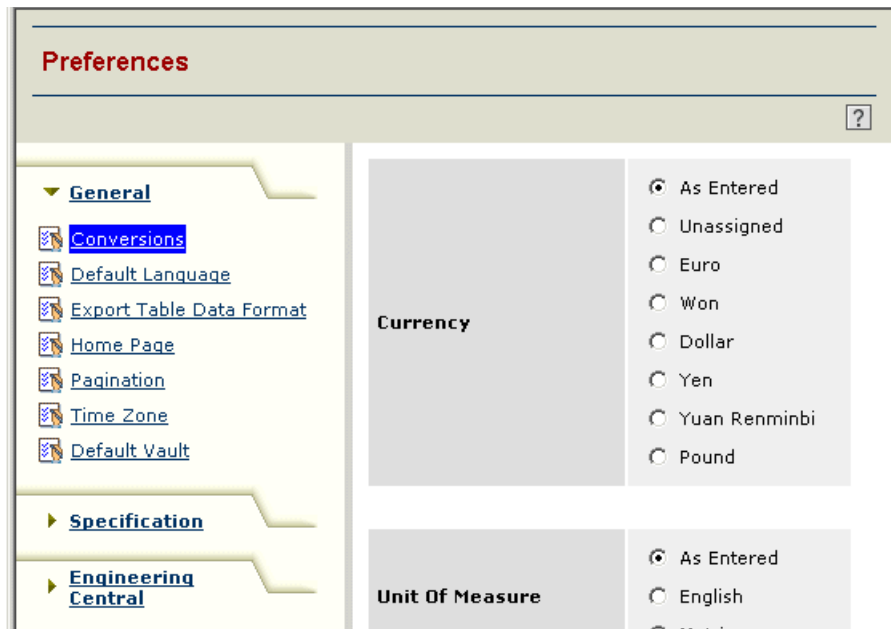
TagLib Parameter	Column Setting	Description
date	Effective Date Expression	The “Effective Date Expression” is assigned to a select clause, which provides the value for the Effectivity Date attribute on a specific object or relationship. The system uses this date to get the currency conversion whose rate period falls within this date. For example, for data for an RFQ Quotation, the following select clause returns the effective date. <pre>to[Supplier Line Item].attribute[Effectivity Date]</pre>
decimalSeparator		Decimal separator symbol that should be used for displaying the currency value. This symbol is defined in <code>emxSystem.properties</code> using the key: <code>emxFramework.DecimalSymbol = .</code>
digitSeparatorPreference		Determines whether to display a thousandths separator for currency and quantity fields (for example, 1,000). This symbol is defined in <code>emxSystem.properties</code> using the key: <code>emxFramework.DigitSeparator = false</code>

For example:

```
<framework:convertCurrency
  from="Dollar"
  to="Yen"
  value="50000"
  date="25/05/2001"
  decimalSeparator="."
  digitSeparatorPreference="true"
```

Unit of Measure Conversion

This preference lets users choose the unit of measure in which they want to view measurement attributes, such as Weight. For example, if a user enters weight data using the Metric unit grams, another user can convert this data to the English unit pounds. Each English unit can be converted to only one Metric unit. Currently, only Sourcing Central uses currency and unit of measure conversions.



When at least one column in a configurable table is defined with the setting format=UOM, the table page includes a Conversion  tool in the page toolbar. (The tool also displays if a column is configured with format=currency, for currency conversion.) .

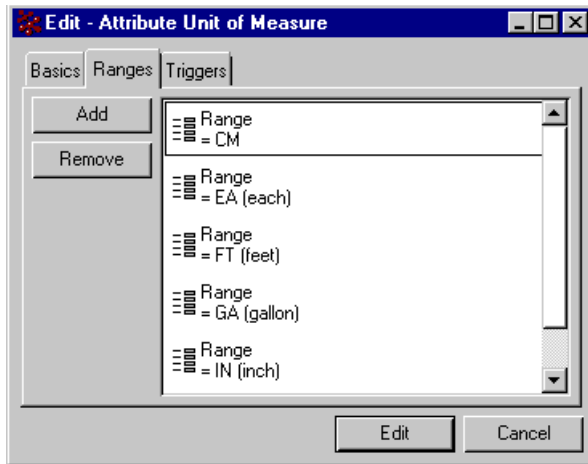
When a user opens a page that has column data configured as format=UOM, all measurement data is listed as the data was entered. When the user clicks the  tool, the page opens in a new window and all measurement columns are converted to the unit of measure chosen on the Currency and Unit of Measure Preferences page, shown above (currency columns are also converted). If the default selection of As Entered is selected as the preferred unit of measure, no conversion is made.

This table summarizes the configuration options for this preference.

Preference	Available choices defined by:	Default value defined by:	User's selection stored:
Unit of measure	Range values on the Unit of Measure attribute	Specifying a default range value for the Unit of Measure attribute	for global use in current session

To define the conversion rates for English and Metric units, perform these steps for each English unit that you want users to be able to convert to a Metric unit:

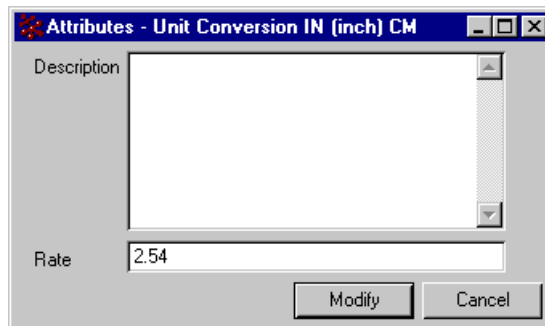
1. In Business Modeler or MQL, add and remove range values for the Unit of Measure attribute so they include all English and Metric units that you will define conversions for.



2. Using Matrix Navigator, PowerWeb, Info Central or CWI, create a business object of type Unit Conversion. The Name should match the English unit and the Revision should match the Metric unit. Both units must be added to the range values for the Unit of Measure attribute. Make sure the object is stored in the eService Administration vault.



3. In the Rate attribute for the Unit Conversion object, enter the conversion rate between the English and Metric unit.



To display the table with the converted measurement data in the new window, the table component uses Tag Lib. The following table lists column settings that need to be configured for Unit of Measure columns.

Taglib Parameter	Column Setting	Description
map		Table component obtains the conversion map from the system using the bean UnitConversion.

from	UOM Expression	The name of the Unit of Measure to convert from. The Unit of Measure Expression setting for the table column is assigned to a select clause, which provides the value for the Unit of Measure attribute for a specific object or relationship. For example, for data for RFQ Quotations, the following select clause returns the currency format. <pre>to[Supplier Line Item].attribute[Unit of Measure]</pre>
to		The name of the unit of measure to convert to. The value for this “to” parameter is obtained from the session and is the preferred currency selected by the user. If not defined, no conversion is applied.
value		Actual value to be displayed in the column (measurement value obtained from database).
decimalSeparator		Decimal separator symbol that should be used for displaying the measurement value. This symbol is defined in <code>emxSystem.properties</code> using the key: <code>emxFramework.DecimalSymbol = .</code>
digitSeparatorPreference		Determines whether to display a thousandths separator for currency and quantity fields (for example, 1,000). This symbol is defined in <code>emxSystem.properties</code> using the key: <code>emxFramework.DigitSeparator = false</code>

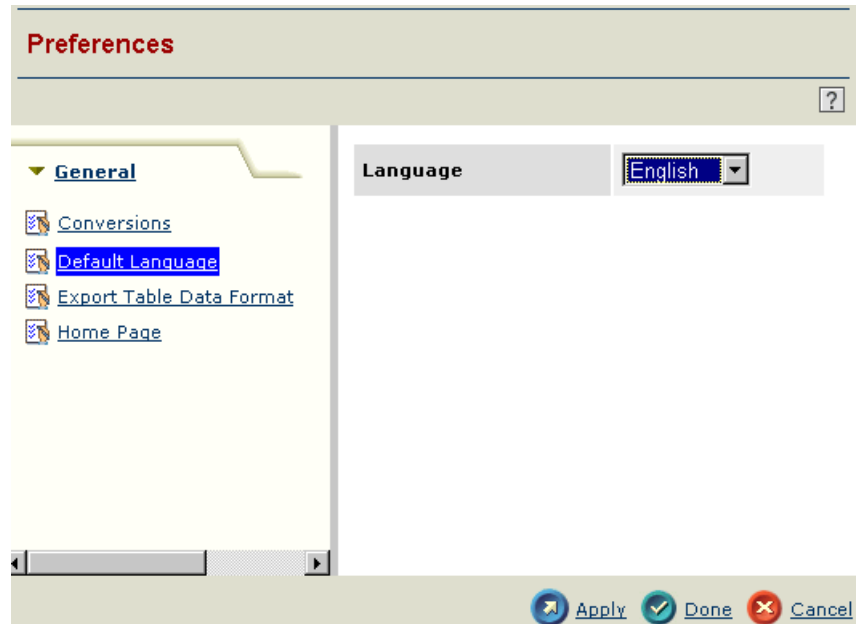
For example:

```
<framework:convertUnit
  map = <%=UnitConversion.getUnitConversionMap(context)%>
    from="IN (inch) "
    to="English"
    value="22323"
    decimalSeparator="."
    digitSeparatorPreference="true"
```

Default Language for System Notifications

This preference lets users choose the language in which they want to receive system notifications.

Some applications include an equivalent preference setting on the person profile page. For example, in Team Central, the page for editing a person’s profile information includes a setting called System Generated Mail Preference. This is the same setting as the Default Language setting in Preferences. If the user chooses a particular language from Preferences, the field on the profile page shows the same language selected and vice versa.



This table summarizes the configuration options for this preferences page.

Preference	Available choices defined by this property in emxSystem.properties	Default value defined in this property in emxSystem.properties:	User's selection stored:
Language for notifications	emxFramework.Preferences.Language.Choices = English, Japanese, Italian, French, German, Chinese	emxFramework.Preferences.Language.Default = English	IconMailLanguagePreference property on the user's person admin object

Export Table Data Format

This preference lets users choose the format they want to use when exporting table data. If the user chooses Text format the additional choices for separators and how to handle carriage returns are also used.

Preferences

General

- Conversions
- Default Language
- Export Table Data Format**
- Home Page

Format

☐ CSV
☐ HTML
☒ Text

Record Separator

Field Separator

Remove Carriage Returns

☒ Yes
☐ No

Apply Done Cancel

This table summarizes the configuration options for this preferences page.

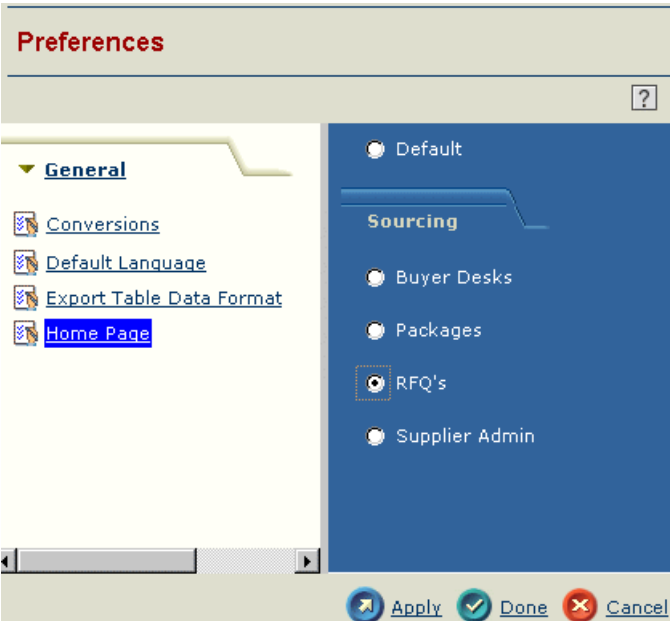
Preference	Available choices defined by this property in emxSystem.properties	*Default value defined in this property in emxSystem.properties:	User's selection stored:
Export table data format	emxFramework.Preferences.ExportFormat.Choices = CSV, HTML, Text	emxFramework.Preferences.ExportFormat.Default = CSV	ExportFormat property on the user's person admin object
Field separator (available only when export type is Text)	emxFramework.Preferences.FieldSeparator.Choices = Pipe, Tab, Comma	emxFramework.Preferences.FieldSeparator.Default = Comma	preference_FieldSeparator property on the user's person admin object
Record separator (available only when export type is Text)	This is a text box so no options are available for selecting.	The default value is New Line	preference_RecordSeparator property on the user's person admin object
Remove carriage returns (available only when export type is Text)	emxFramework.Preferences.RemoveCarriageReturns.Choices = Yes, No	emxFramework.Preferences.RemoveCarriageReturns.Default = Yes	preference_RemoveCarriageReturns property on the user's person admin object
Multiple column value separator	If a column contains multiple values, each value will be separated by the value assigned to this property. The default is a new line (\n).		N/A
* The default value just defines the selected value if the user doesn't change it. The user's preference always overrides the default value.			

You can also define the value used to separate multiple values in a column using this property in emxSystem.properties. The separator only applies to CSV and text formats, not to HTML. This is only a system-wide setting. Users cannot set it through preferences.

emxFramework.Preferences.FieldValueSeparator.Delimiter = \n

Home Page Preference

This preference lets users choose their home page. This is the page that appears when users first log in and when they click the Home Page tool.



This table summarizes the configuration options for this preferences page.

Preference	Available choices defined by this property in emxSystem.properties	Default value defined in this property in emxSystem.properties:	User's selection stored:
Home Page	All commands on the user's My Desk menu are listed, except those with Selectable in Preferences set to False.	Not applicable.	In two properties on the user's person admin object: preference_Menu stores the application menu name and preference_Command stores the specific My Desk command within that application menu

Using Other Configurable Pages

This section describes how to use configurable pages that let users perform specific tasks, such as viewing object history and choosing an object type.

- [Configurable History Page and History Commands](#)
- [Configurable Type Chooser](#)
- [Lifecycle Page](#)

Configurable History Page and History Commands

The framework installs with a configurable History page called emxHistory.jsp. The History page lists the actions that have occurred for the current business object. By passing parameters to the JSP, the page can be configured to:

- display events for the current revision only or for all revisions
- label the revisions “revisions” or “versions”, when configured to display events for all revisions
- show only specific action types
- show or hide the action type filter
- show or hide the text filter

Configurable Options for the History Page

Header text

Whether to display Action Type and text filters

Default action types to filter on

Display history for current revision or all revisions

Labelling of “revision” or “version”

Molded Part CM-202030-01: History

From Rev: 1 to Rev: 2 Refresh

Action Type * and * Filter

Revision 1

Date	User	Action	Action Details	State	Message
Aug 9, 2002 3:16:36 PM	Test Everything	connect	Design Responsibility from Department Mechanical Design Company Name Mechanical Design	Release	
Aug 9, 2002 3:16:24 PM	Test Everything	connect	New Part / Part Revision from ECO ECO	Release	
Aug 9, 2002 3:16:17 PM	Test Everything	override		Review	
Aug 9, 2002 3:16:16 PM	Test Everything	promote		Review	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	EBOM from Mechanical Part CM	Preliminary	
Aug 9, 2002 3:16:01 PM	Test Everything	create		Preliminary	revised from:
Aug 22, 2002 10:49:08 AM	Test Everything	revised		Release	revision: Molded Part CM-202030-01 2

Revision 2

Date	User	Action	Action Details	State	Message
Aug 22, 2002 10:49:12 AM	Test Everything	modify		Preliminary	Production Make Buy Code: Buy was: Make
Aug 22, 2002 10:49:12 AM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	

Preconfigured History Command Objects

The framework also installs with three commands that can be used by applications to call the History page. Each of these commands and the configuration of the History page that

each command calls is described below. All three commands are configured to display the Action Type and text filters and have no prefilter action types.



- AEFHistory command

This command calls the History page with a parameter set so only the actions for the current revision of the business object are listed. The History page always includes the ability for users to filter the list by action type and by specific text strings in any column other than action type. This command is designed to display the History page in the content frame and therefore should be used as a tree category.

Current Revision
Mode

Header

Molded Part CM-202030-01: History

Action Type ... and

Date ▼	User	Action	Action Details	State	Message
Aug 9, 2002 3:16:36 PM	Test Everything	connect	Design Responsibility from Department Mechanical Design Company Name Mechanical Design	Release	
Aug 9, 2002 3:16:24 PM	Test Everything	connect	New Part / Part Revision from ECO ECO	Release	
Aug 9, 2002 3:16:17 PM	Test Everything	override		Review	
Aug 9, 2002 3:16:17 PM	Test Everything	promote		Release	
Aug 9, 2002 3:16:16 PM	Test Everything	override		Preliminary	
Aug 9, 2002 3:16:16 PM	Test Everything	promote		Review	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	EBOM from Mechanical Part CM	Preliminary	
Aug 9, 2002 3:16:01 PM	Test Everything	create		Preliminary	revisioned from:
Aug 22, 2002 10:49:08 AM	Test Everything	revisioned		Release	revision: Molded Part CM-202030-01 2

- AEFHistoryAllRevisions command

This command calls the History page with a parameter set so actions are listed for all revisions of the object. When the History page includes all revisions of the object, it also includes drop-down lists in the upper right corner that let users specify the revisions they want to see. This command is designed to display the History page in a popup window and therefore should be used as a toolbar item.

All Revisions
Mode
Header

Molded Part CM-202030-01: History

From Rev: 1 to Rev: 2 Refresh

Action Type * and * Filter



SubHeader

Revision 1

Date	User	Action	Action Details	State	Message
Aug 9, 2002 3:16:36 PM	Test Everything	connect	Design Responsibility from Department Mechanical Design Company Name Mechanical Design	Release	
Aug 9, 2002 3:16:24 PM	Test Everything	connect	New Part / Part Revision from ECO ECO	Release	
Aug 9, 2002 3:16:17 PM	Test Everything	override		Review	
Aug 9, 2002 3:16:16 PM	Test Everything	promote		Review	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	EBOM from Mechanical Part CM	Preliminary	
Aug 9, 2002 3:16:01 PM	Test Everything	create		Preliminary	revised from:
Aug 22, 2002 10:49:08 AM	Test Everything	revised		Release	revision: Molded Part CM-202030-01 2

Revision 2

Date	User	Action	Action Details	State	Message
Aug 22, 2002 10:49:12 AM	Test Everything	modify		Preliminary	Production Make Buy Code: Buy was: Make
Aug 22, 2002 10:49:12 AM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	

- AEFHistoryAllVersions command

This command calls a History page just like the one for the AEFHistoryAllRevisions command except revisions are referred to as “versions”. For example, Team Central calls document revisions “versions”. This command is designed to display the History page in a popup window and therefore should be used as a toolbar item.

All Revisions
Mode using
“Versions”

Molded Part CM-202030-01: History

From Ver: 1 to Ver: 2 Refresh

Action Type * and * Filter



Version 1

Date	User	Action	Action Details	State	Message
Aug 9, 2002 3:16:36 PM	Test Everything	connect	Design Responsibility from Department Mechanical Design Company Name Mechanical Design	Release	
Aug 9, 2002 3:16:24 PM	Test Everything	connect	New Part / Part Revision from ECO ECO	Release	
Aug 9, 2002 3:16:17 PM	Test Everything	override		Review	
Aug 9, 2002 3:16:16 PM	Test Everything	promote		Review	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	
Aug 9, 2002 3:16:06 PM	Test Everything	connect	EBOM from Mechanical Part CM	Preliminary	
Aug 9, 2002 3:16:01 PM	Test Everything	create		Preliminary	revised from:
Aug 22, 2002 10:49:08 AM	Test Everything	revised		Release	revision: Molded Part CM-202030-01 2

Version 2

Date	User	Action	Action Details	State	Message
Aug 22, 2002 10:49:12 AM	Test Everything	modify		Preliminary	Production Make Buy Code: Buy was: Make
Aug 22, 2002 10:49:12 AM	Test Everything	connect	Part Family Member from Part Family CM	Preliminary	

Parameters Used by the History Page

The emxHistory.jsp page accepts the parameters listed in this table. Additionally, emxHistory.jsp requires the objectId parameter so it knows which object to get history for. The objectId parameter is automatically passed to the page from the tree in the Current Revision mode and from the toolbar item in the All Revisions mode.

Parameter	Description	Accepted Input Values
Header	The text to use for the header of the History page. This parameter can either be a string resource ID or a mixture of macros and text or simply text.	<code>\$(type) \$(name) \$(revision)</code> <code>emxFrameworkStringResource.Common.HistoryPageHeading</code> Object History
HistoryMode	Determines whether the entire revision chain or just the current revision history should be displayed.	AllRevisions—The page displays the entire revision history of the object. CurrentRevision (default)—The page displays the history for the particular revision that is being reviewed. For example, a part may have REV A, Rev B and Rev C. If we are reviewing REV B and the HistoryMode is set to “CurrentRevision”, only the history for REV B is presented. If HistoryMode is set to “AllRevisions”, then the history for REV A, B, C is presented.
preFilter	Determines which action types to display when the History page comes up. If the parameter is not passed, the page lists all action types. If the Action Type filter control is shown on the page (determined by the ShowFilterAction parameter), the preFilter actions are listed in the text box and the filter is applied to the history list when the page first opens. The user can use the Action Type filter to change the filter, adding and removing displayed actions as needed. If the Action type filter control is not shown, the user cannot change the preFilter actions. By setting preFilter actions and hiding the Action Type filter, you can hide specific actions that you do not want users to see.	The parameter accepts two types of values: - Comma delimited list of action types: <code>preFilter=connect, disconnect, create</code> - String Resource ID: <code>preFilter=emxSystem.preFilter.List</code> To specify which string resource properties file to look in to get the value, include the SuiteKey parameter. If the SuiteKey is not passed in, the History page looks for the key in the emxSystem.properties file. For example, in the emxSystem.properties file, the value for the emxSystem.preFilter.List might be “connect, create, disconnect”.
ShowFilterAction	Determines whether to display the action type filter or not. Action Type * ...	true (default)—The action type filter displays. false—The action type filter does not display.
ShowFilterTextBox	Determines whether to display the filter text box or not. * Filter	true (default)—The filter text box displays. false—The filter text box does not display.

Parameter	Description	Accepted Input Values
subHeader	Only needed when using All Revisions mode. Determines whether to display Version or Revision for the SubHeader, which separates the history events for each revision.	Version—The History page looks in the emxFrameworkStringResource.properties file for value of the emxFramework.History.Version property. Revision—The History page looks in the emxFrameworkStringResource.properties file for value of the emxFramework.History.Revision property.
SuiteKey	Determines which string resource property file to look in for the preFilter property key. This parameter is only used when the preFilter parameter is passed in as a resource ID. For example, SuiteKey=eServiceSuiteEngineeringCentral would be required for the preFilter parameter whose value is in the EngineeringCentral.properties file, emxEngineeringCentral.properties.	Possible values for the SuiteKey parameter are: eServiceSuiteMatrixCentral, eServiceSuiteInfoCentral, eServiceSuiteTeamCentral, eServiceSuiteEngineeringCentral, eServiceSuiteProgramCentral, or any other suite name.

Including the Configurable History Page in an Application

This simplest way to use the History page in an application is to use one of the three command objects installed with the framework, described at the beginning of this section. To use one of these command objects in an application, assign the command object to the appropriate menu object. For example, to use the AEFHistory object, assign it to the menu object that represents the tree you want to add the History category to. To use the AEFHistoryAllRevisions or AEFHistoryAllVersions objects, assign the object to the menu object for the toolbar item.

Alternatively, you can configure your own History command object by following these steps:

1. In the href parameter of the admin command object that will call the History page, enter the emxHistory.jsp, plus the needed parameters (described in the table above) and a path indication. For example:

```

${COMMON_DIR}/
emxHistory.jsp?HistoryMode=AllRevisions&Header=emxFrameworkStringResource.Common.HistoryPageHeading&subHeader=Version&preFilter=emxEngineeringCentral.history.preFilterList&SuiteKey=eServiceSuiteEngineeringCentral
${COMMON_DIR}/
emxHistory.jsp?HistoryMode=CurrentRevision&Header=${<type>}
${<name>}
${<revision>}&subHeader=Revision&ShowFilterAction=true&ShowFilterTextBox=false&preFilter=connect,disconnect,promote

```
2. If you are configuring a toolbar item and therefore the page needs to display in a popup window, enter these settings:
Target Location=popup
Row Select=none
Registered Suite=Framework

3. If you are configuring a tree category and therefore the page needs to display in the content frame, enter these settings:
 Target Location=content
 Image=IMAGE_FILENAME
 Registered Suite=Framework
4. Connect the command object to the menu admin object that will contain it. For example, if the command object is for a toolbar item, connect the command to the menu object for the Actions menu. If it's a tree category, connect it to the menu object for the tree.

Configurable Type Chooser

Many business processes within the MatrixOne applications require that users specify a type for a business object. When searching for objects, users can often narrow the search to specified types. The framework installs with a configurable Type Chooser called `emxTypeChooser.jsp`. The Type Chooser lists types defined in the database and lets users choose from the list. The Type Chooser contains text filter fields and a hierarchical tree structure that helps users find the type(s) for which they are looking.

The Type Chooser can be called from a field defined on a configurable Form page or from any standard JSP. In this section, we'll use the term "form page" to refer to this parent page that the Type Chooser is called from. For example, a Create New Part page might require the user to specify a type for a part, as shown below. The user accesses the Type Chooser by clicking the Browse (...) button next to the Type field.

Step 1 of 2: Create New Part

Fields in red italics are required

<i>Part Number or</i>	☐ Autaname
<i>Autaname Series</i>	Not Selected
<i>Type</i>	Radiographic Part Browse
<i>Custom Revision Level</i>	3

The Type Chooser opens with the top-level types that were configured to display. Users can click the + signs to see sub-types of the displayed top-level types or uncheck Top Level Only to see all sub-types. For instructions on how to use the page, see the AEF Online Help.

The screenshot shows a web form titled "Select Type" in a red font. Below the title is a search bar with a dropdown menu set to "begins with" and a text input field containing an asterisk (*). To the right of the search bar is a checked checkbox labeled "Top Level Only" and a "Filter" button. Below the search bar, there is a tree view under the heading "Types". The tree view shows a folder icon followed by "Types", and a sub-item "Part" with a plus sign and a small icon. At the bottom of the form, there are two buttons: a green "Submit" button with a checkmark icon and a red "Cancel" button with an 'X' icon.

By passing parameters to the JSP, the page can be configured to:

- display or not display specific top-level types
- let users choose only one type or multiple types
- display or not display hidden types
- allow users to select abstract types or not
- show icons for types or not
- call a reload method on the form page to updated values based on the selected type

Parameters Used by the Type Chooser

The emxTypeChooser.jsp page accepts the parameters listed in this table.

Parameter	Description	Accepted Input Values
*fieldNameActual	Use to specify the field on the form page to populate the type(s) the user selects in the Type Chooser. The system populates the specified field with the actual Matrix type names, as stored in the database, so the field must be a hidden field. When calling the Type Chooser using a RangeHref on a configurable Form page, the hidden field is created automatically and has the same name as the RangeHref field. Other pages that need to get the type name selected by the user, for example to perform a search, should get the type name from this field instead of the fieldNameDisplay.	The name of a hidden field on the form page. For example, if the form page contains a field defined as follows: <pre><input type="hidden" name="txtSelectedTypeName" value=""></pre> This parameter should be passed to emxTypeChooser.jsp: <pre>fieldNameActual=txtSelectedTypeN ame</pre>
*fieldNameDisplay	Use to specify the text field on the form page to populate the type(s) the user selected in the Type Chooser and display these types to the user. When calling the Type Chooser using a RangeHref on a configurable Form page, the display field is created automatically and has the same name as the RangeHref field with "Display" appended to it. This field contains the internationalized version of the selected type list.	The name of a non-hidden field on the form page. For example, if the form page contains a field defined as follows: <pre><input type="text" name="txtType" value="" onClick="showTypeChooser()"></pre> This parameter should be passed to emxTypeChooser.jsp: <pre>fieldNameDisplay=txtType</pre>
ExclusionList	Use to define the types to exclude from the top-level list of types that display when the Type Chooser opens. The inclusion list and the exclusion list cannot both be passed as parameters. If they are, an error message is displayed. If neither is specified, all types are listed. The chooser displays all types in the top level of the hierarchy, even when Top Level Only is unchecked.	Same as InclusionList.
*Required Setting		

Parameter	Description	Accepted Input Values
formName	Use to specify the name of the form that holds the field specified in fieldNameDisplay. If this parameter is not passed in, the Type Chooser looks for the field name on the first form of the form page. If the field name is not found in the first form, then an error message will be presented.	The name of a form on the form page. For example, if the form page contains a form defined as follows: <pre><form name="searchPage" method="post" action="somejsp.jsp"></pre> This parameter should be passed to emxTypeChooser.jsp: <pre>formName=searchPage</pre>
frameName	Use to specify the name of the frame that contains the form on the form page. Needed for pages that contain forms in multiple frames.	The name of a frame on the form page.
InclusionList	Use to define the top-level list of types to display when the Type Chooser opens. Note that these types do not have to be top-level types (types with no parents), they will just be listed in the top level in the chooser. The inclusion list and the exclusion list cannot both be passed as parameters. If they are, an error message is displayed. If neither is specified, all types are listed. The chooser displays all types in the top level of the hierarchy, even when Top Level Only is unchecked.	- a properties file key If the SuiteKey parameter is passed in, the system looks for the key in the application-specific properties file (for example, emxEngineeringCentral.properties). If the property is not application specific, the system looks in emxSystem.properties. For example, if emxEngineeringCentral.properties contains this property: <i>eServiceEngineeringCentral.Types=type_Part,type_ECR,type_Sketch</i> The parameter to pass would be InclusionList=eServiceEngineeringCentral.Types - a comma delimited list of symbolic names for the types to include: InclusionList=type_Part,type_ECR
ObserveHidden	Determines whether to display hidden types or not.	true (default)—Hidden types are included in the chooser's list of types. false—Hidden types are not included.
ReloadOpener	Determines whether to call a reload() method on the opener/parent page (the form page from which the chooser is launched). Reloading the page updates other fields on the form page based on the type(s) selected from the Type Chooser. The method should be a JavaScript function stored in the header section of the JSP or HTML page.	true—The reload method is called. false (default)—The reload method is not called.
SelectAbstractTypes	Determines whether users can select abstract types.	true—Abstract types can be selected. false (default)—Abstract types cannot be selected but they are displayed in the hierarchical list of types.
*Required Setting		

Parameter	Description	Accepted Input Values
SelectType	Determines whether the Type Chooser has single select radio buttons or multi-select check boxes.	multiselect—Users can choose multiple types using check boxes. singleselect (default)—Users can choose only one type using a radio button.
ShowIcons	Determines whether to display types icons in the Type Chooser.	true (default)—Type icons are displayed. false—Type icons are not displayed. This improves performance.
SuiteKey	Determines which application's properties file to look in for the key specified in the InclusionList or ExclusionList parameter. If the SuiteKey is not passed in, emxSystem.properties is used.	Possible values for the SuiteKey parameter are: eServiceSuiteMatrixCentral, eServiceSuiteInfoCentral, eServiceSuiteTeamCentral, eServiceSuiteEngineeringCentral, eServiceSuiteProgramCentral, or any other suite name.
*Required Setting		

Calling the Configurable Type Chooser from a Page

The Type Chooser can be called from a field defined on a configurable Form page or from any standard JSP. Either way, the page that calls the Type Chooser must have these two fields defined:

- a hidden field that stores the actual type name as stored in the Matrix database.
This is the field defined in the fieldNameActual parameter passed to emxTypeChooser.jsp.
- a field that displays the type(s) selected in the chooser.
This is the field defined in the fieldNameDisplay parameter.

The configurable Form page creates these two fields automatically when you add a field to the web form admin object that has a RangeHref defined. The hidden field has the same name as the RangeHref field added to the form. The display field has the same name with "Display" appended to the end of the field name. To call the Type Chooser, the URL entered for the RangeHref must be emxTypeChooser.jsp and its associated parameters. For more information about configuring Form pages and using RangeHref fields, see [Controlling User Access to User Interface Components](#). See the procedure below for details on the steps for calling the Type Chooser from the configurable Form.

When calling the Type Chooser from a standard JSP, you must define a form on the page that contains a text field to display the selected type and a hidden field. For example:

```
<form name="searchPage" method="post" action="somejsp.jsp">
<input type="text" name="txtType" value=""
onClick="showTypeChooser()">
..
<input type="hidden" name="txtSelectedTypeName" value="">
</form>
```

The URL to call the configurable Type Chooser would look like this:

```
../common/
emxTypeChooser.jsp?fieldNameDisplay=txtType&fieldNameActual=txt
SelectedTypeName&formName=searchPage&SelectType=multiselect&Sel
```

```
ectAbstractTypes=true&
InclusionList=eServiceEngineeringCentral.Types&observeHidden=true&SuiteKey=eServiceSuiteEngineeringCentral&ShowIcons=true
```

If you add a new type in Matrix (and have registered it) and want to include it in the Type Chooser, refresh the cache using the Refresh Cache tool.

To call the configurable Type Chooser from a field on a configurable Form

1. If you will use a property to define a list of types to include or exclude, add this property to the appropriate properties file. When defining the URL for `emxTypeChooser.jsp`, enter the property key in the `InclusionList` or `ExclusionList` parameter. If the file is an application-specific file, you'll need to specify the app using the `SuiteKey` parameter.
2. In Business Modeler, open for editing the web form object that contains the field.
3. Open for editing the field that should call the Type Chooser.
4. In the `RangeHref` parameter for the field, enter the `emxTypeChooser.jsp`, plus the needed parameters (described in [Parameters Used by the Type Chooser](#)) and a path indication.

The value for the `fieldNameActual` parameter should be the name of the field you are editing. So if the field's name is `PartType`, the value should be `PartType`.

The value for the `fieldNameDisplay` parameter should be the name of the field you are editing appended with "Display". For a field named `PartType`, the value for this parameter should be `PartTypeDisplay`.

Here are some example Hrefs:

```
${COMMON_DIR}/
emxTypeChooser.jsp?fieldNameActual=PartType&fieldNameDisplay=PartTypeDisplay&formName=PartList&SelectType=multiselect&InclusionList=eServiceEngineeringCentral.Types&ObserveHidden=False&SuiteKey=eServiceSuiteEngineeringCentral&ShowIcons=False
${COMMON_DIR}/
emxTypeChooser.jsp?fieldNameActual=PartType&fieldNameDisplay=PartTypeDisplay&formName=PartList&SelectType=multiselect&ExclusionList=eServiceEngineeringCentral.Types&ObserveHidden=True&SuiteKey=eServiceSuiteEngineeringCentral&ShowIcons=True
```

Internationalizing Selected Type Names

The Type Chooser supports internationalization. As with all framework components, the system looks up the values for types selected from the chooser from the `emxFrameworkStringResource.properties` file. The value(s) is stored in the hidden field on the form, as defined in `fieldNameActual`. For example, if a user who has their browser set to French selects the type "Part" from the chooser, the `fieldNameDisplay` parameter is populated with the internationalized string value, the equivalent of "Part" in French. The Matrix type name is populated in the hidden field passed in with the `fieldNameActual` parameter.

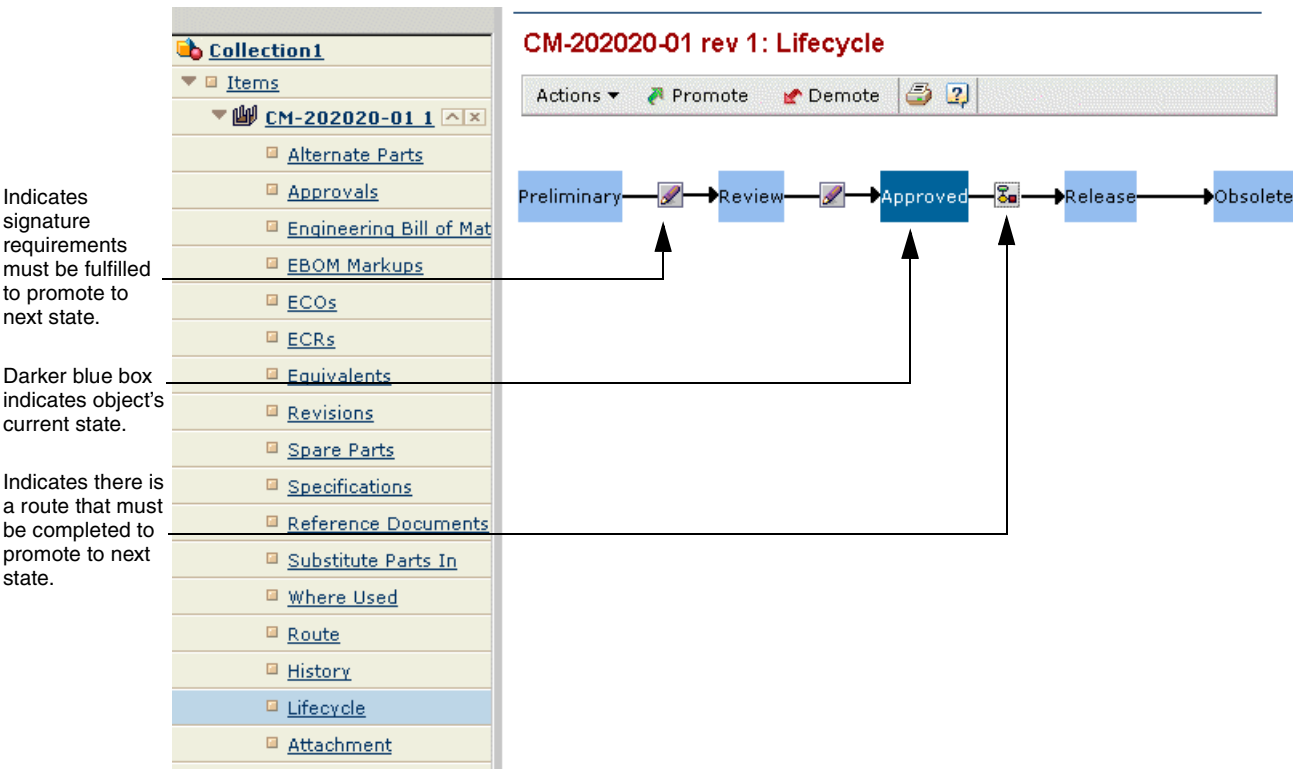
For more information on internationalizing dynamic UI components, see [Internationalizing Dynamic UI Components](#).

Lifecycle Page

The framework's Lifecycle page, `emxLifecycle.jsp`, displays the lifecycle for the current business object, including:

- signature requirements
- branches (does not indicate the branch taken to arrive at a state)
- any blocking routes
- the current state

Users can use the page to view and apply signatures, promote and demote the object, and view blocking routes.



The framework installs with a command object called `AEFLifecycle`, which calls the lifecycle page and is configured to be a category in a navigation tree.

You can pass these parameters to `emxLifecycle.jsp`.

Parameter	Description	Accepted Input Values
actionMenuName	The name of the menu	Name of any menu object that represents a menu. The framework installs with a command object called <code>AEFLifecycleMenu</code> that includes a Promote and Demote link.
header	The title for the page.	Text string or string resource, such as: Lifecycle <code>emxFramework.Lifecycle.LifeCyclePageHeading</code>
*Required Setting		

When configured as part of a navigation tree for an object, the system automatically passes the objectID for the object. From the object ID, the page gets the object's type and policy. From the policy, it retrieves the states, signatures, and branches. For policies with branches, when a user clicks Promote, the system promotes to the state for which signature requirements have been met. When naming signatures for branched states, do not use the prefix "GoTo" or the signature box on the Lifecycle page will be hidden, preventing users from being able to apply signatures.

Standard Commands and Pages

This section describes the standard commands that are installed with the framework.

- [Logout Command](#)
- [Change Password Command](#)
- [Page History Page and Command](#)

Also see [Configuring Preference Pages](#).

Logout Command

The framework installs a command object, called AEFLogoutToolbar, that represents the

Logout  tool. This command is assigned to the Toolbar menu. When a user clicks the Logout tool, the system calls emxLogout.jsp, which ends the current session and returns to the Login page.

If in a Single Signon (SSO) environment, the SSO server should be configured to by pass emxLogout.jsp. The user must exit the browser to log out. For instructions on how to by pass emxLogout.jsp, see [Single Signon](#) in Chapter 7.

The tables below describe the parameters and settings used to implement the Logout command. For details on configuring global toolbar tools, see [Building Menus and the Global Toolbar](#).

Parameter	Description	Installed Input Values
href	The JSP that implements the Logout command. When the JSP is prefixed with <code>\${ROOT_DIR}</code> , the system looks for the JSP in the ematrix root directory, which it gets from <code>emxSystem.properties</code> . The property used is “eServiceSuiteFramework.Root Directory = ..”	<code>\${ROOT_DIR}/emxLogout.jsp</code>
Alt	The internationalization string ID to display as mouse over text.	ID from string resource properties file.
Access	Defines who can access the tool.	All

Setting	Description	Installed Input Values
*Registered Suite	The application the command belongs to. The system looks for image and JSP files in the directory associated with the application. Internationalization details are also obtained based on the Registered Suite.	Framework
Target Location	The frame or window the page should display in.	hiddenFrame
Image	The image displayed on the tool.	buttonToolbarLogout.gif
*Required Setting		

Change Password Command

The framework installs a command object, called AEFChangePasswordToolbar, that represents the Change Password  tool. This command is assigned to the Toolbar menu. The Change Password command calls emxChangePassword.jsp, which lets users change their own password. The page displays in a popup window. Before making the password change, the page checks to make sure the current password the user enters is correct and makes sure the new password and verify new password entries are the same.

Change Password

Fields in red italics are required ?

<i>Username</i>	Test Everything
<i>Current Password</i>	
<i>New Password</i>	
<i>Verify New Password</i>	

 Submit
 Cancel

The tables below describe the parameters and settings used to implement the Change Password command. For details on configuring global toolbar tools, see [Building Menus and the Global Toolbar](#).

Parameter	Description	Installed Input Values
href	The JSP that implements the Change Password page.	emxChangePassword.jsp
Alt	The internationalization string ID to display as mouse over text.	emxNavigator.UIMenu.ChangePassword
Access	Defines who can access the tool.	All

Setting	Description	Installed Input Values
Help Marker	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help .	String The naming convention for help markers is “emxhelp” followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.
*Registered Suite	The application the command belongs to. The system looks for image and JSP files in the directory associated with the application. Internationalization details are also obtained based on the Registered Suite.	Framework
Target Location	The frame or window the page should display in. By default, the page displays in a separate popup window. The target location can be changed to any specific existing frame name to display it in that frame.	popup The window is modal.
Window Height	The window height for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	400
Window Width	The window width for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	400
Image	The image displayed on the tool.	buttonToolbarPassword.gif
*Required Setting		

Page History Page and Command

The framework installs a command object, called AEFPageHistoryToolbar, that

represents the Page History  tool. This command is assigned to the Toolbar menu. The Page History command calls emxPageHistory.jsp, which lists the last 50 pages the user has visited since the user logged in. The number of pages stored by the page is configurable using a property in emxSystem.properties. Users can use the page to revisit the pages, bookmark them, and copy and paste the URLs for the pages into emails to invite other users to visit the pages. The page displays in a non-modal popup window.

For information about how to use the page, click the Help button on the page in the application.

Page History

Refresh

?

Link Name	Suite/Category	Address (URL)
Workspace 100		http://hostname/ematri
Workspaces	Team	http://hostname/ematri
Search	Sourcing	http://hostname/ematri
Comp1 Package 1	Attachment	http://hostname/ematri
Comp1 Package 1		http://hostname/ematri
Packages	Sourcing	http://hostname/ematri
Comp1 RFQ 17 1	Supplier	http://hostname/ematri
Comp1 RFQ 17 1	Line Item	http://hostname/ematri
Comp1 RFQ 17 1		http://hostname/ematri
RFQ's	Sourcing	http://hostname/ematri

To create a Favorite or Bookmark:

Internet Explorer (Windows)
 Right-click a link and select "Add to Favorites".

Netscape (Windows)
 Right-click a link and select "Add Bookmark".

Other platforms
The procedures will be similar but the mouse-click actions may vary.

 Close

The tables below describe the parameters and settings used to implement the Page History command. For details on configuring global toolbar tools, see [Building Menus and the Global Toolbar](#).

Parameter	Description	Installed Input Values
href	The JSP that implements the Page History page.	emxPageHistory.jsp
Alt	The internationalization string ID to display as mouse over text.	emxFramework.PageHistory.P ageHistoryLabel
Access	Defines who can access the tool.	All

Setting	Description	Installed Input Values
Help Marker	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help .	emxhelppagehistory
*Registered Suite	The application the command belongs to. The system looks for image and JSP files in the directory associated with the application. Internationalization details are also obtained based on the Registered Suite.	Framework
Target Location	The frame or window the page should display in. By default, the page displays in a separate popup window.	popup
Popup Modal	The page is non-modal so the page remains open while users continue to visit pages.	false
Window Height	The window height for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	550
Window Width	The window width for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	550
Image	The image displayed on the tool.	ButtonToolbarPageHistory.gif
*Required Setting		

Implementing Context-Sensitive Help

Each MatrixOne application has its own help system and the framework also has a help system for its pages, called the MatrixOne Application Help. All help systems are installed in `ematrix/doc/APP_NAME/LANG`. For example, the English help for Team Central is in `ematrix/doc/teamcentral/en`. The name of the directory for the MatrixOne Application Help is called `common`, so the English help is for it is in `ematrix/doc/common/en`.

To open a help system, this file must be called from within the `ematrix/doc/APP_NAME/LANG` directory. The same file name is called for all application help systems.

When you call the `frames.htm` file, the help system opens to its title page. To call a particular topic within the help system, you must pass the help marker for that topic. (Placing help markers into the help system is part of building the help files and beyond the scope of this document. To get the help marker that should be called for an application page, consult the person responsible for creating the help files.)

Use the following steps to call a particular help topic for a configurable page. This procedure assumes the help system has the directory structure and `frames.htm` file described above and that the corresponding help marker has been included in the help system.

To call context-sensitive help for a configurable page

1. Make sure the command object that calls the configurable page, such as `emxTables.jsp` or `emxForms.jsp`, has the setting:

`Registered Suite=APPLICATION`

Where `APPLICATION` is the same as the suite name defined in the key `eServiceSuites.DisplayedSuites` within `emxSystem.properties`. The “`eServiceSuite`” prefix, if any, can be left off.

Note that even if there is no help marker defined, as described in the next step, when the configurable page’s Help button is clicked, the appropriate application help system is still called based on the Registered Suite. The help system opens to the title page.

2. Pass the help marker to the configurable page in either of these ways:
 - In the command object that calls the configurable page, add the setting:

`Help Marker=MARKER`

Where `MARKER` is the help marker for the help topic to call, for example, `emxhelproucreate`. The command will automatically append the `HelpMarker` parameter to the URL for the configurable page.

- In the `href` parameter for the command object that calls the configurable page, manually enter the parameter:

`HelpMarker=MARKER`

Here’s an example of a URL for a configurable table that includes the help marker:

```
${COMMON_DIR}/  
emxTable.jsp?inquiry=PartList,PTParts,CMParts&inquiryLabel=emxE  
ngineeringCentral.Common.All,PT Parts,CM  
Parts&table=DemoPartTable&header=emxEngineeringCentral.Common.P  
arts&toolbar=DemoPartToolbar&TipPage=emxBlank.jsp&sortColumnNam  
e=Name&FilterFramePage2=${COMMON_DIR}/
```

```
emxTableFilterIncludeSample.jsp&FilterFrameSize=40&selection=mu  
ltiple&SubmitURL=${SUITE_DIR}/  
emxBlank.jsp&SubmitLabel=emxEngineeringCentral.Button.Next&Canc  
elButton=true&CancelLabel=emxEngineeringCentral.Common.Cancel&s  
ubHeader=Bill of Material Level  
1&pagination=8&headerRepeat=12&rememberSelection=true&HelpMarke  
r=emxhelppartlist
```

Lifecycle Help Markers

Lifecycle help markers are called based on the name of the policy that controls the object's lifecycle. The convention for naming lifecycle help markers is "emxhelppolicy" followed by an underscore and then the policy name. For example, the policy that controls most parts is the EC Part policy. The help marker is emxhelppolicy_ecpart.

JSP Programming

Use Built-In Parameters to Make Pages Reusable

The dynamic user interface components automatically append a number of parameters to URLs. For example, the parameters `emxSuiteDirectory`, `objectId`, and `ParentOID` are all passed automatically. For a list of automatically-appended parameters, see [Parameters Automatically Passed to URLs](#) in Appendix B.

It is important that when JSPs are written to be pop-ups (even if they are called from “custom” pages initially) that they use these automatically-appended URL parameters and do not invent new ones. That way, when you call a custom page from one of the configurable pages (`emxTable.jsp`, `emxTree.jsp`), you won't have to make changes.

It's also important that you don't pass in data that you can get from the automatically-appended parameters. Passing extra data that is “not necessary” creates pages that can't be used from a configurable page.

Standard Error Handling

This section describes the error handling mechanisms used for the configurable components.

Whenever an exception or context-specific error happens in a JSP, an error message displays as a “JavaScript” popup alert message box. If there are multiple errors within the page, all error messages are collected by the error object and displayed to user.



To achieve this, an Error Object class called “FrameworkException” is instantiated for every JSP. This class is part of the “com.matrixone.framework.util” package. The error object instance name is `emxNavErrorObject`. The instantiation of the object is done at the top include file, `emxNavigatorTopErrorInclude.jsp`. The error object is used in the processing page for adding the error messages and finally used at the bottom include page, `emxNavigatorBottomErrorInclude.jsp`, for displaying any error message. This bottom include page also has the MQL Notice Include file, `emxMQLNotice.jsp`, which alerts any MQL error message that occurs during the processing.

The Error object “`emxNavErrorObject`” has the request scope, so the new error object is available to every request. Hence the object is destroyed from memory after displaying the page.

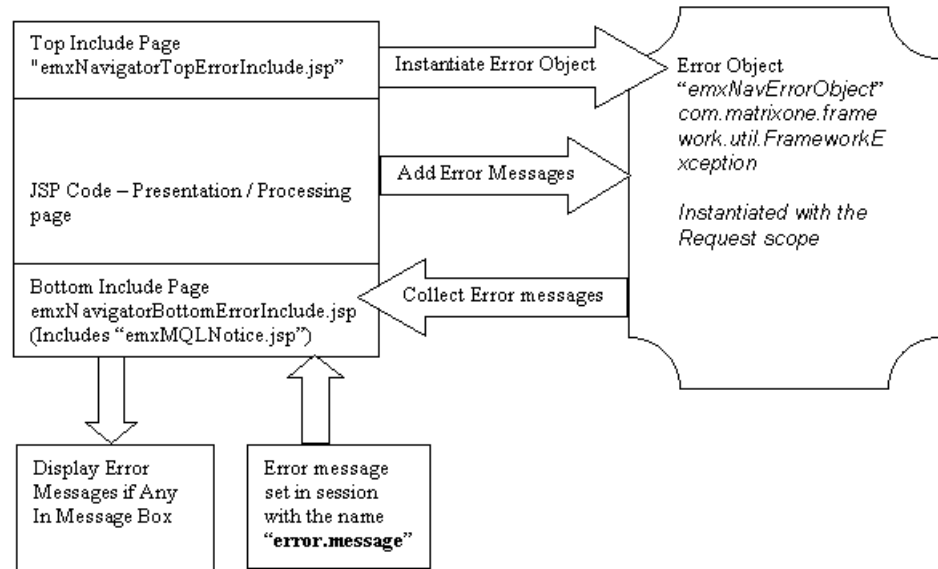
When the error/exception occurs, the page can call the `emxNavErrorObject` object method “`addMessage()`” (on the class `FrameworkException`).

```
emxNavErrorObject.addMessage("ERROR TEXT MESSAGE...");
```

Multiple error messages can be added to the error object and it maintains a list of messages within the error object. At the end of the page, the list of error messages is collected from the error object and displayed to the user in a single message box.

JSP Layout for Error Handling

Every JSP page that implements this error handling approach should have the following page layout.



Implementing Error Handling in JSP Pages that are in the Same Request Scope

Every JSP page that implements error handling with the “emxNavErrorObject” will have to include the Top and Bottom Error Include files given below:

- emxNavigatorTopErrorInclude.jsp
- emxNavigatorBottomErrorInclude.jsp

All the processing Java code within the JSP page (including jsp:forward and jsp:include) will be in the try.. catch.. finally block. If an error occurs in the processing code, an appropriate error message can be added to the error object using the addMessage() method. The addMessage method can be called within the page just after getting the error message or within the finally block.

The following sample skeleton code shows the syntax and method for implementing the error message. “emxNavErrorObject” is the instance name of the error handling class FrameworkException, which is instantiated in the Top Include.

```
// JSP code...
String emxErrorString = "";

try {
    .....
    .....
    // Access check
    ....
    ....
}
```



```

emxNavErrorObject.addMessage("User does not have access to this
object..");

.....

.....

} catch ( Exception e) {
.....
.....

    emxErrorString = (ex.toString()).trim();

} finally {

emxNavErrorObject.addMessage(emxErrorString);

}

```

Error Handling in JSP pages that are Pure Processing and Not in the Same Request Scope

If the error is occurring in a pure processing page, which is not within the same request scope of the referring page, the following procedure can be followed to implement error handling.

When there is an error message in the processing page that needs to be displayed to the user, set a session attribute with name “error.message” with the value assigned to the ERROR TEXT to be displayed. For example:

```

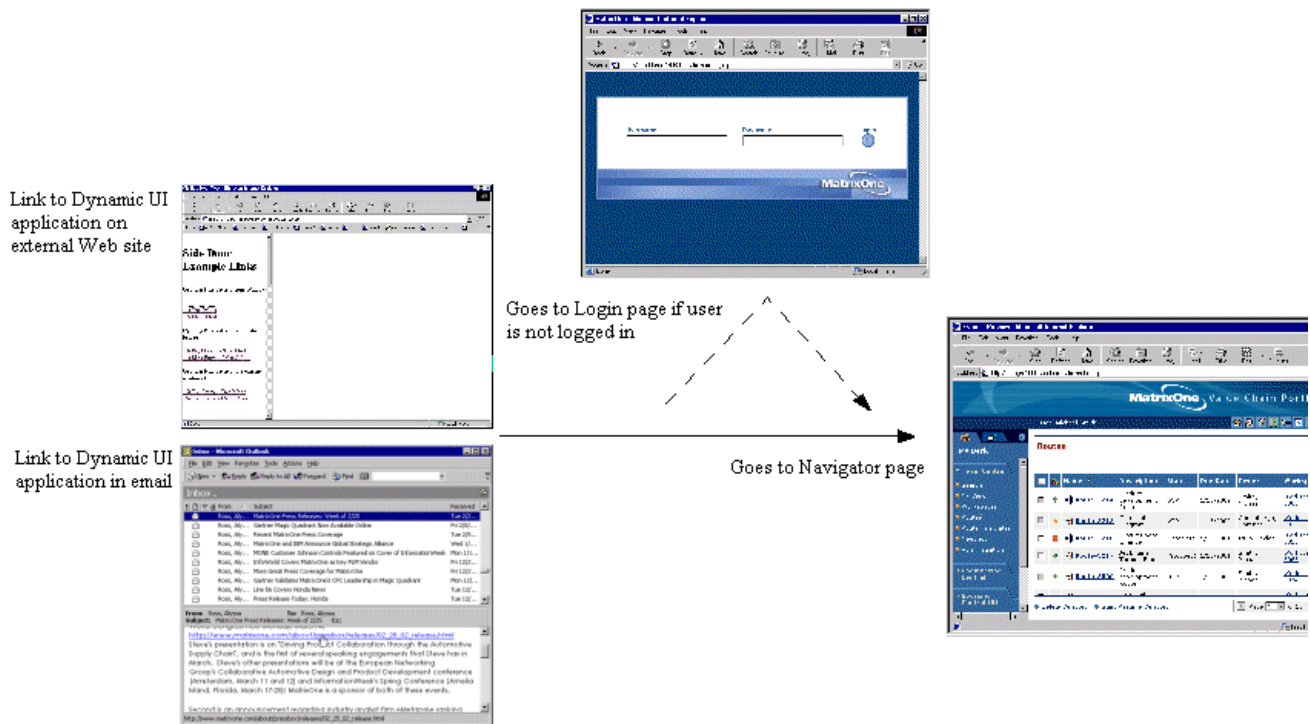
session..setAttribute("ERROR TEXT MESSAGE....");

```

If the JSP that displays subsequent to the processing page has implemented the error handling approach with “emxNavErrorObject”, the session error message set in the session displays. This subsequent JSP page must follow the JSP layout given in the [JSP Layout for Error Handling](#) section.

Accessing the Applications Externally

You can configure links on external Web applications and email messages that let people access applications based on the dynamic user interface. The link must call the main Navigator page for the dynamic user interface, `emxNavigator.jsp`. When a person clicks the link and is already logged in (and there is still a valid session context), the system launches the Navigator page directly. If the person is not logged in, the system presents the Login page. After submitting a successful login, the Navigator page displays.



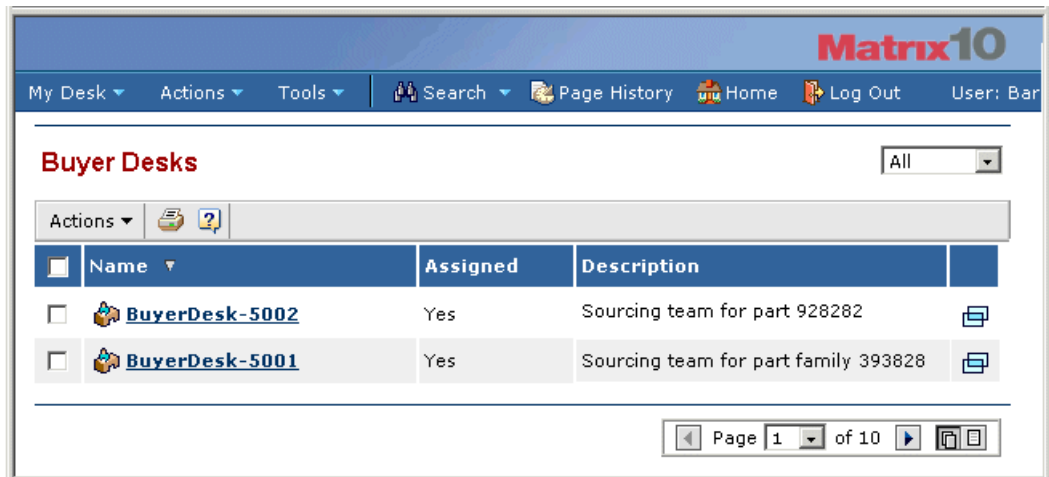
This section describes how to configure the link so the Navigator page displays the way you want it to.

Specifying the Content of the Navigator Page

The Navigator page can appear in one of two modes: Menu or Tree. You configure the mode using the mode parameter appended to the URL for the Navigator page. The mode parameter accepts either Menu or Tree as its value. The rest of this section describes these modes and configuration options for them.

Menu Mode

When the Navigator page is in Menu mode, it contains the banner, global toolbar, My Desk menu, Actions menu and Tools menu as it normally does when using the applications. The content frame can contain any application page, such as a table page that lists business objects or a navigation tree and Properties page for one business object. This graphic shows the Navigator page in Menu mode with a table page in the content frame.

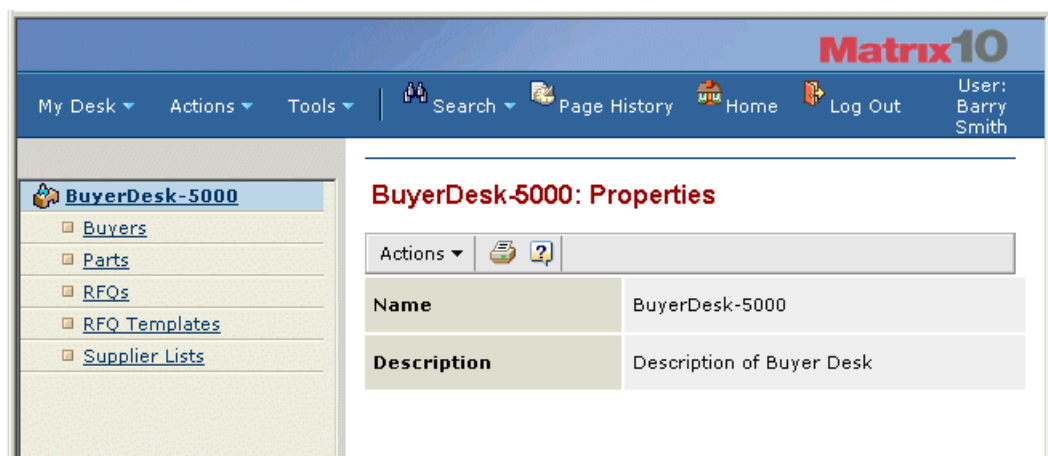


To display the Navigator page in Menu mode with any page other than a navigation tree for an object, you specify the page using the `ContentPage` parameter. For instructions on specifying the content of the Navigator page, see [Configuring the Default Home Page for the Content Frame](#). For example, if the Buyer Desk table page is `emxBuyerDeskTable.jsp`, the following URL would call the page using the Menu mode.

```
./ematrix/common/emxNavigator.jsp?mode=Menu&ContentPage=../sourcingcentral/emxBuyerDeskTable.jsp
```

Menu mode is the default mode for the Navigator page so you do not have to specify the parameter if you want Menu mode.

To display the Navigator page in Menu mode with the tree and Properties page for a business object in the content frame, include the `objectId` parameter. For example, to call this page:

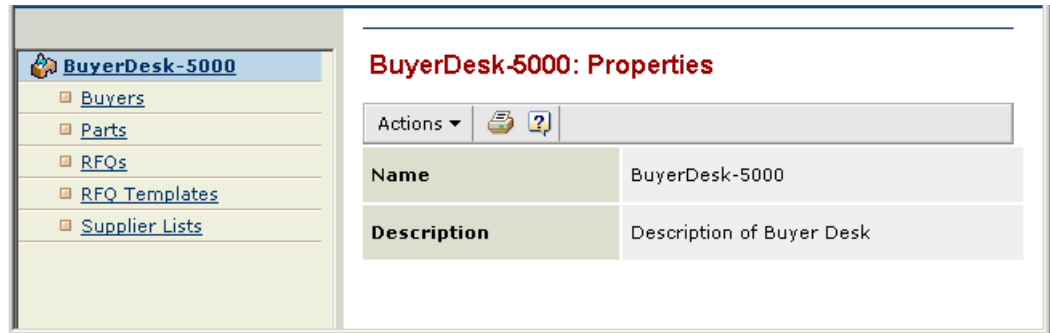


Use this URL:

```
./ematrix/common/emxNavigator.jsp?objectId=53028.50507.27254.238&mode=Menu
```

Tree Mode

When the Navigator page is in Tree mode, it does not contain the banner, global toolbar, My Desk menu, Actions menu, or Tools menu and the content is *always* the tree for a business object. Use Tree mode when you want to let the user see details for one business object without providing access to other features or applications. Because the Tree mode always displays details for a business object, the `objectId` parameter must be included whenever the Tree mode is used. The page will error if you don't include the `objectId` when using the Tree mode. For example, to show this page:



Use this URL:

```
./ematrix/common/  
emxNavigator.jsp?objectId=53028.50507.27254.238&mode=Tree
```

Specifying the Location of the Navigator Page

Like any URL, when the Navigator page is called from a link on a Web page or email message, you use the target href tag to specify where the page should appear: in a new window, in another frame in the current window, or in the current frame.

Portal Parameter

When opening the Navigator page in a different frame or window, use the portal parameter to make sure the Login page doesn't replace the frame that contains the link. When the Navigator page is called from within the MatrixOne applications and the Login page is required (for example, maybe the system has timed out and a login is required again), the system replaces the current window with the Login page. But if you are calling the Navigator page from an external Web site and want the page to appear in another frame or window, you would typically want the Login page to open in that other frame or window and not within the current frame or window (not within the frame that contains the link).

To ensure the `emxNavigator.jsp` has no effect on the frame with the link that calls the page, append the portal parameter to `emxNavigator.jsp` and assign the value of `true`. If the portal parameter is not included, it is assumed to be `false`. For example:

```
./ematrix/common/emxNavigator.jsp?portal=true
```

Opening the Navigator in Current Window or Frame

Additionally, to open the Navigator page in a frame within the external Web or portal page that includes the link, include the following bolded code at the top-level page of the implementing external JSP page.

```

//*****
<html>
<%
// JSP code here...
%>

<head>
// include START

    <script language="javascript" src="scripts/
emxUIConstants.js"></script>
    <script language="javascript" src="scripts/emxUIObjMgr.js"></
script>
    <script language="javascript" src="scripts/emxUINavbar.js"></
script>
    <script language="javascript" src="scripts/emxUIModal.js"></
script>

    <script language="javascript" src="scripts/emxUITree.js"></
script>
    <script> var tempTree = new jsTree("emxUITree.css"); </script>

// include END

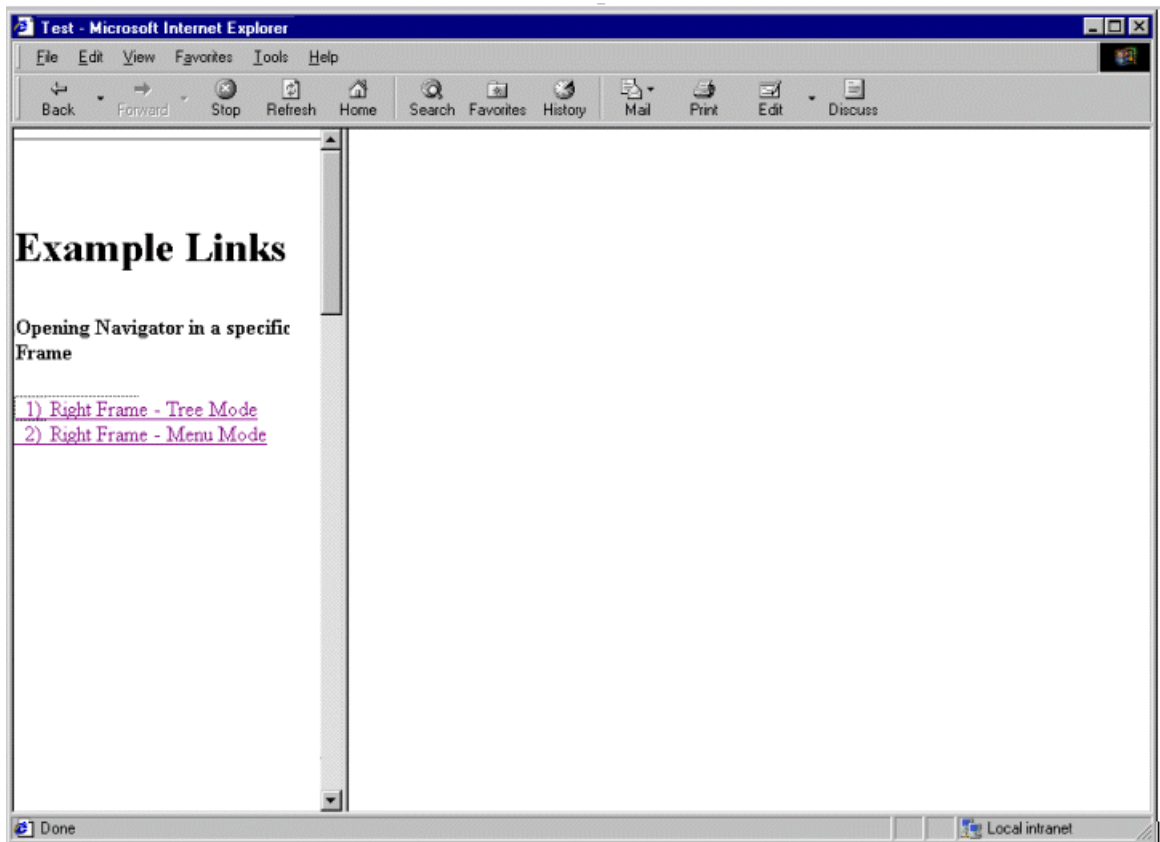
</head>

<%
//JSP code here...
%>
//
*****

```

Examples

For example, suppose you have an external Web page with links to the Navigator page in the left frame, as shown below. Link 1 displays the Navigator page in Tree mode and Link 2 displays it in Menu mode.

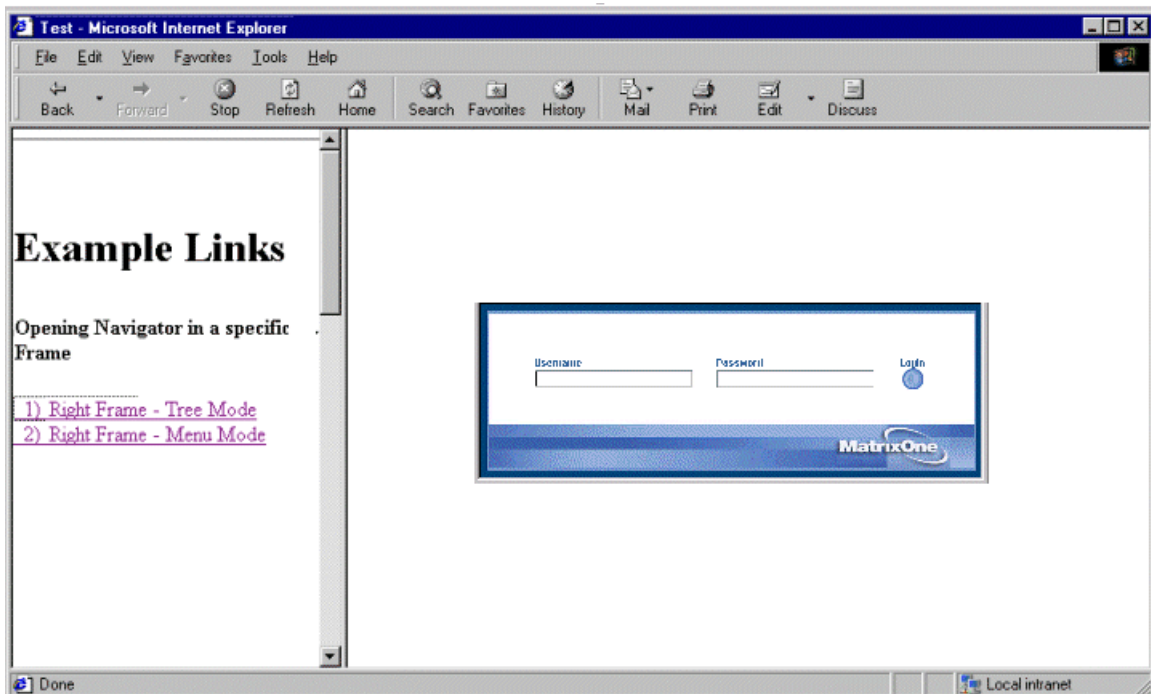


When a link is clicked, you want the Navigator page to display in the right frame, which is named rightFrame. The href tag for Link 1 would look like this:

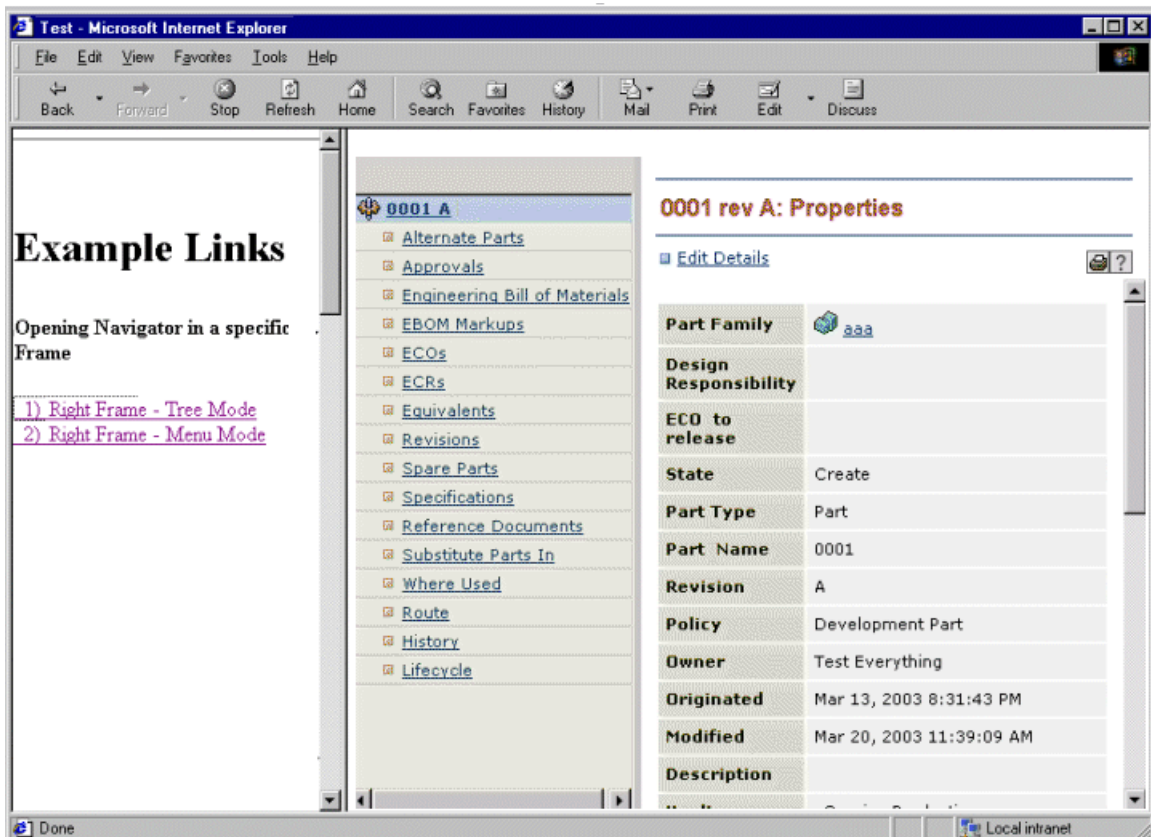
```
<a  
  href="emxNavigator.jsp?objectId=53028.50507.27254.238&mode=Tree  
    &portal=true" target="rightFrame" > 1) Right Frame - Tree Mode  
</a>
```

Since the Tree mode is specified, the objectId parameter is required so the system knows which business object to show details for. The portal parameter is included because the page shouldn't affect the current frame, the left frame.

If a user clicks Link 1 and isn't logged in, the Login page would display in the right frame:



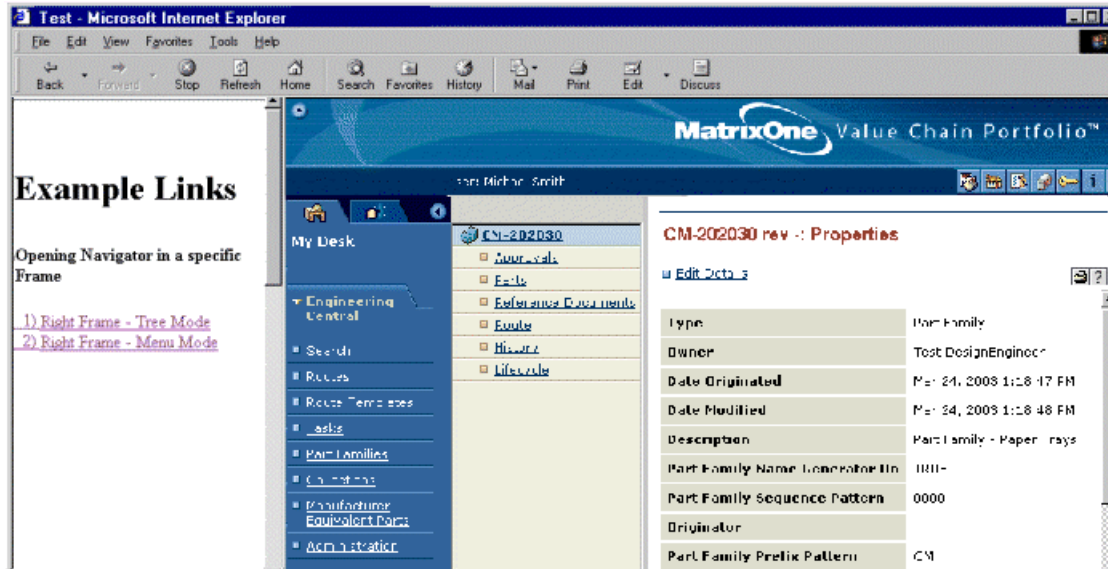
After logging in, the Navigator page opens with the tree and Properties page for the object. Because Tree mode is passed, the page doesn't contain the banner, tabs, or global toolbar.



If Link 2 is configured to display details for the same object but using Menu mode, the href would look like this:

```
<a  
  href="emxNavigator.jsp?objectId=53028.50507.27254.238&mode=Menu  
  &portal=true" target="rightFrame" > 2) Right Frame - Menu Mode  
</a>
```

Assuming the user is logged in, the resulting page would look similar to this:



A

Appendix A: Triggers and Programs

Utility Trigger Programs

This section describes the trigger programs that are used by many MatrixOne applications, which we'll call utility trigger programs. These programs are installed with the framework. For each program, the section lists the purpose of the program, accepted argument inputs, processing details, and returns. The procedure for specifying the input values is the same for all trigger programs and is described in [Configuring Automatic Business Rules \(Triggers\)](#) in Chapter 5.

To see which trigger programs are used by an application and to find out which administrative object contains the trigger, see the Administrator's Guide for the application. The Administrator's Guides also describe the trigger programs that are specific for the application.

Make sure you use the symbolic name for any administrative objects that you specify as an argument input. For example, if you need to specify a relationship for an argument input, use the symbolic name for the relationship and not the standard name (relationship_GBOM instead of GBOM).

Previous Revision Promotion

The trigger program `eServicecommonPreviousRevisionPromotion_if.tcl` is an action trigger for a promote event. The program promotes to a specified state the previous revision of the object that is being promoted. This ensures that when an object is promoted to a particular state, its previous revision is also promoted. If the program encounters a trigger attached to the state to which the revision is getting promoted, the program presents an error message.

Arguments

Argument Number	Argument Name	Description and Inputs
1	sState	The state to which the program should promote the previous revision. The program gets the state names from the policy that governs the object that is being promoted. The program accepts three values for this argument: • <code>next</code>—Indicates the previous revision for the object has to be promoted to its next state regardless of the current state of the object being promoted. • <code>last</code>—Indicates the previous revision has to be promoted to its last state in the lifecycle. • <code>[specific state]</code>—Indicates the revision has to be promoted to that state. If a null is passed as the input, the program takes the current state of the object being promoted as the state to which the revision should be promoted.

Returns

0 if the previous revision is successfully promoted; otherwise 1 and an error message.

Required Connection Check

This trigger program `eServicecommonTrigcRequiredConnection_if.tcl` checks to see if an object has specific connections. The following table shows the kinds of information the program can check for.

Does the selected object have any:	For example:
Connections with other objects.	Does the object have any relationships?
Connections with other objects in one direction.	Does the object have any relationships in which the object is on the From end?
Connections with other objects through specified relationships.	Does the object have any connections to other objects with the Includes Part relationship.
Relationships with objects of specified types.	Does the object have any connections to a Part object?

You can combine the checks listed above. For example, you can have the program check if the object has a connection in which the object is on the From end of any Supplier Response relationship and in which the connected object is an Initial Quotation type.

Arguments

Argument Number	Argument Name	Description and Inputs
1	Direction	The direction(s) the program should search from the selected object to connected objects. The program accepts three values: - from—Search only connections in which the selected object is on the From side of the relationship. For example, suppose three objects are connected like this: Object 1 --> Object 2 --> Object 3 If the direction input is “from” and Object 2 is the selected object, then the program only considers the connection between Objects 2 and 3. - to—Search only connections in which the selected object is on the to side of the relationship. “(blank)”—Search all connections, regardless of the direction.
2	RelationList	The relationships the program should check for, formatted as a comma-delimited list of symbolic names for the relationships. For example, to check for the Includes Part and Supplier Response relationships, the input would be: relationship_IncludesPart,relationship_SupplierResponse
3	TypeList	The types the program should check for, formatted as a comma-delimited list of symbolic names for the types. For example, to check for connections with Initial Quotation and Final Quotation objects, the input would be: type_InitialQuotation,type_FinalQuotation

This table shows how to configure the argument inputs depending on the kind of check you want to perform.

To check for:	Use these inputs:		
	Direction	RelationList	TypeList
Any connection	blank	blank	blank
Any connection in specified direction	To or From	blank	blank
Any connection to objects of specified type	To, From, blank	blank	specified
Any connection of specified relationship	To, From, blank	specified	blank
Any connection of specified relationship to objects of specified type	To, From, blank	specified	specified

Returns

0 if at least one specified connection exists. Otherwise the program returns 1 along with an error message indicating no connection was found and giving details of the specified connection.

Required File Check

The program `eServicecommonTrigcRequiredFormat_if.tcl` can check to see if any files are checked in to the:

- Selected object
- Selected object for the default format
- Selected object for a specified format

Arguments

Argument Number	Argument Name	Description and Inputs
1	sSearchKey	Specifies the format(s) to check for. The argument accepts three values: • list Check formats listed in IFormatPropertyList argument. • default Check only default format of selected object. • all Check all the formats supported by selected object.
2	IFormatPropertyList	This argument has meaning only if sSearchKey is set to "list". The argument accepts a list of the symbolic names for of the formats to check for. For example, {format_Word format_generic ...}.
3	bSizeCheck	Checks whether the size of checked in files is 0 bytes. The argument accepts two values: • True The program checks the size of all the checked files (of the format specified by the sSearchKey argument) and fails if all the checked in files are 0 bytes. • False The program does not check the file sizes.

Processing

If the input for sSearchKey is:	The program:
list	Searches for checked files in all the formats listed in argument IFormatPropertyList. If any format is not supported by the selected object, the program ignores that format.
default	Searches for checked in files in only the default format for the selected object.
all	Searches for checked in files in all the supported formats of selected object.

Returns

If bSizeCheck is False and:

- The program finds a checked in file of the specified format, it returns 0 for success.
- The program does not find a checked in file of the specified format, it returns 1 along with an MQL message that explains the reason for failure.

If bSizeCheck is True and:

- The program finds a checked in file of the specified format and the checked in files are not 0 bytes, it returns 0 for success.
- The program does not find a checked in file of the specified format or the checked in files are all 0 bytes, it returns 1 along with an MQL message that explains the reason for failure.

Check Relative State

The trigger program eServicecommonCheckRelState_if.tcl is a check trigger for a promote event. It searches a specified relationship looking for a specified object type and determines if all of the objects found are at the minimum acceptable state that has been specified.

Arguments

Argument Number	Argument Name	Description and Inputs
1	sRelationship	The relationship type to search for. If the argument input is null, the program checks all the relationships in the direction specified for sDirection. If the specified relationship is not assigned to the target object type, the program returns a FAIL.
2	sTargetObject	The object type to search for at the other end of the relationship. If null, the program checks all the objects that can be reached through the relationship specified for sRelationship.

Argument Number	Argument Name	Description and Inputs
3	sTargetState	The state the target objects should be in. If the specified state is not a valid state defined for the target object, then the program returns a FAIL.
4	sDirection	The direction to search for. The program accepts three values: TO, FROM, BOTH. If null, the program searches in the FROM direction.
5	sComparisonOperator	The operator the program should use for checking the target state. The values are: - LT (Less Than)—The program returns a FAIL if the target object is in a state greater than the target state. - GT (Greater Than)—The program returns a FAIL if the target object is in a state less than the specified state. - EQ (Equal)—The program returns a FAIL if the target object is in a state other than the specified state. - LE (Less Than Or Equal)—The program returns a FAIL if the target object is in a state greater than or equal to the specified state. - GE (Greater Than Or Equal)—The program returns a FAIL if the target object is in a state less than or equal to the specified state. - NE (Not Equal)—The program returns a FAIL if the target object is in the specified state. If null, the program uses EQ.
6	sObjectRequirement	A flag that specifies whether the object is required or optional. The values are: Required and Optional. If null, the program uses Optional.

Returns

Pass if all objects are in the minimum required state. Otherwise, the program returns fail. The specific conditions for passing and failing depend on the Comparison Operator.

For example, suppose the comparison operator is LT (less than) and the target object has states defined as A, B, and C.

If the target state is:	And the current state is:	The program returns:
A	A, B, C	fail
B	A	pass
B	B, C	fail
C	A, B	pass
C	C	fail

Now suppose the comparison operator is EQ (equals).

If the target state is:	And the current state is:	The program returns:
A	A	pass
A	B, C	fail
B	A, C	fail

If the target state is:	And the current state is:	The program returns:
B	B	pass
C	C	pass
C	A, B	fail

Valid Revision Sequence

The program eServiceValidRevisionChange_if.tcl is a check trigger for a create event. This program ensures the revision sequence is maintained for a business object by not allowing an object to be created if there is an existing object of the same type and name and of any revision.

Arguments

The program accepts no arguments.

Returns

If the program finds no object of the same type and name (with any revision), it returns zero and the object is created. If the program finds an object of the same type and name with any revision, the program returns a non-zero value, prevents the create event, and shows a message telling the user to use Object>New>Revision instead of Object>New>Original.

Relative Float Action

The program eServicecommonRelativeFloatAction_if.tcl is a promote action trigger. When an object is promoted, the program gets the previous revision of the object and floats all the specified relationships in the specified direction.

Arguments

Argument Number	Argument Name	Description and Inputs
1	sRel	The relationship to float. Must be the symbolic name (for example, relationship_PartRule).
2	sDirection	The relationship direction to float. Accepts three values: from, to and both.
3	sOperation	A flag that specifies which revision of the object to look for. It accepts two values: PREVIOUS, LATEST
4	sState	If argument 3 is LATEST, this argument should specify the state in which the latest revision should be. It is passed in as the symbolic name for a state (for example, state_Release).

Processing

Suppose there are four versions for an object A: A1, A2, A3 and A4. The trigger gets fired when A4 is promoted to its next state. If the third argument input is LATEST, the trigger program finds the LATEST revision for A that is in the state specified by the fourth argument. If the third argument is PREVIOUS, the program gets the previous revision, which is A3. It then traverses through the relationship specified in argument 1 in the direction specified by argument 2 and gets all the connected objects, then connects them to A4.

Set Originator Attribute

The Java Program Object `emxcommonSetOriginator_if` is an action program for create and revise events. The program sets the Originator attribute value to the name of the current user. The program looks up the actual Originator attribute name because the customer could change name of the attribute.

When the trigger program runs, it retrieves the necessary environment variables, such as TYPE, NAME, REVISION, VAULT, USER, EVENT, NEWREV, and APPREALUSER. Because creation is often done as the Shadow Agent person, the program checks the environment variable APPREALUSER. If set, it is used instead of USER for setting the originator.

Two eService Trigger Program Parameters objects that call this trigger program are installed with the framework. These objects call the program whenever any type is created or modified. See [Automation for All Types](#).

Arguments

Argument Number	Argument Name	Description and Inputs
1	attrName	The symbolic name for the attribute that is to be populated.

Returns

0 for success; non-zero for failure.

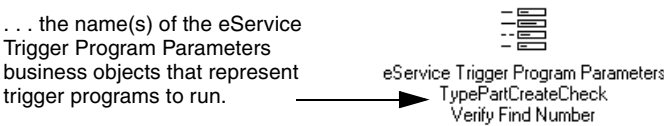
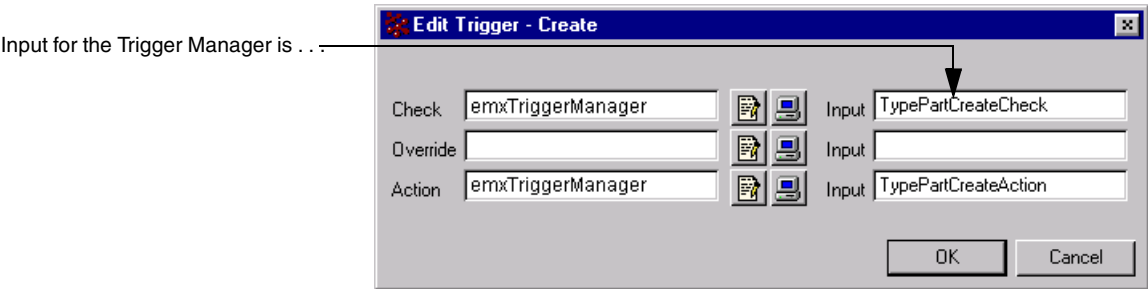
Automated Processes

This section lists automated processes installed with the framework. Most automation is installed with each application or with common components. For a list of triggers installed with an application, see the Administrator’s Guide for that application. For a list of triggers installed with common components, see the *Common Component Administrator Guide*.

All trigger programs are run using a Trigger Manager program called emxTriggerManager. This Java program is specified as the Action for all triggers. The name of the eService Trigger Program Parameters business object that represents the specific trigger program to run is specified as the Input to pass to the Trigger Manager. The naming convention for the eService Trigger Program Parameters object indicates the schema object the trigger is associated with, trigger event, and the type of trigger.

[ADMIN OBJECT TYPE][ADMIN OBJECT NAME][TRIGGER EVENT][TYPE OF TRIGGER]

For example, an eService Trigger Program Parameters object named “TypePartCreateCheck” indicates that the trigger is a Create Check trigger on the Part type. The revision indicates that the purpose of the trigger is to verify the find number for the part.



Automation for All Types

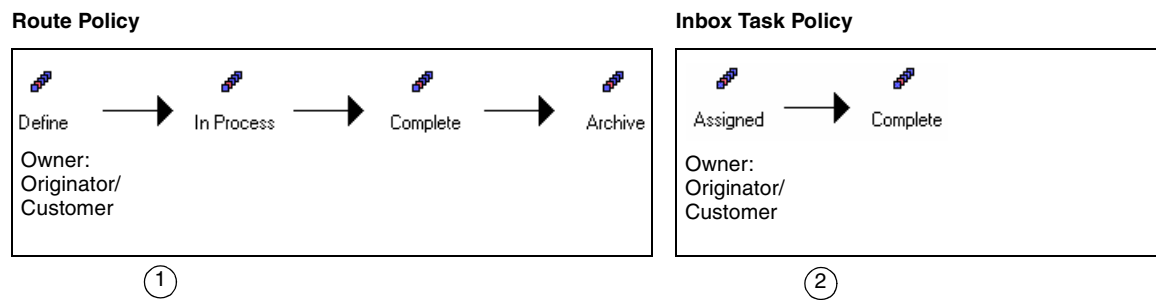
This diagram shows two triggers that are added for almost every type in the framework.

eService Trigger Program Parameters Object, Name and Revision	Trigger Program and JPO Method	Description of Action
TypeAllCreateAction PopulateOriginator	emxTriggerManager, which calls emxcommonSetOriginator_if For information on this trigger program, see the Set Originator Attribute .	Store user’s name in Originator attribute

eService Trigger Program Parameters Object, Name and Revision	Trigger Program and JPO Method	Description of Action
TypeAllReviseAction PopulateOriginator	emxTriggerManager, which calls emxcommonSetOriginator_if For information on this trigger program, see the Set Originator Attribute .	Store user's name in Originator attribute

Automation for Routes and Inbox Tasks

This section describes the triggers and programs that are executed throughout the lifecycles for routes and tasks. These programs are responsible for creating pending tasks for people who are in the route and sending notifications to the route owner and route members as the route progresses.



For email notifications to work correctly, both the sender's and recipient's emails must be entered in the users' Person objects.

1 Initiate Route

When a route is started, it is promoted from Define to In Process state. This fires a trigger, which:

- Creates an Inbox Task object for each person in the route who has an order number of 1.

Details:

Expand Route Node relationship and get all relationships with Route Sequence = 1

For each node from above:

Create an Inbox Task

Attach the Inbox Task (via Project Task relationship) to the Person the Route Node is attached to

Copy the following attribute data from the Route Node relationship to the Inbox Task:

Route Action

Route Instructions

Approval Status

Schedule Completion Date

Approvers Responsibility

Attach Inbox Task to Route (via Route Task relationship)

Attach Inbox Task to the Project Member (via Member Task) that the Route is attached to via Member Route (this

identifies the creator of the route to which the task belongs)

Copy 'Route Node Id' attribute of Route Node relationship to that on Inbox Task

- Emails each person who has a pending task.
- Changes the Route Status attribute on the Route from Not Started to Started.
- Sets Current Route Node attribute on Route to 1.

2 Complete Task

When a route member completes a task, the task is promoted from Assigned to Complete and a trigger fires, which:

- Populates the Actual Completion Date attribute with the current date.
- Copies data from the following attributes on Inbox Task to the Route Node relationship (this is necessary because the Inbox Task object is deleted when the route is complete).
 - Approval Status
 - Actual Completion Date
 - Comments
- If the approval status is Reject, emails the route owner and changes the Route Status to Stopped. No further tasks are created for the route when a route is Stopped.
- If the approval status is Started and there are no other tasks in the Assigned state for the route (if there are other tasks still in Assigned state, then nothing should be done because not all tasks for this order level are complete), creates an Inbox Task for each person who is to receive the folder and who has the next order number. Enters the usual data (action, instructions, status, etc.). Notifies each person. Increments Current Route Node attribute on attached Route object.

Details:

Expand the attached Route and get all Inbox Task objects whose current state == Assigned

If none of the Inbox Task objects are returned and Route Status == Started:

Increment Current Route Node attribute on attached Route object

Expand Route Node relationship and get all Relationship Ids whose Route Sequence == Current Route Node value

If the number of Route Node relationships from above > 0, for each node from above:

Create an Inbox Task

Attach the Inbox Task (via Project Task relationship) to the Person the Route Node is attached to

Copy the following attribute data from Route Node relationship to Inbox Task

Route Action

Route Instructions

Approval Status

Schedule Completion Date

Approvers Responsibility

Attach Inbox Task to Route (via Route Task relationship)

Attach Inbox Task to the Project Member (via Member Task) that the Route is attached to via Member Route (this identifies the creator of the route to which the task belongs)

Copy 'Route Node Id' attribute of Route Node relationship to that on Inbox Task

If processing rule for Inbox Tasks is set to Any then program delete all the rest of the tasks in same sequence and also disconnects all the Persons associated with that task from Route.

If there are no other people to be routed to, promote the route to Complete and change the route status to Finished.

3 Complete Route

When a route is promoted to Complete, another trigger fires, which:

- Sets the Route Status attribute on the Route to Finished.
- Performs the operation indicated in the Route Completion Action attribute on the Route:

Notify the Route object owner

Or:

Promote the object that is attached to the Route via the Object Route relationship

Automation for Routes

This table lists the automation related to routes that is installed with the framework.

eService Trigger Program Parameters Object, Name and Revision	Trigger Program and JPO Method	Description of Action
N: PolicyRouteStateDefinePromoteAction R: Initiate Route	eServicecommonTrigcInitiateRoute_if.tcl	When a route is started, it is promoted from Define to In Process and this trigger is fired. The trigger: – Creates an Inbox Task object for each person who has an Order number of 1 in the route. Copies all the information for each task (instructions, action, scheduled completion date, etc.) to the Inbox Task object. – Notifies each person via email.
N: RelationshipRouteNodeCreateAction R: emxRouteJPO	calls emxRoute grantAccess method	When a person is added to a route, this trigger grants the person access to the route (Read Write).
N: RelationshipRouteNodeDeleteAction R: emxRouteJPO	emxRoute deleteRouteTaskUserObject method	When a person is removed from a route, this trigger deletes the person's task.
N: RelationshipRouteNodeDeleteCheck R: emxRouteJPO	emxRoute revokeAccess method	When a person is removed from a route, this trigger revokes the person's access to the route.
N: RelationshipRouteScopeCreateAction R: emxRouteJPO	emxRoute inheritAccess method	When a route and the Route Scope relationship is created, this trigger grants Read access to the route for all Workspace Leads.

Automation for Inbox Tasks

This table lists the framework automation related to Inbox Tasks, which are the tasks created for routes.

eService Trigger Program Parameters Object, Name and Revision	Trigger Program and JPO Method	Description of Action
N: PolicyInboxTaskStateAssignedPromoteAction R: Complete Task	eServicecommonTrigCompleteTask_if.tcl	When a person completes a task, the Inbox Task object is promoted from Assigned to Complete and this trigger is fired. The trigger: – Enters the actual completion date. – If the approval status is Reject, stops the route and notifies the route owner. – If there are no other tasks to be completed for the route, notifies the route owner that route is complete and changes the Route status to Finished.

Automation for Organizations

This table lists the automation related to organizations business objects. The Organization policy governs Company and Business Unit objects. Whenever the term “company” is used in the automation descriptions below, the information also applies to business units.

eService Trigger Program Parameters Object, Name and Revision	Trigger Program and JPO Method	Description of Action
N: RelationshipCollaborationRequestCreateAction R: Notify Collaboration	eServicecommonTrigaNotifyCollaboration_if.tcl	When a Company Representative requests to collaborate with another company, the system connects the companies using the Collaboration Request relationship. When this relationship is created, the system fires a trigger which sends an email to: – all Company Representatives for the company that is being invited to collaborate; the email says there is a collaboration request from the requesting company – all Company Representatives for the company making the request; the email confirms that an email has been sent to the other company
N: RelationshipCollaborationRequestDeleteAction R: Notify Collaboration	eServicecommonTrigaNotifyCollaboration_if.tcl	When a Company Representative withdraws a request to collaborate with another company or the invited company rejects the invitation, the system sends an email to Company Representatives for both companies; the email says the collaboration request has been withdrawn/rejected.
N: RelationshipCollaborationPartnerCreateAction R: Notify Collaboration	eServicecommonTrigaNotifyCollaboration_if.tcl	When a company accepts a request to collaborate with another company, the system connects the companies with the relationship Collaboration Partner. This causes the system to fire a trigger that sends an email to: – all Company Representatives for the company that made the request; the email says the invited company has accepted the request – all Company Representatives for the company that accepted the request; the email confirms that the request has been accepted

eService Trigger Program Parameters Object, Name and Revision	Trigger Program and JPO Method	Description of Action
N: RelationshipCollaborationPartnerDeleteAction R: Notify Collaboration	eServicecommonTrigaNotifyCollaboration_if.tcl	When a Company Representative dissolves a collaboration partnership, the system sends an email to: – all Company Representatives for the partner company; the email says the collaboration partner has been removed and specifies which company initiated the action – all Company Representatives for the company that dissolved the partnership; the email confirms that the partnership has been dissolved
N: TypeCompanyChangeNameAction R: Update Company Group Name	eServicecommonTrigaSyncAdminObjectName_if.tcl	When the host company changes the name of the company business object called Company Name, the trigger changes the name of the group called Company Name to match the name of the company object.
—	Matrix route user and route message	Change owner to Corporate.

Automation for Packages

Package objects are governed by the Package policy. Configurable automation for Packages is described in the table that follows.

eService Trigger Program Parameters Object	Trigger Program and JPO Method	Description of Action
N: PolicyPackageStateStartedPromoteCheck R: StateStartedPromoteCheck	eServiceCheckRoutes_if.tcl	Prevents promotion if the package is in an unfinished route. This trigger is only used if the route creator specified the state as a block promotion state for the package.

Automation for Persons

This table lists the framework automation related to persons.

eService Trigger Program Parameters Object	Trigger Program and JPO Method	Description of Action
N: PolicyPersonStateInactivePromoteAction R: Notify Employee Status	eServicecommonTrigaNotifyEmployeeStatus_if.tcl	When the administrator activates an employee, the person business object is promoted to Active. The promotion fires a trigger that notifies the employee that the account is active, so the employee can now log in.
N: AttributeEmailAddressModifyAction R: Notify Email Change <i>In Common Components 10, this trigger is deactivated and a AttributeEmailAddressModifyOverride trigger is added instead.</i>	eServicecommonTrigaNotifyEmailChange_if.tcl	Sends email to person whenever a change is made to their email address. The email is sent to both the old and new email address and gives the old and new address, the person who made the change, and the date and time of the change.
N: PolicyPersonStateActiveDemoteAction R: Notify Employee Status	eServicecommonTrigaNotifyEmployeeStatus_if.tcl	When the administrator deactivates an employee, the person business object is demoted to Inactive. The demotion fires a trigger that notifies the employee that the account is deactive.

eService Trigger Program Parameters Object	Trigger Program and JPO Method	Description of Action
N: RelationshipCompanyRepresentativeCreateAction R: Sync Admin Group Role	eServicecommonTrigaSyncAdminGroupRole_if.tcl	When a person is assigned as a Company Representative, the Company Representative relationship connects the person business object to the company bus object. When the relationship is created, this trigger assigns the person to the Organization Manager role.
N: RelationshipCompanyRepresentativeDeleteAction R: Sync Admin Group Role	eServicecommonTrigaSyncAdminGroupRole_if.tcl	When the Company Representative relationship is deleted, this trigger removes the person from the Organization Manager role.
N: RelationshipEmployeeCreateAction R: Notify New Employee	eServicecommonTrigaNotifyNewEmployee_if.tcl	When a person is added, the system connects the person business object to the company using the Employee relationship. When this relationship is created, this trigger notifies the person via IconMail and email (if configured).

Automation for Quotations

RFQ Quotation objects are normally governed by the RFQ Quotation policy. If the RFQ is revised or a supplier is removed from the RFQ, the system changes the policy to Suspended. Configurable automation for RFQ Quotations that is installed with the framework is described in the table that follows.

Request for Quotes (RFQs) were called Request to Suppliers (RTS) in previous versions of the application and therefore some administrative objects and properties refer to RTSes instead of RFQs.

eService Trigger Program Parameters Object	Trigger Program and JPO Method	Description of Action
N: PolicyRTSQuotationStateOpenPromoteCheck R: StateOpenPromoteCheck	eServiceCheckRoutes_if.tcl	Prevents promotion if the quotation is in an unfinished route. This trigger is only used if the route creator specified the state as a block promotion state for the quotation.
N: PolicyRTSQuotationStateReviewPromoteCheck R: StateReviewPromoteCheck		
N: PolicyRTSQuotationStateReturnedPromoteCheck R: StateReturnedPromoteCheck		

Automation for RFQs

RFQ objects are governed by the RFQ policy. Configurable automation that is installed with the framework is described in the table that follows.

Request for Quotes (RFQs) were called Request to Suppliers (RTS) in previous versions of the application and therefore some administrative objects and properties refer to RTS instead of RFQ.

eService Trigger Program Parameters Object	Trigger Program and JPO Method	Description of Action
N: PolicyRequestToSupplierStateStartedPromoteCheck R: StateStartedPromoteCheck	eServiceCheckRoutes_if.tcl	Prevents promotion if the RFQ is in an unfinished route. This trigger is only used if the route creator specified the state as a block promotion state for the RFQ.
N: PolicyRequestToSupplierStateInitialPackageReviewPromoteCheck R: StateInitialPackageReviewPromoteCheck		
N: PolicyRequestToSupplierStatePackageSentPromoteCheck R: StatePackageSentPromoteCheck		
N: PolicyRequestToSupplierStateResponseCompletePromoteCheck R: StateResponseCompletePromoteCheck		
N: PolicyRequestToSupplierStateFinalPackageReviewPromoteCheck R: StateFinalPackageReviewPromoteCheck		

Coding Examples

This section shows snippets of code supplied in the framework and MatrixOne Application JSPs that perform specific tasks. The code calls servlets and methods from the Matrix server ADK. For more information on using the Matrix server ADK, see the *Matrix Server ADK Programming Guide*.

Using Push and Pop to a Shadow Agent

When a program needs to perform actions that require privileges the current user doesn't have, you can "push" the context to a shadow agent, such as the User Agent person. After performing the actions, you can then "pop" the context back to that of the initial user.

To invoke push/pop to User Agent in a JSP, create a Tcl program that uses the push/pop shadow agent program, called `eServicecommonShadowAgent.tcl` (located in `EMATRIXHOME/framework/common`). The following code snippets show how to call the program and then push context to the shadow agent and pop the context back after the actions are performed. Also see the Tcl script `eServicecommonConnectDisconnectObjects.tcl`, which lets a user connect and disconnect an object even if the user doesn't have connect or disconnect access for the object. This program is also located in `EMATRIXHOME/framework/common`.

```
#The utLoad procedure loads other tcl utilities procedures
#Start of utLoad
proc utLoad { sProgram } {
    global glUtLoadProgs env
    if { ! [ info exists glUtLoadProgs ] } {
        set glUtLoadProgs {}
    }
    if { [ lsearch $glUtLoadProgs $sProgram ] < 0 } {
        lappend glUtLoadProgs $sProgram
    } else {
        return ""
    }
    if { [ catch {
        set sDir "$env(TCL_LIBRARY)/mxTclDev"
        set pFile [ open "$sDir/$sProgram" r ]
        set sOutput [ read $pFile ]
        close $pFile
    } ] == 0 } { return $sOutput }
    set sOutput [ mql print program '$sProgram' select code dump ]
    return $sOutput
}
# end utload

#include the push/pop shadow agent file by calling the
#utLoad Procedure
eval [ utLoad eServicecommonShadowAgent.tcl ]
```

```

#call the pushShadowAgent procedure to set the context to
#the shadowAgent
pushShadowAgent

#Next execute the commands the Shadow Agent needs to do
set sCmd "mql connect businessobject....."
set sCmd2 "mql add property....."

#If it is required that we exit on error on any command,
#popShadowAgent must be called to restore the original user
#context in the error handling code.
set mqlret [ catch {eval $sCmd} outStr]
set mqlret [ catch {eval $sCmd2} outStr]
.....
#If all commands were successful, call popShadowAgent procedure
#to restore the original user context
popShadowAgent

```

In a JSP, you invoke the program that uses eServicecommonShadowAgent.tcl via the MQL Command class. These snippets show how to invoke the eServicecommonConnectDisconnectObjects.tcl program in a JSP:

```

MQLCommand connectMQL = new MQLCommand();
connectMQL.open(context);
String connectMQLString = "execute program
eServicecommonConnectDisconnectObjects.tcl connect
"+sParentObjId+" "+sMarkupId+" relationship_EBOMMarkup {}";

// The tcl program
eServicecommonConnectDisconnectObjects.tcl switches the context
to a shadow agent, connects the objects, then pops the context
back to the original user.
connectMQL.executeCommand(context,connectMQLString);
String result = connectMQL.getResult().trim();
String error = connectMQL.getError();
connectMQL.close(context);

```

Setting a Target Page

```

String pageName = request.getRequestURI();
if (request.getQueryString() != null) {
    pageName += "?" + request.getQueryString();
}
Framework.setTargetPage(session, pageName);

```

Checking that the User is Logged In

```

if (!Framework.isLoggedIn(request)) {
    %>
    <jsp:forward page ="SampleLogin.jsp"/>
}

```

```

    <%
    return;

```

Getting the User Context

```
matrix.db.Context context = Framework.getContext(session);
```

Looking Up an Object by Symbolic Name

This code looks up the symbolic name for the Comments attribute.

```

    <%
    // load the comments attribute name
    String commentsStr =
    Framework.getPropertyValue(session, "attribute_Comments");

    // read the comments attributes
    String comments = getAttribute(session, rtsObject,
    commentsStr);
    %>

```

Opening an Object and Getting Information

This code is from a JSP that displays details about a business object.

```

// open the current BusinessObject
String busId = request.getParameter("busId");
BusinessObject busObj = new BusinessObject(busId);
busObj.open(context);

//get the BusinessObject TNR
String busType = busObj.getTypeName();
String busName = busObj.getName();
String busRevision = busObj.getRevision();

//get the BusinessObject basics
BusinessObjectBasics busBasics = busObj.getBasics(context);
String busOwner = busBasics.getOwner();
String busOriginated = busBasics.getCreated();
String busModified = busBasics.getModified();
String busDescription = busBasics.getDescription();

//get the current state of the businessobject
State busState = getCurrentState(session, busObj);
String busCurrentState = busState.getName();

```

Reading Attributes

This code gets the name and value for an object's attributes. The code highlighted in bold and green is the relevant code.

```

// read the related business object attributes
AttributeItr attItr = new
AttributeItr(busObj.getAttributes(context).getAttributes());

```

```

while (attItr.next()) {
    Attribute attribute = attItr.obj();
    AttributeType attrType = attribute.getAttributeType();
    attrType.open(context);
    if (colorFlag) {
        rowColor = "even";
        colorFlag = false;
    } else {
        rowColor = "odd";
        colorFlag = true;
    }
}
%>
<tr class="<%= rowColor%>">
    <td width="50%"><%= attItr.obj().getName() %></td>
    <td width="50%"><%= attItr.obj().getAttribute() %><nbsp;</td>

```

Checking a User's Role

This code checks to see if the current user is a buyer.

```

<%
// see if the user is a buyer
if (!isBuyer(session)) {
%>
<jsp:include page="eServiceEnd.jsp" />
<%
return;
}

```

Displaying MQL Notices

You can use the program emxMQLNotice.jsp, which is located in the ematrix directory, to fetch and display MQL notices. Call the program using an include statement, like the following:

```

<jsp:include page="emxMQLNotice.jsp" flush="false"/>

```

Getting Admin Object Names from Symbolic Names

Java

The method described below can be used to get admin object names in JSPs, JavaBeans, and JPOs.

To get the name of an admin object from the cache based on the name stored in the symbolic name admin property, use this method:

```

PropertyUtil.getSchemaProperty("Symbolic name");

```

For example, to get the actual admin name for the type whose symbolic name property is “type_Part”, use this code:

```

String partAdminName =
PropertyUtil.getSchemaProperty("type_Part");

```

The code would return the actual type name “Part” from the cache.

After getting the name of an admin object, you would typically get the internationalized version of the name. For instructions, see [Sample Code for Getting Admin Object Translations](#) in Chapter 6.

For more information on symbolic names for admin objects, see [How the System Identifies Administrative Objects: Symbolic Name Properties](#) in Chapter 5.

Tcl

This section describes how to get admin object names when writing a Tcl program. However, Java is the preferred method for writing programs. For example, use a JPO instead of Tcl for triggers.

The program `eServiceSchemaVariableMapping.tcl` contains the procedure `eServiceGetCurrentSchemaName` that gets the current name for an admin object. The signature for using the procedure is:

```
proc eServiceGetCurrentSchemaName { sItemName sProgramName
    sPropertyName {sStateName ""}}
```

Where:

- `sItemName` is the admin object type, such as attribute, type, relationship.
- `sProgramProg` is the name of the program on which all the admin object's properties are registered.
- `sPropertyName` is the admin object's symbolic name.

For example, to get the name of the admin object whose symbolic name property is `attribute_Originator`:

```
set sAttrOriginator [eServiceGetCurrentSchemaName attribute
    eServiceSchemaVariableMapping.tcl attribute_Originator]
```

For example, to get a policy state name:

```
set sAttrOriginator [eServiceGetCurrentSchemaName state
    eServiceSchemaVariableMapping.tcl state_Active
    policy_eServiceTriggerProgramPolicy]
```

Getting Symbolic Names

Java

The method described below can be used to get symbolic names for admin objects in JSPs, JavaBeans, and JPOs.

To get the symbolic name for an admin object, use this method:

```
FrameworkUtil.getAliasForAdmin(context, "Type", "admin name",
    "useCache value");
```

Where:

- “Type” is the admin type, such as Type, Attribute, Relationship, etc.
- “admin name” is the name of the admin object, such as Part or Quantity.
- “useCache value” is a Boolean to get the value from the cache or from the database. True gets the value from cache and false gets it from the database.

For optimum performance, obtain the symbolic name from the cache and not from the database.

Example 1

To get the symbolic name for the Part type admin object Part from the cache, use this code:

```
String typePart = FrameworkUtil.getAliasForAdmin(context,
    "Type", "Part", true);
```

The code would return “type_Part”.

Example 2

To get the symbolic name for the EBOM relationship admin object from the cache, use this:

```
String releBOM = FrameworkUtil.getAliasForAdmin(context,
    "Relationship", "EBOM", true);
```

The code would return “relationship_EBOM”.

For more information on symbolic names for admin objects, see [How the System Identifies Administrative Objects: Symbolic Name Properties](#) in Chapter 5.

Tcl

This section describes how to get the symbolic name for admin objects when writing a Tcl program. Java is the preferred method for writing programs. For example, use a JPO instead of Tcl for triggers.

The program eServiceSchemaVariableMapping.tcl contains the procedure emxGetPropertyFromAdminName that gets the symbolic name for an admin object. The signature for using the procedure is:

```
proc emxGetPropertyFromAdminName {sAdminType sRegProg
    sAdminName}
```

Where:

- sAdminType is the admin object type, such as attribute, type, relationship.
- sRegProg is the name of the program on which all the admin object's properties are registered.
- sAdminName is the current name of the admin object whose symbolic name you want.

For example, to get the symbolic name for the attribute named Originator:

```
set sAttOriginatorProperty [emxGetPropertyFromAdminName
    "attribute" eServiceSchemaVariableMapping.tcl Originator]
```

Use the emxGetPropertyFromStateName procedure to get the symbolic name for a state. the signature for using the procedure is:

```
proc emxGetPropertyFromStateName {sPolicyName sStateName}
```

Where:

- sPolicyName is the name of the policy that contains the state.
- sStateName is the name of the state whose symbolic name you want.

For example:

```
set sStateProp [emxGetPropertyFromStateName Person Active]
```

Migrating to Version 10 Tree API for Custom Pages

The version 10 release of the framework has a number of tree enhancements and there are many changes to the existing tree code base. (For screen shots and parameters to use for the configurable tree, see *Building Navigation Trees* in Chapter 8.) In most cases, customers will not have to make changes to their code before using this version of the AEF. There are one or two situations that may require some modification to a customer's custom JSPs before they can use version 10 of the AEF.

Customers that meet the following criteria should not be impacted by the version 10 tree changes:

- Customers who have not written any custom JSP code and have been able to leverage the configurable components in their entirety.
- Customers who have written custom JSP code but who have not manipulated the tree nodes within that code. Refer to the following sections to determine if your code is considered to manipulate the tree nodes and if so, the calls that should be used to migrate.

Issue #1: Deleting Node & Reloading Tree Page via Custom JavaScript

If you have code that deletes an object displayed in the tree and that code refreshes the tree display so that node is removed, you may have used one of the following techniques to refresh the tree.

Technique 1 that will Require Migration

Reload the tree page assuming the Frameset structure will not change. To find situations like this, you should review any pages that refresh the tree page using a command like:

```
parent.window.opener.parent.parent.document.location.reload();
```

Notice how the use of "...parent.parent..." assumes the Frameset layout in the tree page. Because the new Tree Frameset structure has changed, the above line of code will fail once the new tree pages are installed.

Technique 2 that will Require Migration

Similar to the technique above, a customer may have targeted a specific URL assuming the Frameset structure as well. Below is an example of this code:

```
top.opener.parent.parent.document.location.href=emxTree.jsp?mode=insert&objectId=...;
```

Again notice the use of "...parent.parent..." and how it assumes the Frameset layout.

Recommended Solutions/Substitutions for above Techniques

The above techniques assume that the tree frame is at 2 or 3 levels above the opener frame. When the tree layout is changed for version 10, this will be affected. So it is recommended to use one of the following techniques in this case:

Issue #2: Renaming Tree Nodes via Custom JavaScript

Inserting a tree node to existing Tree (mode=insert)

If the goal is to add a new node to the tree, the following can be used:

```
top.addContextTreeNode(objectId, jsTreeID, emxSuiteDirectory);
```

Creating a root node to get new Tree (mode=replace)

Find the tree content frame with the name “content” using one of the API “findFrame()” available at the top frame object and target that frame as given below:

```
var contentFrameObj = top.findFrame(top.opener, top, "content");
contentFrameObj.document.location.href=
emxTree.jsp?mode=insert&objectId=...;
```

If you have code that renames an object displayed in the tree and that code accesses private methods to change the name of the tree node, you may have used one of the following techniques to change the node name.

Technique 1 that will Require Migration

This technique first accesses a private data member (**nodemap**) to get the tree node and then uses a function **changeObjectName()**, deprecated now, to modify that node. Because this data member (**nodemap**) is private that is not to be used as APIs, it is not guaranteed from one release to the next. Look for code like the following on pages that rename object names that are displayed in the tree.

```
node = tree.nodemap["<%=objectId%>"] [0];
node.changeObjectName("<%=treeLabel%>", false);
```

Recommended Solutions/Substitutions for Technique 1

Note that the method “ChangeObjectName()” is currently deprecated. It will work fine for now without any issues but is not guaranteed to work in the future. The equivalent recommended method is “changeName()”.

The following approach is recommended to address the above use case:

Get the tree object for any given objectId and apply the change name method on that:

```
objDetailsTree.getObject("<%=objectId %>").changeName
("<%=treeLabel%>" , bRefresh);
```

Technique 2 that will Require Migration

This technique for changing a node name has the same issue as the previous technique in that it accesses the same private data member (**nodemap**), but this technique uses a different private function (**setName**) than the previous technique. This technique also determines which node to change in a different manner. Below is another example of code that should not be used.

```
strName = labelArray[k];
for (var i=0; i < node.tree.nodemap[strObjectID].length; i++)
{
    // Change the name only if it matches with the old name
    if (node.tree.nodemap[strObjectID][i].name ==
strObjectOldName)
        node.tree.nodemap[strObjectID][i].setName(strName);
}
```



```
}
```

Recommended Solutions/Substitutions for above Technique 2

The second technique loops through the tree nodes and again accesses a private data member (**nodemap**) to get the tree node, to change the name (label). Use the following approach in the above case:

Get the tree object for any given objectId and apply the change name method on that:

This issue may be addressed with the code similar to the following:

```
for (var i=0; i<labelArray.length; i++) {  
    tree.getObject[labelArray[i]].changeName(labelArray[i+1],  
false);  
}
```

Preventing Similar Issues in the Future

Because JavaScript does not permit controlling the use of Private and Public methods as is done in other languages like Java and C++, there is always be the possibility that programmers will access the Tree's private data members and functions. To help prevent this from happening, use only the methods described in *Tree JavaScript API for Custom JSPs* in Chapter 8.

B

Appendix B: Dynamic UI Parameters and Settings

Introduction

This section contains alphabetical lists of parameters and settings used by dynamic user interface components. For details on how to specify the parameters and settings for a particular component and other information about configuring the components, see [Chapter 8, *Configuring the User Interface*](#).

The parameters and settings are divided into these sections:

- [*Administrative Object Parameters*](#)
- [*Settings*](#)
- [*URL Parameters*](#)
- [*Parameters Automatically Passed to URLs*](#)

Administrative Object Parameters

This section lists parameters for the administrative objects that represent dynamic user interface components, such as menu and command objects. You can define these parameters using the create and edit dialog boxes for the administrative objects in Business Modeler or using commands in MQL. The parameter names reflect the names using in Business Modeler and in some cases are different than the command used in MQL for the parameter. For information on the appropriate MQL commands to use, see the *MQL Guide*.

A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z

Parameter	Used For UI Component	Description	Accepted Values/ Examples
Access (user MQL command)	command objects for submenu links, toolbar tools, toolbar items, tabs table object columns	The persons, roles, and groups who can access the UI component. When you assign a role or group, all child roles/groups also receive access. To make the component available to all users, regardless of role/group assignments, choose All. Note that if no users are assigned access, the system assumes all users have access. For information on how this parameter works with other settings and parameters that control access to links, see Controlling User Access to User Interface Components in Chapter 8.	Names of group, role, person administrative objects. or All (default)
Alt	command objects for toolbar tools	The text to display in the ToolTip that appears when a user moves the mouse pointer over the UI component. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components in Chapter 8. The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	emxFramework.Home.Logout Change Password
Applies To	table object columns	The item to apply the select express to: Business object or relationship. Applying an expression to a relationship is used to get a table of objects related to a known object. For example, getting all EBOMs for a part.	Business Object Relationship
Arguments	inquiry objects	The arguments are used to replace the macros within the code. And the symbolic names will be translated into the actual name during substitution. ID is a reserved keyword for the inquiry objects used with the configurable table components. If the macro \${ID} is used in the code section, the argument ID must be assigned to any dummy value. This value is replaced by the request parameter "objectId" at run time.	PART=type_Part EBOM_REL=relationship_EBOM ID = dummy

Parameter	Used For UI Component	Description	Accepted Values/ Examples
Channel	portal objects	The channels associated with the portal	Names of channel objects, such as: AEFCollections AEFPowerViewChannel SPCSCOChannel
Command (specified in the channel Items tab in Business Modeler)	channel objects	The command objects that represent the tabs in the channel. The order of the commands is the order the tabs appear in the channels in the user interface.	Names of command objects, such as: SPCSCOInstructions SPCRoutesTreeCategory
Code	inquiry object	The code section determines the output object list for a specific business requirement. The code is generally an MQL temp query or expand bus command that selects the found objects' ids. The macro <code>\${ID}</code> is substituted with the business object id during run time. This object Id is available to <code>emxTable.jsp</code>, passed in from the tree node or any other commands that operate based on a specific business object. For example, typically when <code>emxTable.jsp</code> is called from a tree, the "objectId" parameter is passed with the specific OID assigned to the current tree node and that OID will get substituted with <code>\${ID}</code> at run time. <i>Note that this <code>\${ID}</code> is not same as the RPE variable <code>\${OID}</code>.</i> All the admin types like Type names (Part, Buyer Desk), attribute names, relationship names must be defined as macros and each macro will be assigned with its symbolic name in the arguments list. All other macros (like <code>\${PART}</code>, <code>\${EBOM_REL}</code>) defined within the code are to be assigned to the appropriate symbolic names in the Argument, if required. The macros are evaluated at run time by converting the symbolic name to actual name.	<pre>temp query bus \${PART} * * select id dump expand bus \${ID} from relationship \${EBOM_REL} select businessobject id select relationship id dump </pre>
Expression	table object columns	The select expression to be used to get the column data. This expression is applied to either the relationship or business object, as specified in the Applies To options. Note that the Expression and Applies to options in Business are equivalent to the businessobject and relationship MQL commands. For more information on specifying column data, see Defining Table Column Data in Chapter 8.	For business objects: - type - name <code><attribute[attribute_Originator].value></code> For relationships: <code><attribute[attribute_FindNumber].value></code> <code><attribute[attribute_Qty].value></code>

Parameter	Used For UI Component	Description	Accepted Values/ Examples
Format	inquiry objects	The format determines how that line is to be reformatted before printing the output. If the table column is defined based on a list of business object IDs, the format must be a list of business object IDs separated by a new line. The format value should be \${OID} and an example output is: <pre>12333.3453.56765.3443 12533.3453.56765.3453 12633.3453.56765.3943 ...</pre> If the table column is defined based on business object and relationship IDs, the format must be a list of business object and relationship ID pairs, separated by the new line character. In each pair, the relationship ID comes first and is separated by the business object ID by a ~. The format value should be \${RELID}~\${OID} and an example output is: <pre>9089.34345.56567.21312~12333.3453.56765.3443 4564.3445.567.7868~12533.3453.56765.3453 6465.2342.4566.3212~12633.3453.56765.3943 ...</pre>	<pre>\${OID} \${RELID}~\${OID}</pre>
Heading (label MQL command)	table objects	The text that should appear on the column header. Either a string resource ID for the text string or the actual text string that should appear. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components in Chapter 8. The system first looks for a string resource ID that matches the entered value. If it finds one, it uses the value for the ID. If it doesn't find one, it displays the entered text.	<pre>emxEngineeringCentral.com mon.Name emxTeam.Common.Project Name Description</pre>
Height	channel objects	The height of the channel.	The default is 260.

Parameter	Used For UI Component	Description	Accepted Values/ Examples
href	command objects for the Toolbar, My Desk and Actions menus menu objects for navigation trees table object columns	The URL to call when a user clicks the label or image that represents the UI component. For menu objects that represent the root node for a tree, the href should always point to the Properties page for the tree's object. This page is always displayed in the right frame, which is called "detailsDisplay" and not in another window or frame. The URL for command objects that represent tree categories are always displayed in the right frame and not in another window or frame. The URL that gets executed when a toolbar item is clicked is displayed in the frame or window specified in the Target Location setting. The possible values/cases are: • A JSP and any associated parameters. You can specify the path of the JSP using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros in Chapter 8. • A JavaScript function. • A JSP with embedded macros and expressions. For information on the macros and expressions, see the <i>Business Modeler Guide</i> or <i>SQL Guide</i>.	<pre>emxLogout.jsp \${SUITE_DIR}/ emxpartCreatePartDialog.jsp ?mode=Promote&id=0 javascript:window.open('em xChangePassword.jsp', 'ChgPwd', 'width=400,height=400') javascript:showModalDialog ('emxCreatePart.jsp', '400', '400') javascript:showModalDialog ('\${SUITE_DIR}/ emxCreateSketch.jsp', '600', '600') \${COMMON_DIR}/ emxPartDetails.jsp \${COMMON_DIR}/ emxTree.jsp javascript:window.open('em xEditPart.jsp', 'EditPart', 'width=400,height=400')</pre>
href	command objects for tabs	The URL to display when a user selects the tab. Tabs can display content from the following sources: • configurable table • configurable form • JSP page • URL If the specified URL contains a header parameter, it is not used because the header would be redundant with the tab label. For example, if the URL contains emxTable.jsp with a header specified, the table header is not used.	<pre>emxTable.jsp?xxxxx emxForm.jsp?xxxxx Sample.jsp?xxxxx http://www.matrixone.com</pre>
Icon	—	Icon for the object within the Matrix core. This parameter is NOT the image that displays on the dynamic user interface (for example, on a toolbar button). The Image setting is used to specify the UI image for components that have images associated with them. Many components, such as submenus, don't have images associated with them.	The name of an image file, such as Part.gif.

Parameter	Used For UI Component	Description	Accepted Values/ Examples
Items	menu objects for top-level menus menu objects for submenus menu objects for navigation trees menu objects for toolbars	Specifies child items assigned to an administrative object. The top-level My Desk and Actions menu objects are assigned menu objects that represent submenus for each application. The My Desk and Actions menu submenus for each application and the Tools menu are assigned commands. Menu objects that represent navigation trees are assigned commands, which represent tree categories. The order of the menus and commands in the Items tab is the order they appear on the user interface.	Names of menu objects. Names of command objects, such as: TMCWorkspacesMyDesk TMCCreateWorkspaceActions ENCBOMTreeCategory AEFRelToItemTreeCategory TMCRoutesTreeCategory ENCCreatePartActionLink TMCDeleteActionLink
Label	menu objects for My Desk and Actions tab submenus command objects for submenus and menus command objects for toolbar items	The text that should be used to represent the UI component on the interface. For example, the text to represent the submenu, link in a menu, tree or tree category, or toolbar item. The label can be defined using: - Any of the regular expression macros (including the special macros “\${TYPE}” “\${NAME}” and “\${REVISION}”). These keywords are considered as macros and evaluated at run time to display the current object type name and revision. - Static text to be displayed. - String resource ID defined in the string resource property file. To internationalize the text, you must use a string resource ID. See Internationalizing Dynamic UI Components in Chapter 8. The system initially considers every label text as a string resource ID. If it finds the value for this ID in the resource file, it displays the obtained value. If not, the text is displayed as is. The labels for tree categories can include a count of the number of items in the category. See Parameters for Tree Category Command Objects in Chapter 8. Tree category labels can also be defined using a JPO. See the Label Function and Label Program settings. If these JPO settings are defined for a category, they override any label defined using this parameter. The TreeLabel URL parameter for emxTree.jsp also overrides any label specified for the root node menu object.	<pre> \$<type> \$<name> \$<attribute[attribute_Weight].value> Engine - \$<type> \$<name> Connected ECR emxFramework.Common.Part \${TYPE} \${NAME} </pre>

Parameter	Used For UI Component	Description	Accepted Values/ Examples
sorttype	table object columns	Defines how the column values should be sorted. <i>This parameter currently is not available in Business Modeler and can only be set using the sorttype command in MQL. For example:</i> <pre> add table TestTable1 system column name Type label emxEngineeringCentral.Part.Type user all businessobject type setting "Registered Suite" EngineeringCentral sorttype numeric ; </pre>	numeric—Sorts the column values numerically. alpha (Default)—Sorts the column values alphabetically. other—Sorts the column values using a custom algorithm defined as a JPO and specified in the Sort Program setting for the column.
Test	inquiry objects	Use the Test tab to determine if the inquiry will parse the output as you have designed the JSP to expect to receive it.	—

Settings

This table lists and describes the settings for administrative objects that represent UI components. Note that the name and value for each setting are case sensitive.

A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z

Setting	Used For UI Component	Description	Accepted Values/Examples
Access Expression	command objects for menus and toolbar items, tabs, table columns, form fields, tree categories	Controls access to the component based on a valid Matrix expression. The expression gets evaluated at runtime. If the expression evaluates to True and no other access control prevents access, the component is shown. Otherwise it is hidden. The expression can contain any symbolic name, such as “attribute_weight”. For command objects assigned to menus, the value for this setting must be a valid context-based expression. The expression must be related to the person currently logged into the application.	- attribute[attribute_Weight].value > 100 The toolbar item displays only if the Weight attribute on the business object is greater than 100. - attribute[attribute_Originator].value == owner For menu command objects: - context.user.name=="Test Everything" - context.user.assignment == "Design Engineer" - context.user.assignment == "\$<role_DesignEngineer>"
Access Function	command objects for menus and toolbar items, table columns, form fields, tree categories	The name of the JPO method to invoke in the JPO specified for the Access Program setting. The Access function gets the input parameter as a HashMap, which contain all the request parameters that were passed into the Form page. The JPO method must return an object of class Boolean. If the returned value is true and no other access control prevents access, the component is displayed. If false, it is hidden. To see a sample JPO method to control access, see Sample JPO for Controlling Access in Chapter 8. The setting Access Program Method is equivalent to Access Function and is supported for backward compatibility. If both are used, precedence is given to Access Function.	The name of an access check method in the JPO specified in the Access Program setting, such as: emxAccessCheck()
Access Map		Equivalent to Access Mask, which takes precedence over Access Map.	See Access Mask.

Setting	Used For UI Component	Description	Accepted Values/Examples
Access Mask	command objects for menus and toolbar items, table columns, form fields, tree categories	Specifies the accesses the user must have for the current business object in order for the component to be displayed. If defined on menu commands, the access mask applies to the Person business object for the current user. If the user does not have all the specified accesses in the business object's policy for the current state, the component is hidden. If the user has the access and no other access control prevents access, the component is displayed. Note that the Access Mask setting does not work on a table that is launched from the My Desk tab. For more information, see Access Parameters and Settings in Chapter 8. The current objectId parameter value must be passed into the JSP so it can determine the object's policy accesses and validate the Access Mask. You can specify multiple accesses by separating the accesses with a comma. The equivalent setting Access Map is supported for backward compatibility but if both are used, Access Mask is given precedence.	Any set of accesses, separated by a comma. For example: Modify Delete ToConnect ToDisconnect FromConnect FromDisconnect Modify,Delete FromConnect,FromDisconnect
Access Program	command objects for menus and toolbar items, table columns, form fields, tree categories	Controls access to the UI component based on the output from a method in the specified JPO program. The program must be defined in Matrix. This setting requires that the Access Function setting also be specified. If Access Function is not set, the Access Program setting is ignored. The following input values are required for the program: • A list of all of the request parameters in a HashMap • Method name as a string • JPO Program Name as a string • Context The output must be a Boolean.	Name of a JPO defined as a program object in Matrix, such as: JPOAccessControl
Access Program Method		Equivalent to Access Function, which takes precedence.	See Access Function.
Action Type	command objects for toolbar items	Tells the configurable toolbar that the command is a separator instead of a link. If the setting is not included, the toolbar treats the command as a standard link.	Separator—A line that separates one or more links. If the command is assigned to the top-level menu and is therefore on the toolbar itself, it is considered a vertical separator. Otherwise, it is considered a horizontal separator.

Setting	Used For UI Component	Description	Accepted Values/Examples
Admin Type	table object column form object field	This setting is used for language translation of the column's data based on the specified admin type. Based on the admin type, the value is translated and presented. For attributes, it is necessary to provide the symbolic name of the attribute, which starts with "attribute_" and is followed by the attribute name with no spaces.	Type State Role Relationship attribute_UnitOfMeasure
Allow Manual Edit	form object field table object column	When true, users can manually edit the form row or table column. Applicable only when the range parameter is set to a URL or when the setting format is assigned to date. It is ignored in all other cases. When this setting is true, the Admin Type setting is ignored.	true false (default)
Alternate OID expression	table object column	When a column's data is configured to show as a hyperlink using the href parameter, the objectId to be passed in can be different from the objectId for that row. This expression is used to obtain the objectId to be passed in with the column link.	<code><to[relationship_NewPartPartRevision].from.id></code> <code><to[relationship_EBOM].from.id></code>
Alternate OID expression	form object field	By default, when a field's data is configured to show as a hyperlink using the href parameter, the system passes the ID for the business object the form page applies to. Using this setting, you can have the system pass the ID(s) for a different object, namely, the ID(s) for the object(s) returned from the expression specified in this setting.	<code><to[relationship_NewPartPartRevision].from.id></code> <code><to[relationship_EBOM].from.id></code> To see an example of a field that uses an Alternate OID expression and an Alternate Type expression, see Field Values with Hyperlinked Data Using an Alternate OID and Alternate Type Icon in Chapter 8.
Alternate Policy expression	table object column	This setting is required to display the state of any connected objects and show the value translated. This setting is applicable only when the Admin Type setting is set to State.	Any select expression that evaluates to a policy of a connected object.
Alternate Type expression	table object column form object field	When the Show Alternate Icon setting is true, this expression is used to obtain the object type. Based on the obtained type, the corresponding icon is displayed.	<code><to[relationship_NewPartPartRevision].from.type></code> <code><to[relationship_EBOM].from.type></code>

Setting	Used For UI Component	Description	Accepted Values/Examples
Auto Filter	table object column	Determines whether users can filter the table rows based on data in the column. If at least one column in the table has Auto Filter set to true, the Filter  tool displays in the page toolbar. This setting cannot be applied to any column that has multiple values for a single data column cell. For more information on table filtering, see Methods for Filtering a Table in Chapter 8. Configuring many columns for auto filtering or applying auto filter to inappropriate columns can potentially have high performance impact when viewing the filtered list. For optimum performance, define a limited number of columns with Auto Filter. Also choose columns that have a limited set of possible values on which the user may want to filter the table data. For example, state and owner columns are good candidates. Columns such as description and name are inappropriate because of the unlimited number of values that will result.	true—The Filter tool displays in the page toolbar. When clicked, the column is available on the Auto Filter Selection page. false (default)—The column is not available in the Auto Filter Selection page.
Cols	form object field	Used when the input type is set to textarea. This setting limits the length of the textarea on the form and specifies the visible width in average character widths. If not specified, it uses the HTML default, which is 25.	25 40 50
Column Icon	table object column	Use to display an icon for the column's data instead of other data. Required when the setting "Column Type" is assigned to "icon". See Column Values as Icons in Chapter 8.	Name of an image file such as: images/NewWindow.gif images/EditItem.gif

Setting	Used For UI Component	Description	Accepted Values/Examples
Column Type	table object column	The setting “Column Type” is used when no expression is defined for the column data. For more information on specifying column data, see Defining Table Column Data in Chapter 8. For recommendations on improving the performance of table columns whose data is generated with a JPO program (Column Type set to program or programHTMLOutput), see Improving Performance of Table Columns in Chapter 8. For a sample JPO program to get column data, see Sample JPO for Getting Table Column Values in Chapter 8.	The column type can be one of the following: • program—The values for this column are obtained from a program (JPO). With this setting, the program and function name are required as settings. • programHTMLOutput—Same as the “program” setting above, except that the column value output is in HTML format. Column values are placed in table cell between <td> and </td> tags. This setting ignores other column settings such as Show Type Icon, href, format, and Alternate OID expression. • icon—Used when the column values are shown as an icon. The setting “Column Icon” must be defined with the icon to be displayed. • checkbox—Used when the column values are check boxes shown grayed or not-grayed based on the access to the object in that row. This access can be based on business logic and defined in a JPO or it can be role-based and defined by assigning roles to the column. If the checkboxes have no access restrictions, this setting is not required and you can create checkboxes by passing in the parameter selection=multiple to emxTable.jsp. • separator—Used to define a column of white space between standard data columns. A separator is especially useful to separate two groups of columns.
Confirm Message	command object for toolbar items	Provides a JavaScript confirmation message when users click on the toolbar item. For example, a custom delete message can be configured for the onClick event to display a JavaScript confirm message to the user. The Confirmation dialog has the “OK” and “Cancel” button. Clicking OK proceeds with the processing and Cancel cancels the operation.	The actual text to display in the confirmation message or a string resource property key. To internationalize the message, a string resource key must be used: Are you sure you want to delete this object? emxFramework.common.alertMsg

Setting	Used For UI Component	Description	Accepted Values/Examples
Currency Converter	menu objects for navigation trees command objects for menu links and tools, tree categories, and toolbar items table object columns form object fields	Specifies whether the target page should include the Currency Converter tool. In the href URL called when the UI component is clicked, the system passes a parameter called CurrencyConverter and includes the value specified for this setting. If the value is True, the target page includes the Currency Converter tool. If this setting is not included, the value is assumed to be false and the target page does not display the Currency Converter tool.	True False (default)
Currency Expression	table object columns	The name of the currency to convert from. Required for columns whose values are monetary and that can be converted from one currency to another using the Conversion tool and defined exchange rates. The value should be a select clause that returns the value for the Currency attribute for a specific object or relationship, or the actual value. The currency to be converted from should be the currency in which the user entered the data in (the As Entered currency).	For example, to convert currency data for an RFQ Quotation, the following select clause returns the supplier's currency format. to[Supplier Line Item].attribute[Currency]
Default	form object field	If a field's value is empty or null and this setting is defined, the default value is displayed for the field.	The default value you want to display. This can be a string resource key or the actual characters you want to fill in as the default. The wildcard (*) can be used for search criteria fields. emxFramework.Common.default All *
Default Category	menu objects for navigation trees	Specifies the category that should be selected when the tree first opens or is first inserted into another tree. By default, the root node is selected, which means the object's Properties page displays. If the tree contains a category that users frequently want to see, you can use this setting to have it selected instead. Alternatively, you can pass the DefaultCategory parameter to emxTree.jsp. If both are defined, this URL parameter overrides the setting. Note that a tree does not look for default categories defined for its sub-trees (assigned sub-menus).	The name of a category command object that should be selected when the tree first opens. The command object must be assigned to the tree menu object and if it is not, the root node is selected. Default Category=PMCWBS

Setting	Used For UI Component	Description	Accepted Values/Examples
Display Format	table object columns form object fields	Specifies the number of the date format for any column or field where the value of the format setting is “date.”	3 - SHORT (12/12/52) 2 - MEDIUM (Dec 12, 1952) 1 - LONG (December 12, 1952) 0 - FULL (Tuesday, December 12 1952 AD) Default is set in emxSystem.properties: emxFramework.DateTime.DisplayFormat=MEDIUM. emxSystem.properties uses words, but the Display Format setting uses numbers.
Display Time	table object columns form object fields	Controls whether the time is displayed along with the date for columns and fields whose format is set to date.	–true –false Default is set in emxSystem.properties for the property emxFramework.DateTime.DisplayTime.
Editable	form object fields	Use to indicate whether the field is displayed as editable or read only. Only applies for Edit mode. View mode ignores the setting.	true (default)—Users can edit the field when shown on the Edit mode form. false—Users cannot edit the field when shown on the Edit mode form. The field looks just like it does in View mode except it is never hyperlinked.
Effective Date Expression	table object columns	Used for columns whose values are monetary and that can be converted from one currency to another using defined exchange rates. The value should be a select clause that provides the value for the Effectivity Date attribute on a specific object or relationship. The system uses this date to get the currency conversion whose rate period falls within this date. If this setting is not added, the current date is used.	For example, for currency data for an RFQ Quotation, the following select clause returns the effectivity date. to[Supplier Line Item].attribute[Effectivity Date]
Expand Function	command object for tree category	Specifies the function in the JPO specified in the Expand Program setting that gets the list of child nodes for the tree category.	Name of a method in the Expand Program JPO.

Setting	Used For UI Component	Description	Accepted Values/Examples
Expand Inquiry	command objects for tree categories	Use to configure the tree category so a list of business objects is inserted under it automatically, without users having to first view details for objects. For example, Folder categories are often configured to display all folders without users having to view the folders first. Use this setting to define the list of business objects using an inquiry admin object. You can also define the list using a JPO with the Expand Program and Expand Function settings. If both an inquiry and JPO are specified, the JPO takes precedence. For more configuration options for this type of category, implications for performance, and rules for configuring the inquiry object, see Configuring Dynamic Expand for Tree Categories in Chapter 8.	Name of an inquiry admin object that retrieves the business objects to list for the category: TMCFolder TMCSUBFolders ENCEBOMList

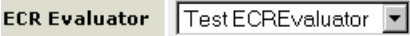
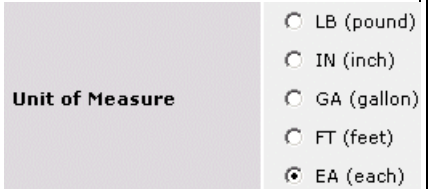
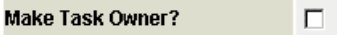
Setting	Used For UI Component	Description	Accepted Values/Examples
Expand Program	command object for tree category	Use to configure the tree category so a list of business objects is inserted under it automatically, without users having to first view details for objects. For example, Folder categories are often configured to display all folders without users having to view the folders first. Use this setting to define the list of business objects using a JPO. You can also define the list using an inquiry with the Expand Inquiry setting. If both an inquiry and JPO are specified, the JPO takes precedence. This parameter specifies a JPO that contains a method to get the list of objects to insert under the category. The method is specified using the Expand Function setting. For more configuration options for this type of category, implications for performance, and inputs/outputs for the JPO, see Configuring Dynamic Expand for Tree Categories in Chapter 8. Important Note: <i>This feature is recommended primarily for hierarchically organized categories such as Folders and Subfolders. For best performance and consistency of behavior across applications, the preferred method of navigation is to use the standard tree behavior that does not use a dynamic expand. If conditions require the use of this setting, try to restrict its use to categories that will contain relatively few objects.</i>	Name of a JPO program added to the database.
Export	table object columns	Specifies whether column data is exported or not. Use this to change the export value on a column-by-column basis. By default, all column types except programHTMLOutput are exported. If you want to additionally include programHTMLOutput, or exclude other column types from the export, change this setting.	–true –false

Setting	Used For UI Component	Description	Accepted Values/Examples
Field Column Headers	form object field	Used in conjunction with the Field Type=Table Holder setting. Specifies the labels for the column headings. The number of labels should be the same as the value for the Field Table Columns setting and should be separated by a comma. <i>See Fields Arranged in a Table in Chapter 8.</i>	Comma-separated list of column heading labels: Min,Max,Avg
Field Row Headers	form object field	Used in conjunction with the Field Type=Table Holder setting. Specifies the labels for the row headings. The number of labels should be the same as the value for the Field Table Rows setting and should be separated by a comma. <i>See Fields Arranged in a Table in Chapter 8.</i>	Comma-separated list of row heading labels: Weight,Volume
Field Size	form object field	Determines the width of a textbox input type field. The width is given in pixels except when Input Type is textbox or not set. In that case, its value refers to the (integer) number of characters.	Number of pixels, for example: 30 20 is the default.
Field Table Columns	form object field	Used in conjunction with the Field Type=Table Holder setting. Defines the number of columns for the table. <i>See Fields Arranged in a Table in Chapter 8.</i>	2, 3, ...
Field Table Rows	form object field	Used in conjunction with the Field Type=Table Holder setting. Defines the number of rows in the table. <i>See Fields Arranged in a Table in Chapter 8.</i>	1, 2, ...
Field Type	table object columns	Setting used in only in edit mode while updating the table data.	basic—The column displays basic information for the business object. Basic information includes name, type, originated, policy, etc. Specifying basic as the field type is only needed when the column is editable. The only editable basic information is: type, name, revision, current, policy, description, owner, vault. attribute—The column displays values for an attribute on the business object, such as Originator or Weight.

Setting	Used For UI Component	Description	Accepted Values/Examples
Field Type	form object field	This setting is used for several purposes: - To indicate that the field's data should be obtained from a program or image instead of an expression. - When the field data is obtained from an expression, if the data is basic information or an attribute. The system needs to know whether a field's data is basic information or an attribute in order to update the information correctly. Specifying whether the field is basic or an attribute is only required for fields that will be editable. - To indicate the field is a dummy field that defines fields to display in a table or group. For more information on specifying field data, see Defining Table Column Data in Chapter 8. The field type can be one of the following: - program—The values for this field are obtained from a program (JPO). With this setting, the program and function name are required as settings. - programHTMLOutput—Same as the “program” setting above, except the field value output is in HTML format. Field values are placed in the table cell between <td> and </td> tags. This setting ignores other field settings such as Show Type Icon, href, format, and Alternate OID expression. - image—The field's value is an image. To specify the image file, use the Image setting. - basic—The field displays basic information for the business object. Basic information includes name, type, originated, policy, etc. Specifying basic as the field type is only needed when the field is editable. The only editable basic information is: type, name, revision, current, policy, description, owner, vault. See the column on the right for more acceptable values.	The field type can be one of the following (continued from column on left): - attribute—The field displays values for an attribute on the business object, such as Originator or Weight. - Section Header—Adds a new section heading between the form fields. The setting Section Level determines the heading level. See Field as Section Header and Separator in Chapter 8. - Section Separator—Adds white space to separate fields and sections. - Group Holder—Groups the fields under the field in one row. Uses the Group Count setting to determine the number of fields to group. When a field has a Field Type of Group Holder, it serves as a dummy field to define the group and does not appear on the form. - Table Holder—Arranges the fields under the field in columns and rows. Table Holder fields serve as dummy fields to define the fields to display in a table. Also see these settings: Field Column Headers, Field Row Headers, Field Table Columns, Field Table Rows. - emxTable—Embeds a configurable table in the form. Used in conjunction with the table setting and either the inquiry or program setting. See Field that Embeds a Configurable Table in the Form in Chapter 8.

Setting	Used For UI Component	Description	Accepted Values/Examples
format	form object field	Use to specify the type of data in the field. If the Editable setting is true and the field is on an Edit mode form, the system uses these format values to validate the field value. Validation takes place on the client side, before updating the object displayed in the form. For more information on validation, see Validating Form Field Data in Chapter 8.	- date—Uses the tag lib “emxUtil:izDate” to format the displayed field value. To see an example of a field configured as a date field and that has date validation, see Field Value with Dates in Chapter 8. - currency - numeric—Use if the value must be validated as a number before updating the values. Applicable only in Edit mode.
format	table object column	Specifies the format to display the column data. If at least one column has the format set to currency or UOM, the Conversion tool displays in the table page’s page toolbar. When a user clicks the tool, the system opens a new window and displays all column data defined with format=currency and UOM to the currency and unit of measure selected in preferences. For information on the currency and unit of measure preferences, see Unit of Measure Conversion and Currency Conversion in Chapter 8.	date—Displays the column values as a date. Uses the tag lib “emxUtil:izDate” to format the display. currency—Displays the column values as currency. UOM—Displays the column values as Unit of Measure and enables the Unit of Measure conversion interface. email—Displays the column values as an email address. When a user clicks the email address, the email editor configured in the client is presented.
function	form object field	The name of the method to call within the JPO program specified in the program setting. This method within the JPO is used to get the field values if the setting “Field Type” is set to “program” or “programHTMLOutput”. For more information, see Field Values Obtained from Program in Chapter 8.	The name of a function in the program JPO, such as: - getAssignedBuyerDesk - getPackageAccess - getParentPart
function	table object column	The name of the method to call within the JPO program specified in the Program setting. This method within the JPO is used to get the column values if the setting “Column Type” is set to “program” or “programHTMLOutput” or “checkbox”. See Column Values as Program Output in Chapter 8.	getAssignedBuyerDesk getPackageAccess getParentPart getCurrentState
Group Count	form object field	Used in conjunction with the Field Type=Group Holder setting. Specifies the number of fields below the Group Holder field to place in the grouped row. See Fields that are Grouped in Chapter 8.	2, 3, 4, ...

Setting	Used For UI Component	Description	Accepted Values/Examples
Group Header	table object column	Defines header text to display over several consecutive columns. For example, if you want a group header over three consecutive columns, add this setting to each column and assign the same value for each. You may want to separate grouped columns using a separator column, which is just a column of white space. Add a separator using Column Type=Separator. To see an example, see Grouped Columns and Column Separator in Chapter 8.	Static text or string resource id.
Help Marker	menu objects for navigation trees command objects for menu links and tools, tree categories, and toolbar items table object columns form object field	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help in Chapter 8. In the href URL called when the UI component is clicked, the system passes a parameter called HelpMarker and includes the marker text specified for this setting.	The naming convention for help markers is the page title, as displayed at the top of the visual page, prefixed with “emxhelp”. The marker is all lowercase with no spaces.
Hide Label	form object fields	Displays or hides the label for a particular row on a form.	True False
Image	command objects for toolbar tools command objects for tree categories	Name of the image file that should be displayed on toolbar tool commands or next to the label for the tree root node or tree category. You can specify the path of the image file using any of the standard directory macros or you can leave off the path designation to use the registered directory. For more information, see Directory Macros in Chapter 8.	- iconPerson.gif - buttonReports.gif \${COMMON_DIR}/buttonEdit.gif \${SUITE_DIR}/iconCreateECR.gif <i>If no image is specified for a tree root node or category, the system displays the standard no image icon to indicate there is an error in the definition of the tree.</i>

Setting	Used For UI Component	Description	Accepted Values/Examples
Image	form object field	Use to specify an image file when the field value should be an image only. This setting is required when the Field Type setting is set to image. This file must exist in the application server (not in the database). You can make the image a hyperlink by including a URL in the href parameter. To see an example of a field with an image, see Field Values as Hyperlinked Image in Chapter 8.	images/newPart.gif images/EditItem.gif
Image	toolbar menu object	Use to specify the image used for the menu. This is optional if a label has been defined but required if there is no label.	\${COMMON_DIR}/ iconSmallOrganization.gif
Input Type	form object field	Only used for forms in edit mode. Specifies the type of HTML control to display for user input. The setting accepts the values listed below and those listed in the cell on the right: textbox—This is the default. The field has a single-line box for typing text. This graphic shows a field configured as a Text Box with the Required setting equal to true.  textarea—The field has a multi-line box for typing text.  combobox—The field has a drop-down list of options and users can only select one value. Use combo boxes for attributes that have defined ranges and for attributes whose ranges are determined with a Range Helper URL. To see an example of a field with a combo box, see Field for Attribute with Choices in Combo Box in Chapter 8.  See the column on the right for more acceptable values.	listbox—The field has a list of range values. Although users can select more than one item in the list by holding the Ctrl key, it's best to use check boxes for multiple selection fields. radiobutton—There is a radio button next to each range value. To see an example of a field configured with radio buttons, see Field for Attribute with Choices in Radio Buttons in Chapter 8.  checkbox—The field has a check box next to each range value. For fields that allow multiple selections, use check boxes instead of a list box. 
inquiry	form object field	Used only for fields defined with Field Type=emxTable. Either this setting or the program setting must be used to define the objects to retrieve. Specifies the inquiry admin object that should be used to retrieve the business objects to be included in the table.	Names of inquiry admin object separated by a comma. inquiry=SCSBuyerDesk,SCSBuyerDeskAssigned inquiry=ENCAIIParts, ENCRReleasedParts

Setting	Used For UI Component	Description	Accepted Values/Examples
Label	command objects for structure trees	The label used to show the list of structures in the Structure Selector combo box. If there is only one command attached to the structure menu, no combo box is needed so the Label setting is ignored.	A string resource ID or text string: emxTeam.common.Folders Discussions EBOM
Label Function	menu objects for navigation trees command objects for tree categories	Specifies the function in the JPO specified in the Label Program setting that gets the value for the root node label.	The name of a method in the Label Program JPO.
Label Program	menu objects for navigation trees command objects for tree categories	Use to define the label for the root node/tree category using a JPO. This parameter specifies a JPO that contains a method to get the value for the label. The method is specified using the Label Function setting. The input for this JPO must include: –A list of all of the Request Parameters in a HashMap –Method Name as a string –JPO Program Name as a string The output should be the string that is returned to the tree and used for the label. There are two other ways to define the label for a tree node: using the TreeLabel URL parameter passed to emxTree.jsp and using the Label parameter for the menu object. The URL parameter takes precedence over the other two methods, then the JPO, then the Label parameter. Tree category labels can also be defined using a Label parameter for the command object. The JPO takes precedence over the parameter.	The name of a JPO that has been added to the database. <i>If a JPO is configured for a tree category or root node label and the JPO returns an empty string or null, an exception is thrown.</i>
Maximum Length	form object field	Limits the number of characters that can be entered in a field with Input Type of textbox. If not specified, it uses the HTML default (unlimited).	Number of characters, for example: 30
Maximum Length	toolbar menu object	Limit the number of characters that are displayed on drop-down menu label. The Title property of the button is used as the Alt text to display the full label.	Any number of characters, such as 25 If this setting is not set, the toolbar displays the entire label.

Setting	Used For UI Component	Description	Accepted Values/Examples
Message URL Label	menu objects for navigation trees	Specifies the label to use for URL links that call the tree. For example, notifications that users have new tasks include URL links that open the Inbox Task tree for the user's task. By default, the URL link label shows the name of the Inbox Task object, which is an autogenerated name. But if the route creator entered a name for the task, it's better to show that entered name for the URL link. To have the URL link show the task's entered name, you would add the Message URL Label setting to the tree menu object for the Inbox Task. The value would be a select macro for the attribute that contains the entered name, in this case the Title attribute. When building the URL link, the system first looks for the Message URL Label setting on the app specific tree menu, for example, ENType_InboxTask. If not found there, the system looks for the setting in the common tree for that object type. If the common tree doesn't contain the setting, the system defaults to the object's Type,Name,Revision (for example, Inbox Task IT-0000101).	The value can be any select statement, plain text or a string resource ID. For example: • <code><attribute[attribute_Title].value></code> • Task Title • <code>emxEngineeringCentral.TaskTitle</code> The string resource Id can contain macros also, as in the following example: <pre>emxEngineeringCentral.TaskTitle = <type> <name> <revision>: Urgent Task</pre>
Nowrap	table object columns	If the setting is true for any column, then the text displayed in that column will not wrap at any time. The default is false.	–true –false
Popup Modal	command objects for menu links and tools, and toolbar items table object columns	If the setting Target Location is set to popup, the window can be configured as modal or non-modal.	true—the popup window is modal false—the popup window is non-modal If the setting is not specified, the window is modal.
Popup Modal	form object field	Use to configure whether the popup range helper window is model or non-modal. Only used when the setting Target Location is set to popup.	true—the popup window is modal false—the popup window is non-modal If the setting is not specified, the window is modal.

Setting	Used For UI Component	Description	Accepted Values/Examples
PreExpand	command objects for structure trees	Determines whether to let the JPO program decide whether to add the [+] icon or to simply add the [+] without running the JPO.	true—The Structure Program and Function defined for the Structure command is run to get the list of objects. Then for each object, it runs the same program to determine whether a [+] icon is needed. If the list of objects is greater than 1, a plus [+] is added to the structure node. false (default)—The JPO Program and Function is run to get the list of objects, but the same program and function is not run again for each object. Instead, a plus icon [+] is automatically added for each structure node.
PreExpand Category	command objects for tree categories	Used in conjunction with the Expand Program and Expand Function settings or the Expand Inquiry setting, which inserts a list of objects under the tree category without requiring users to first view details of those objects. Use this setting to determine whether the tree pre-processes the JPO or inquiry to determine if there are any objects to be listed. Pre-processing offers the benefit of determining in advance that there are no business objects in the list so the + sign for expansion is not shown. But pre-processing lowers performance. For more information, see Configuring Dynamic Expand for Tree Categories in Chapter 8. <i>Administrators who configure this setting as true need to be aware that there are performance implications to doing so. While it is nice to only display the + sign when data is present, the tree must actually go and get the data at the time the tree displays to do this. Therefore, if there are many tree categories configured with PreExpand Category set to true, each of those categories will be evaluated when the tree is displayed regardless of whether the user actually intends on viewing data in those categories or not. This potentially unnecessary processing lowers performance.</i>	true—Pre-processing is allowed so if there are no objects, the + sign for expanding the list is not shown. false (default)—Pre-processing is not allowed so the + sign is displayed even if there are no business objects in the list. The inquiry or JPO is not processes until the user clicks the + sign.

Setting	Used For UI Component	Description	Accepted Values/Examples
PreExpand Function	command objects for tree categories	Specifies a method in the JPO defined in the Expand Program setting that configures the category for dynamic expand. This method determines whether there are any business objects in the business object list retrieved from the JPO, thereby determining whether the + sign should be displayed. <i>Use of this setting requires a change in Matrix core that has not yet been implemented.</i>	Name of a method in the Expand Program JPO.
Printer Friendly	menu objects for navigation trees command objects for menu links and tools, tree categories, and toolbar items table object columns form object field	Passes the PrinterFriendly parameter and the entered value to the JSP specified in the href URL. JSPs that use the PrinterFriendly parameter, such as emxTable.jsp and emxForm.jsp, show the Printer Friendly tool when the setting is True and hide it when False. If the setting is not included, emxTable.jsp and emxForm.jsp show the tool by default. Users can click the tool to get a version of the current page that can be printed with the browser's Print button. Note that you can also specify the PrinterFriendly parameter in the href URL for JSPs that use it.	True False
program	form object field	Use to specify the name of the JPO program to use to get the field's data. This program is used to get the field values when the Field Type setting is set to "program" or "programHTMLOutput". Also can be used for fields defined with Field Type=emxTable to define the objects in the table. Alternatively, inquiry can be used. <i>Using a JPO to populate field data is recommended only when a select expression cannot be used to obtain the field value.</i> To see an example of a field that uses a program, see Field Values Obtained from Program in Chapter 8.	The name of a JPO, such as: • SCSBuyerDeskForm • TMCPackagesForm • ENCPartForm • AEFUtilForm

Setting	Used For UI Component	Description	Accepted Values/Examples
Program	table object column	The name of the JPO program to use to get the column's data. This program is used to get the column values when the setting "Column Type" is set to "program" or "programHTMLOutput" or "checkbox". <i>Using a JPO to populate column data is recommended only when a select expression cannot be used to obtain the values.</i> See Column Values as Program Output in Chapter 8.	emxSCSBuyerDesk emxTMCPackages emxENCParentPart emxCommonTableUtil
Pull Right	toolbar menu object	Determines whether the drop-down links are displayed in the same level as the menu or pulled right to display the links in the next level. Applies only to second-level drop-down menus. For examples, see Drop-Down Menus Can Display to the Right in Chapter 8.	True (default) False
Range Function	form object field table object columns	Use to specify the name of the method in the JPO specified in the Range Program setting. See the Range Program setting below for more information.	The name of a function in the Range Program JPO, such as: • getAssignedRange • getClassificationRange • getPartUOM
Range Program	form object field	Use to specify the name of a JPO that contains a method to get the field value ranges (choices). A range program is used only when the Input Type setting is combobox, radiobutton, or checkbox. If the choices need to be presented on a page in a popup window, for example in a chooser, use the RangeHref parameter instead. To see an example of a field configured with a range program, see Field for Attribute with Choices in Combo Box in Chapter 8. Also see Writing a JPO for Getting Field Range Values (Choices).	The name of a JPO, such as: • SCSBuyerDeskForm • TMCPackagesForm • ENCPartForm • AEFUtilForm
Range Program	table object columns	Use to specify the name of a JPO that contains a method to get the column value ranges (choices). A range program is used only when the Input Type setting is combobox or popup. The corresponding method which runs to get the values of the column should not have any HTML code.	The name of a JPO, such as: • SCSBuyerDeskForm • TMCPackagesForm • ENCPartForm • AEFUtilForm

Setting	Used For UI Component	Description	Accepted Values/Examples
Registered Suite	menu objects for top-level menus and for My Desk and Actions submenus command objects for menu links and tools menu objects for navigation trees table object columns menu objects for toolbars command objects for toolbar items	The application the UI component belongs to. The system looks for files related to the component in the registered directory for that application, which is specified in <code>emxSystem.properties</code>. Based on the application name, the system passes the following parameters in the href URL (if one is configured for the component): • <code>suiteKey</code> • <code>SuiteDirectory</code> • <code>StringResourceFileId</code> <i>This setting is required for all dynamic UI components.</i>	The value must be set without any spaces, for example, <code>EngineeringCentral</code> or <code>Framework</code>. The value must be set to the suite name as defined in the key <code>eServiceSuites.DisplayedSuites</code> within <code>emxSystem.properties</code>. If the suite name starts with “eServiceSuite” then this prefix can be skipped and assign the remaining text to the setting. For example, if the suite name in <code>emxSystem.properties</code> is “eServiceSuiteEngineeringCentral”, then the word “EngineeringCentral”, can be assigned as “Registered Suite”. In the href URL that is called when the item is clicked, the system passes a parameter called <code>suiteKey</code> and includes the property name that maps to the value specified for the setting.
Required	form object field	Use to indicate the value for this field is required. This setting is applicable for editable fields displayed on the Editable mode form.	true—When completing the form, the user must enter a value for the field. The field label appears in red italic text. If the user doesn’t enter a value and clicks Done, a JavaScript message appears that prompts the user to enter a value. false (default)—When completing the Edit form, the user can leave this field blank.
Row Select	command objects for toolbar items	Used only for actions on table pages. Specifies whether the JSP specified for the href expects one, at least one, or no rows in the table to be selected. If the Row Select setting is set to “single” or “multi”, the setting Submit must be set to “true” in order to get the details of the selected item(s) from table. If the setting is not specified, it is assumed to be none.	single—Appropriate only for actions that appear on table pages that have radio buttons or check boxes for each row. The JSP specified for the href expects exactly one row in the table to be selected. If a single row is not selected, the system asks the user to select a single row. multi—Appropriate only for actions that appear on table pages that have check boxes or radio buttons for each row. The JSP specified for the href expects at least one row in the table to be checked. If multiple rows are not checked, the system asks the user to select multiple rows. none (default)—The JSP specified for the href expects no row in the table to be selected. If a row is selected, the JSP does not use it.

Setting	Used For UI Component	Description	Accepted Values/Examples
Rows	form object field	Used when the input type is set to textarea. This setting limits the height of the textarea on the form and specifies the number of visible text lines. If not specified, it uses the HTML default, which is 5.	5 10 20
Section Level	form object field	Used in conjunction with the Field Type: Section Header setting to define the level of heading. To see an example, see Field as Section Header and Separator .	Two heading levels are available: 1 (default)—Font for heading label is large and a horizontal line is included above the label. 2—Font for heading label is smaller and the heading is in the same gray rectangle as standard fields.
Selectable in Preferences	command objects for menu links and tools	Only for commands in My Desk submenus. Controls whether the link is available as a preferred home page. If not set, the system assumes true and the command is listed as a home page preference. Commands that call pages to display in a popup window, such as an object search command, should not be available as a home page preference.	True False
Show Alternate Icon	table object column	If the column value to be displayed with the icon is different from the current row's object icon, set this setting to true. In order to get the right alternate icon, the Alternate Type expression setting must be defined with the expression to obtain the object type.	true or false
Show Alternate Icon	form object field	If the field value needs to be displayed with an icon that is different from the current object type icon, set this setting to true. In order to get the right alternate icon, the Alternate Type expression setting must be defined with the expression to obtain the object type.	true false (default)
Show Clear Button	form object field	Adds a Clear hyperlink next to the field. This setting only applies to textarea/textbox input type fields. When the Clear link is clicked, it clears the content of the textbox/textarea input type field.	True False

Setting	Used For UI Component	Description	Accepted Values/Examples
Show Type Icon	table object column	If set to true, the column value is displayed along with the object type icon as defined in the <code>emxFramework.smallIcon</code> property in <code>emxSystem.properties</code>. The icon is displayed to the left of the column data. If no property is defined for the type's icon, the system looks for a property defined for the parent type, then grandparent type and so on. If no property is defined for any type in the hierarchy, the system uses the default icon specified in the <code>emxFramework.smallIcon.defaultType</code> property.	The default is false. true or false
Show Type Icon	form object field	If set to true, the field value is displayed along with the type icon for the business object the form page currently displays information for. A type's icon is defined in the <code>emxFramework.smallIcon</code> property in <code>emxSystem.properties</code>. The icon is displayed to the left of the field data. If no property is defined for the type's icon, the system looks for a property defined for the parent type, then grandparent type and so on. If no property is defined for any type in the hierarchy, the system uses the default icon specified in the <code>emxFramework.smallIcon.defaultType</code> property.	true false (default) To see an example of a field with a type icon, see Field Values as Hyperlink and Type Icon in Chapter 8.
Sort Program	table object column	Specifies the JPO program name that contains a custom sorting algorithm. This setting is used only when the column parameter sorttype is assigned to "other". For guidelines and parameters that must be used for the custom sorting program and sample programs, see Custom Sorting Programs in Chapter 8.	Name of a JPO added as a program in the database, such as: FindNumberSort CustomSort
Sort Type	table object column	Determines how the column is sorted. If the sorttype parameter is used, this setting overrides the parameter.	date integer real
Sortable	table object column	Controls whether users can sort the table based on data in the table column. If a column is sortable, users can click the column heading to sort the table based on that column.	The default is true. true or false

Setting	Used For UI Component	Description	Accepted Values/Examples
sortColumnName	form object field	Used only for fields defined with Field Type=emxTable. Specifies the column by which the table should be sorted when the page is first loaded. If no column is specified, the rows are listed in the order they are retrieved from the database.	Name of column in the table specified in the table=TABLE_NAME setting.
sortDirection	form object field	Used only for fields defined with Field Type=emxTable. The direction to sort the rows by.	ascending (default)—Sort a to z or 0 to <i>n</i> . descending—Sort z to a or <i>n</i> to 0.
Structure Function	command objects for structure trees	JPO method name to be used for getting the object list.	The name of a method in the JPO specified in the Structure Program setting, such as: getWorkspaceFolders getEBOMs
Structure Inquiry	command objects for structure trees	Inquiry admin object name that gets the object list to display within the structure. This is an alternative to the JPO approach and the JPO approach takes precedence. <i>This setting is not implemented.</i>	The name of an inquiry admin object: SCSWspFolderStructure ENCEBOMList
Structure Menu	menu object for navigation tree	Specifies the menu admin object that defines a structure tree for the object type.	The name of a menu admin object.
Structure Program	command objects for structure trees	JPO program that contains the method to be used for getting the object list to display within the structure. The method is specified in the Structure Function setting. For instructions on writing the JPO program, see Guidelines for Writing Structure Tree JPO in Chapter 8.	The name of a JPO program added to the database. SCSWspaceStructure ENCBOMStructure
Submit	command objects for toolbar items	Specifies whether the system should send the object ID(s) (and relationship ID, if applicable) for the current page to the JSP specified in the href parameter. For form pages, the submitted data is just the object ID for the business object the page is about. For table pages, the submitted data is the ID for each selected item in the table, which can include object and relationships. In either case, the IDs are sent using the relBusIdList parameter. If the setting is not specified, it is assumed to be false.	true—Submits the IDs to the URL specified in the href parameter. Should be used for actions that depend on selected table rows. false—The URL specified in the href parameter is directly called from the toolbar item and no IDs from the current page are used. Use this value for links that do not require the business object IDs, such links for creating a new object.

Setting	Used For UI Component	Description	Accepted Values/Examples
table	form object field	Specifies the name of a table admin object to embed in the form. Used only for fields defined with Field Type=emxTable. See Field that Embeds a Configurable Table in the Form in Chapter 8.	Name of table admin object. For example: table=SCSBuyerDesk table=ENCParts
Target Location	command objects for menu links and tools table object columns command objects for toolbar items	Controls where the page specified in the href parameter appears.	content—The page replaces the tree frame. mainframe—The page appears in the main content frame, to the right of the tree frame and tab frames, replacing the page that is already there. (not used for table columns) popup—Page appears in new window. The window modality is set using the Popup Modal setting. _top—Page replaces the entire body of the browser window. topFrame—The page replaces the banner frame. (not used for table columns) hiddenFrame—The frame called hiddenFrame is part of the top level Navigation window frameset and can be used to submit the Table frame and carry out background processing. This frame is not available for popup windows though so listHidden is a better frame to use. listHidden—A hidden frame within the table frameset. Use this frame for Target Location for any processing in the context of a table. For table columns, this value can be set to any valid frame name, which is available to the table body frame.

Setting	Used For UI Component	Description	Accepted Values/Examples
Target Location	form object field	Controls where the page specified in the href parameter appears when a user clicks the hyperlinked data.	popup—Page appears in a new window. The window modality can be set with the Popup Modal setting. Popup is the default value and is used if the setting is not included or if the named frame cannot be found. content—The page replaces the content frame. mainFrame—Page appears in the frame that includes the content and menu frames. _top—Page replaces the entire body of the browser window. This value can be set to any valid frame name that is available to the form body frame.
Tip Page	menu objects for navigation trees command objects for menu links and tools, tree categories, and toolbar items table object columns form object field	Specifies whether the target page should include the Tip Page tool and the URL that should be called when the tool is clicked. In the href URL called when the UI component is clicked, the system passes a parameter called TipPage and includes the URL specified for this setting. The target page includes the Tip Page tool and when clicked, the specified URL is executed. If this setting is not included, the target page does not display the Tip Page tool.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the table.jsp: TipPage=../doc/customcentral/tippage.html

Setting	Used For UI Component	Description	Accepted Values/Examples
Tree Scope ID	menu objects for tree nodes	Passes the current tree's objectId to all its categories, to subtrees inserted into the tree, and to categories in the subtrees, to the nth level, in the parameter name specified as the value for this setting. This setting is valid only for tree menu objects and not for tree category command objects. For example, suppose you want to pass the object ID for the current workspace to all categories and subtrees in the workspace tree. You want to pass the object ID using a parameter called workspaceId. To do this, you would add the Tree Scope ID setting to the menu admin object for workspace trees (which is called type_Project due to admin object renaming) and enter "workspaceId" as the value. The parameter and value "workspace Id=OBJECTID OF THE WORKSPACE will be available to all the categories and subtrees of the current workspace to the nth level until another workspace object gets inserted or until any explicit URL parameter with the same name "workspaceId" is passed to the tree.	The name of the parameter you want to use to pass the object ID. For example: workspaceId projectId partId <i>You can also pass specific parameters from the tree menu object to its categories using the AppendParameter parameter for the emxTree.jsp.</i>
UOM Expression	table object columns	The name of the Unit of Measure to convert from. Required for columns whose values are measurements that can be converted from English units to Metric units or vice versa. The value can be a select clause that returns the value for the Unit of Measure attribute for a specific object or relationship or the actual value. The unit of measure to convert form should be the unit of measure in which the user entered the data (the As Entered unit of measure).	For example, for measurement data for RFQ Quotations, the following select clause returns the supplier's unit of measure format. <pre>to[Supplier Line Item].attribute[Unit of Measure]</pre>
Update Function	form fields table object columns	Specifies a method name in the JPO given in the Update Program setting. See Update Program for more information.	The name of a function in the Update Program JPO, such as: • setAssignedBuyerDesk • setPackage Access • setPartClassification

Setting	Used For UI Component	Description	Accepted Values/Examples
Update Program	form fields table object columns	Specifies a JPO that contains a method to set the field value when the field is displayed on an Edit mode form and when the user clicks Done. This program is used only when the Field Type setting is program or programHTMLOutput. <i>Using an update program is recommended only when the Field Type is not attribute or basic.</i> See Writing a JPO for Updating Field Values in Chapter 8.	The name of a JPO, such as: • SCSBuyerDeskForm • TMCPackagesForm • ENCPartForm • AEFUtilForm
Validate	form object field	Use to specify the name of a Javascript function that validates the fields before the form is updated.	Name of Javascript function, such as: checkDate checkNumberFormat
Validate Type	form field object	This setting is used to validate any field value with any specific characters defined in emxSystem.properties.	Basic—The field values are validated against the value of emxFramework.Javascript.BadChars. Name—The field values are validated against the value of emxFramework.Javascript.NameBadChars. Restricted—The field values are validated against the value of emxFramework.Javascript.RestrictedBadChars.
Window Height	command objects for menu links and tools command objects for toolbar items	The window height for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	400 800
Window Height	form object field	Use to define the height of the popup range helper window.	Number of pixels, such as 700. The default is 600.
Window Width	command objects for menu links and tools command objects for toolbar items	The window width for the new popup window. This value is used only when the setting for Target Location is set to “popup”.	400 800
Window Width	form object field	Use to define the width of the popup range helper window.	Number of pixels, such as 700. The default is 600.

URL Parameters

This table lists the parameters that you can specify for configurable JSPs.

A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z

Parameter	Accepted by JSP	Description	Accepted Input Values
AppendParameters	emxTree.jsp	Passes common parameters to the tree categories in the tree. To use the parameter, set it to tree and then include the parameters to pass. Parameters added to emxTree.jsp that are not with AppendParameters are passed only to the root tree node and not to categories. <i>You can also use the Tree Scope ID setting for the tree menu object to pass the current tree's object ID to all categories and subtrees.</i>	To make Param1 and Param2 available to the children categories: <pre><a href=" ../common/emxTree.jsp?objectId=<%=sPartId%>&mode=insert&jsTreeID=<%=jsTreeID%>&AppendParameters=true&Param1=Value1&Param2=Value2" class="object" target="content"><%=sPartName%></pre> To make ProjectId and Workspace available to the tree categories: <pre><a href=" ../common/emxTree.jsp?objectId=<%=sPartId%>&mode=insert&jsTreeID=<%=jsTreeID%>&AppendParameters=true&ProjectId=<%=sProjId%>&Workspace=<%=sWorkspace%>" class="object" target="content"><%=sPartName%></pre> To make ProjectId and Workspace available only to the root tree node and not to the tree categories: <pre><a href=" ../common/emxTree.jsp?objectId=<%=sPartId%>&mode=insert&jsTreeID=<%=jsTreeID%>&&ProjectId=<%=sProjId%>&Workspace=<%=sWorkspace%>" class="object" target="content"><%=sPartName%></pre>
CancelButton	emxTable.jsp	Displays the Cancel link in the table footer frame. When a user clicks Cancel, the whole browser window gets closed. So the Cancel link can only be used for a table page that is opened in new window. The label for the Cancel link can be configured using the CancelLabel parameter. If CancelLabel is not passed in, the default label "Cancel" is used.	true—Shows the Cancel link. Use only for tables displayed in popup windows. false (Default)—Does not show the link.

Parameter	Accepted by JSP	Description	Accepted Input Values
CancelLabel	emxTable.jsp	Defines the label for the Cancel link, if you want something other than “Cancel”.	Any static text or string resource id. CancelLabel=emxEngineeringCentral.Common.Cancel
CommandName	emxNavigator.jsp	Use along with the MenuName parameter to define the page that should be displayed in the Content frame of the Navigator page when users log in. The MenuName parameter defines the name of a menu object that represents a submenu and the CommandName parameter defines the name of a command object from which the href URL is to be obtained. Note that the links in the Actions tab submenus cannot be used for display in the content frame.	The name of a command object that is assigned to a menu object that represents an application submenu on the My Desk tab. For example: emxNavigator.jsp?MenuName=TMCMYDesk&CommandName=TMCRoutesMyDesk Where TMCMYDesk is the name of the menu object for the Team Central My Desk submenu and TMCRoutesMyDesk is the name of the command object for the Routes link. This command object contains an href parameter that points to the JSP for the Routes page.
ContentPage	emxNavigator.jsp	Defines the JSP that should appear in the Content frame (called the Home page in the user interface) when the Navigator page is called. For more information on defining the page displayed in the Content frame, see Configuring the Default Home Page for the Content Frame in Chapter 8.	Any JSP plus parameters: emxNavigator.jsp?ContentPage=../engineeringcentral/emxengchgInboxFrameset.jsp <i>Currently, the URL assigned to the ContentPage parameter can contain only one parameter/value pair as part of the URL. Any additional parameters are ignored.</i>
DefaultCategory	emxTree.jsp	Specifies the category that should be selected when the tree first opens or is first inserted into another tree. By default, the root node is selected, which means the object’s Properties page displays. If the tree contains a category that users frequently want to see, you can use this parameter to have it selected instead. Alternatively, you can use the Default Category setting for the tree’s menu object. If both are defined, this URL parameter overrides the setting. Note that a tree does not look for default categories defined for its sub-trees (assigned sub-menus).	The name of a category command object that should be selected when the tree first opens. The command object must be assigned to the tree menu object and if it is not, the root node is selected. DefaultCategory=PMCWBS
editLink	emxTable.jsp emxForm.jsp	Use to display the Edit toolbar item on the table or form page. The Edit link is automatically configured to call the editable version of the current table or form.	true false (default) emxForm.jsp?form+ENCPart&editLink=true

Parameter	Accepted by JSP	Description	Accepted Input Values
emxSuiteDirectory	configurable pages, many standard pages, most custom pages	The name of the directory for the application under ematrix.	engineeringcentral
ExclusionList	emxTypeChooser.jsp	Use to define the types to exclude from the top-level list of types that display when the Type Chooser opens. The inclusion list and the exclusion list cannot both be passed as parameters. If they are, an error message is displayed. If neither is specified, all top-level types are listed.	Same as InclusionList.
Export	emxTable.jsp	Shows or hides the Export  tool on the page toolbar. When users click the tool, the system exports the table data to a file in the user's preferred format in their preferred format: CSV, HTML, or Text. Users choose their preferred format using the Preferences tool in the global toolbar. For information on configuring the export preference, see Export Table Data Format in Chapter 8. For instructions on using the preference page, click Help from the preference page.	true (Default)—The Export tool displays, allowing users to export the table data. false—The tool does not display.
FilterFramePage	emxTable.jsp	Specifies the name of the JSP that will be used in the custom filter frame, which is between the header and body. You can use the directory macros to refer the location of the page. The path of the page is relative from the Apps/Framework/VERSION/commonUI directory. For details on implementing the custom filter, see Implementing a Custom Filter in Chapter 8.	<code>\${SUITE_DIR}/emxTeamProjectTableFilter.jsp</code> <code>\${ROOT_DIR}/emxPartECOSatusFiletr.jsp</code> <code>\${COMMON_DIR}/emxCommonStateFilter.jsp</code> <code>emxCommonStateFilter.jsp</code>
FilterFrameSize	emxTable.jsp	Controls the size of the filter frame in pixels. The value should be at least 40, which is the default.	40 80 160

Parameter	Accepted by JSP	Description	Accepted Input Values
fieldNameActual	emxTypeChooser.jsp	Required for the Type Chooser. Use to specify the field on the form page to populate the type(s) the user selects in the Type Chooser. The system populates the specified field with the actual Matrix type names, as stored in the database, so the field must be a hidden field. When calling the Type Chooser using a RangeHref on a configurable Form page, the hidden field is created automatically and has the same name as the RangeHref field. Other pages that need to get the type name selected by the user, for example to perform a search, should get the type name from this field instead of the fieldNameDisplay.	The name of a hidden field on the form page. For example, if the form page contains a field defined as follows: <pre><input type="hidden" name="txtSelectedTypeName" value=""></pre> This parameter should be passed to emxTypeChooser.jsp: <pre>fieldNameActual=txtSelectedTypeName</pre>
fieldNameDisplay	emxTypeChooser.jsp	Required for the Type Chooser. Use to specify the text field on the form page to populate the type(s) the user selected in the Type Chooser and display these types to the user. When calling the Type Chooser using a RangeHref on a configurable Form page, the display field is created automatically and has the same name as the RangeHref field with "Display" appended to it. This field contains the internationalized version of the selected type list.	The name of a non-hidden field on the form page. For example, if the form page contains a field defined as follows: <pre><input type="text" name="txtType" value="" onClick="showTypeChooser(")></pre> This parameter should be passed to emxTypeChooser.jsp: <pre>fieldNameDisplay=txtType</pre>
form	emxForm.jsp	Specifies the web form admin object used for presenting the form page.	Name of web form admin object. emxForm.jsp?form=ENCPart
formHeader	emxForm.jsp	Use to define the content of the heading that appears at the top of the form page, in the header frame. The value can be either a string resource id or the label itself. The text in the label and string resource ID can contain any valid select expression.	<pre>emxForm.jsp?form=ENCPart&header=Properties emxForm.jsp?form=ENCPart&header=Properties:<type><name> emxForm.jsp?form=ENCPart&header=emxQuoteCentral.AssignedPackages.AssignedPackages</pre>

Parameter	Accepted by JSP	Description	Accepted Input Values
formName	emxTypeChooser.jsp	Use to specify the name of the form that holds the field specified in fieldNameDisplay. If this parameter is not passed in, the Type Chooser looks for the field name on the first form of the form page. If the field name is not found in the first form, then an error message will be presented.	The name of a form on the form page. For example, if the form page contains a form defined as follows: <pre><form name="searchPage" method="post" action="somejsp.jsp"></pre> This parameter should be passed to emxTypeChooser.jsp: <pre>formName=searchPage</pre>
frameName	emxTypeChooser.jsp	Use to specify the name of the frame that contains the form on the form page. Needed for pages that contain forms in multiple frames.	The name of a frame on the form page.
header	emxTable.jsp	The content of the heading that appears at the top of the table page.	Any alphanumeric text or a string resource ID. header=Buyer Desk header=emxQuoteCentral.AssignedPackages.AssignedPackages
Header	emxHistory.jsp	The text to use for the header of the History page. This parameter can either be a string resource ID or a mixture of macros and text or simply text.	<pre>\$(type) \$(name) \$(revision)</pre> emxFrameworkStringResource.Common .HistoryPageHeading Object History
header	emxPortal.jsp	The content of the heading that appears at the top of the portal page.	Any alphanumeric text or a string resource ID. header=Summary View header=emxSpecificationCentral.SCOSummaryView.Summary View
headerRepeat	emxTable.jsp	Specifies how often the header row should be repeated. A repeating header row makes it easier for users to identify the content of each column when scrolling through a long table. If the parameter is not used, the header row repeats every 15 rows (or whatever number is assigned to the emxSystem.properties key emxNavigator.UITable.HeaderRepeat).	Integer For example, 11 would mean the header row is repeated every 11 rows. To turn off headerRepeat, pass in 0 as the value for the parameter.
HelpMarker	emxTable.jsp and most standard pages	Specifies the name of the help marker to call for context-sensitive help. For information about implementing help, see Implementing Context-Sensitive Help in Chapter 8.	String The naming convention for help markers “emxhelp” followed by the object or feature and then the action, for example, emxhelproucreate and emxhelpprojectedit. The marker is all lowercase with no spaces.

Parameter	Accepted by JSP	Description	Accepted Input Values
HistoryMode	emxHistory.jsp	Determines whether the entire revision chain or just the current revision history should be displayed.	AllRevisions—The page displays the entire revision history of the object. CurrentRevision (default)—The page displays the history for the particular revision that is being reviewed. For example, a part may have REV A, Rev B and Rev C. If we are reviewing REV B and the HistoryMode is set to “CurrentRevision”, only the history for REV B is presented. If HistoryMode is set to “AllRevisions”, then the history for REV A, B, C is presented.
InclusionList	emxTypeChooser.jsp	Use to define the top-level list of types to display when the Type Chooser opens. Note that these types do not have to be top-level types (types with no parents), they will just be listed in the top level in the chooser. The inclusion list and the exclusion list cannot both be passed as parameters. If they are, an error message is displayed. If neither is specified, all top-level types are listed.	- a properties file key If the SuiteKey parameter is passed in, the system looks for the key in the application-specific properties file (for example, emxEngineeringCentral.properties). If the property is not application specific, the system looks in emxSystem.properties. For example, if emxEngineeringCentral.properties contains this property: <i>eServiceEngineeringCentral.Types =type_Part,type_ECR,type_Sketch</i> The parameter to pass would be InclusionList=eServiceEngineeringCentral.Types - a comma delimited list of symbolic names for the types to include: InclusionList=type_Part,type_ECR
inquiry	emxTable.jsp	Specifies the inquiry admin object that should be used to retrieve the business objects to be included in the table and any inquiries for the table page filter. emxTable.jsp uses the first inquiry object to display the list in the table when the page is first loaded. It uses all inquiry objects for the table filter list in the header. If there is only one inquiry, the filter in the header is not displayed. Currently, to build the filter list, you must use either JPOs or Inquiry objects. You cannot use both.	Names of inquiry admin object separated by a comma. inquiry=SCSBuyerDesk,SCSBuyerDesk Assigned inquiry=ENCAAllParts, ENCREleasedParts

Parameter	Accepted by JSP	Description	Accepted Input Values
inquiryLabel	emxTable.jsp	This parameter is used with the “inquiry” parameter. Specifies the labels for each inquiry in the filter list in the table page header. Each label is associated to the corresponding inquiry admin object name specified in the “inquiry” parameter. The label may have the actual text or the string resource id for internationalization.	One or more labels (equal to the number of objects specified in the “inquiry” parameter) for the inquiry admin object names separated by a comma. inquiryLabel=All,Assigned inquiryLabel=emxEngineeringCentral.Common.All,emxEngineeringCentral.Common.Released
jsTreeID	emxTree.jsp, configurable pages, many standard pages, most custom pages	The ID of the existing tree object, which is required to update the tree with “insert” mode. - When the tree is constructed for the first time, the “jsTreeID” parameter will not have any valid value. - Whenever a category within the tree is clicked, the system passes the current “jsTreeID” to that page and if that page needs to update the tree, it must call emxTree.jsp with the same “jsTreeID” value.	Active NodeId, for example, root_7 or root_7_0_6
massUpdate	emxTable.jsp	Use to turn on or off mass update controls on the editable table page.	true (default)—Mass update is available for the page. false—Mass update not is available for the page.
MenuName	emxNavigator.jsp	Defines which application submenu should be expanded when the Navigator page is called. If you don’t specify a submenu, the first menu in the My Desk will be expanded. Can also be used along with the CommandName parameter to define the page to appear in the Content frame on login. For more information, see the CommandName parameter above.	The name of a menu object that represents an application submenu. For example: emxNavigator.jsp?MenuName=TMCMYDesk
MenuTab	emxNavigator.jsp	Defines which tab is selected when the Navigator page is called: My Desk or Actions. If the parameter isn’t included with the URL, the My Desk tab is selected.	Actions—The Actions tab is selected. My Desk—The My Desk tab is selected. For example: emxNavigator.jsp?MenuTab=Actions

Parameter	Accepted by JSP	Description	Accepted Input Values
mode	emxForm.jsp	Specifies whether the form page should be view or edit mode.	view (default)—form is read only edit—form is editable emxForm.jsp?form=ENCPart&mode=edit
mode	emxTree.jsp emxForm.jsp	Controls whether the tree for the object is inserted into the current tree or replaces the current tree. For information, see Two Modes for Tree Display .	Insert (default)—Insert the tree for the selected object into the current tree, if there is one. Replace—Replace the current tree, if there is one, with the tree for the selected object.
mode	emxNavigator.jsp	Used when calling the Navigator page from an external link on a Web site, portal page, or email message. Defines whether the Navigator window contains the banner, toolbar, My Desk tab, and Actions tab as it normally does when using the applications. For more information on calling the Navigator externally, see Accessing the Applications Externally in Chapter 8.	Menu—The Navigator page contains all standard elements. Tree—The Navigator page contains only the tree frame and the content frame. For example: emxNavigator.jsp?mode=Menu&ContentPage=../sourcingcentral/emxBuyerDeskTable.jsp
objectBased	emxTable.jsp	Use to configure a table page so it lists items other than business objects. For example, the items such as collections and IconMail message are not business objects but can be listed in a table by passing the URL parameter objectBased=false.	true (default)—The table page lists business objects. false—The table page lists items that are not business objects.
objectId	emxForm.jsp	Use to specify the business object for which the properties need to be displayed.	2233.5567.2323.4678 For example: emxNavigator.jsp?objectId=29547.17050.1266.35700
objectId	emxTree.jsp emxNavigator.jsp configurable pages, many standard pages, most custom pages	When used with emxTree.jsp, defines the business object ID for which the tree will be constructed. For example, if the object is of type ECR, the tree object of menu type “type_ECR” will be constructed. If the menu “type_ECR” does not exist, the tree will be based on the “Default Tree” configuration. When used with emxNavigator.jsp, defines the business object whose navigation tree should be displayed when the Navigator window is called.	Valid business object ID. emxForm.jsp?objectId=3243.32424.232

Parameter	Accepted by JSP	Description	Accepted Input Values
ObserveHidden	emxTypeChooser.jsp	Determines whether to display hidden types or not.	true (default)—Hidden types are included in the chooser's list of types. false—Hidden types are not included.
pagination	emxTable.jsp	Specifies the number of rows to show per page. If the rows span more than one page, users can use the Left and Right Arrow buttons or the Page drop-down list to navigate to other pages. For more information on pagination controls, see How the Pagination Controls Work in Chapter 8. If this parameter is not specified, the system uses the value set in the pagination property in emxSystem.properties. When installed, this property is set to 10 rows per page. If 0 is specified, all objects are listed on one page.	Any number. Use 0 if you don't want to paginate. If 0 is specified, the pagination controls do not include the controls for navigation.
parentOID	configurable pages, many standard pages, most custom pages	Passed to toolbars if there is a valid objectId passed in to emxTable.jsp (in case of tree calling the table page).	Business object ID, for example, 46697.12656.52860.3882
portal	emxNavigator.jsp	Used when calling the Navigator page from an external link to ensure the Login page doesn't replace the frame that contains the link. For more information on calling the Navigator externally, see Accessing the Applications Externally in Chapter 8.	true—Turns the portal mode on to ensure the Login page and Navigator page have no effect on the page that contains the URL link to the Navigator page. false—Turns the portal mode off. If the parameter isn't specified, the portal mode is turned off. ./Apps/Framework/VERSION/commonUI/emxNavigator.jsp?portal=true
portal	emxPortal.jsp	Specifies the menu admin object that represents the portal page.	Name of menu admin object defined to represent a portal page.

Parameter	Accepted by JSP	Description	Accepted Input Values
portalTable	emxTable.jsp	Specifies the alternate table that can be used within a channel tab. Displaying table component listings in half page-width channels will typically cause the user to scroll horizontally to the view all columns in the table. For this situation, you can create an alternate table with fewer columns for display specifically within a channel tab. For example, if a standard ECO New Parts table contains 10 columns, you could define an alternate ECO New Parts table that contains just 4 columns.	Name of alternate table
preFilter	emxHistory.jsp	Determines which action types to display when the History page comes up. If the parameter is not passed, the page lists all action types. If the Action Type filter control is shown on the page (determined by the ShowFilterAction parameter), the preFilter actions are listed in the text box and the filter is applied to the history list when the page first opens. The user can use the Action Type filter to change the filter, adding and removing displayed actions as needed. If the Action type filter control is not shown, the user cannot change the preFilter actions. By setting preFilter actions and hiding the Action Type filter, you can hide specific actions that you do not want users to see.	The parameter accepts two types of values: - Comma delimited list of action types: preFilter=connect, disconnect, create - String Resource ID: preFilter=emxSystem.preFilter.List To specify which string resource properties file to look in to get the value, include the SuiteKey parameter. Possible values for the SuiteKey parameter are: eServiceSuiteMatrixCentral, eServiceSuiteInfoCentral, eServiceSuiteTeamCentral, eServiceSuiteEngineeringCentral, eServiceSuiteProgramCentral, or any other suite name. If the SuiteKey is not passed in, the History page looks for the key in the emxSystem.properties file. For example, in the emxSystem.properties file, the value for the emxSystem.preFilter.List might be "connect, create, disconnect".

Parameter	Accepted by JSP	Description	Accepted Input Values
PrinterFriendly	emxTable.jsp emxForm.jsp configurable pages, many standard pages, most custom pages	Specifies whether the page should include the Printer Friendly tool. If this parameter is not included, the value is assumed to be True and the page displays the Printer Friendly tool. You can have the system pass this parameter automatically by entering the PrinterFriendly setting for the command object that calls the page.	true (default) false
program	emxTable.jsp	Used only when there is no “inquiry” parameter is passed in to emxTable.jsp. This parameter is used as an alternative approach to the inquiry object approach. The “program” approach uses a JPO program object to fetch the object list to be displayed in the table. The parameter value can have one or more sets of values separated by a comma “,”. emxTable.jsp uses the first set of values (JPO name and method name) to display the list in the table when the page is first loaded. It uses all the sets for the table filter list in the header. Currently, to build the filter list, you must use either JPOs or Inquiry objects. You cannot use both. If there is only one program, the filter in the header is not displayed.	This parameter is assigned to one or more set of values that will form a JPO program name and the method name. The format of the parameter value is: program=<JPO program name>:<JPO method name> The program name and the method name are separated by a colon “:”. program=emxTableBuyerDesk:getBuyerDesk,emxTableBuyerDesk:getAssignedBuyerDesk For guidelines on writing a JPO for getting a list of business objects, see Guidelines for Writing a JPO for Getting Object List in Chapter 8. For a sample of a JPO that gets a list of business objects, see Sample JPO for Getting List of Objects in Chapter 8.
programLabel	emxTable.jsp	Used with the parameter “program” defined above. Specifies the labels for each program in the filter list in the table page header. The value may contain one or more labels (equal to number of “program” values) separated by a comma “,”. Each label is associated to the corresponding JPO object and method name specified in “program” parameter. The label may have the actual text or the string resource id for internationalization.	programLabel =All,Assigned programLabel = emxEngineeringCentral.Common.All, emxEngineeringCentral.Common.Released

Parameter	Accepted by JSP	Description	Accepted Input Values
RegisteredDirectory	emxForm.jsp	The directory where the help files are located.	Directory name within ematrix.
relID	emxForm.jsp configurable pages, many standard pages, most custom pages	Use to specify the relationship used to get the relationship attribute values for relationship fields.	Valid relationship ID. emxForm.jsp?relId=3243.32424.232
ReloadOpener	emxTypeChooser.jsp	Determines whether to call a reload() method on the opener/parent page (the form page from which the choose is launched). Reloading the page updates other fields on the form page based on the type(s) selected from the Type Chooser.	true—The reload method is called. false (default)—The reload method is not called.
rememberSelection	emxTable.jsp	Controls whether the system remembers the items a user selects on one page when the user navigates to another page in the table. The emxSystem.properties key emxNavigator.UITable.Pagination.RememberSelection sets the default for table pages. Use this parameter to override the default for a single page. For more information, see How the Pagination Controls Work in Chapter 8.	true—Selected items are remembered across the table's pages. false—Selected items are not remembered across pages.
SelectAbstractTypes	emxTypeChooser.jsp	Determines whether users can select abstract types.	true—Abstract types can be selected. false (default)—Abstract types cannot be selected but they are displayed in the hierarchical list of types.

Parameter	Accepted by JSP	Description	Accepted Input Values
selection	emxTable.jsp	Controls whether the table page adds a column of check boxes or radio buttons in the left-most column of the table.	multiple—Users can select more than one row in the table. A check box is displayed in the left column of each row. There is no access restriction for the check boxes. If this parameter is passed, it overrides any check box column added to the table admin object. For more information, see Column Values as Check Boxes in Chapter 8. single—Users can select one row in the table. A radio button is displayed in the left column of each row. none—A selection column is not added to the table page. (A check box column can still be displayed by adding the column to the table admin object.)
SelectType	emxTypeChooser.jsp	Determines whether the Type Chooser has single select radio buttons or multi-select check boxes.	multiselect—Users can choose multiple types using check boxes. singleselect (default)—Users can choose only one type using a radio button.
ShowFilterAction	emxHistory.jsp	Determines whether to display the action type filter or not.	true (default)—The action type filter displays. false—The action type filter does not display.
ShowFilterTextBox	emxHistory.jsp	Determines whether to display the filter text box or not.	true (default)—The filter text box displays. false—The filter text box does not display.
ShowIcons	emxTypeChooser.jsp	Determines whether to display types icons in the Type Chooser.	true (default)—Type icons are displayed. false—Type icons are not displayed. This improves performance.
sortColumnName	emxTable.jsp	Specifies the column by which the table should be sorted when the page is first loaded. If no column is specified, the rows are listed in the order they are retrieved from the database.	Name of table column.
sortDirection	emxTable.jsp	The direction to sort the rows by.	ascending (default)—Sort a to z or 0 to <i>n</i>. descending—Sort z to a or <i>n</i> to 0.
StringResourceFileId	configurable pages, many standard pages, most custom pages	The name of the String Resource file for the application.	emxFrameworkStringResource

Parameter	Accepted by JSP	Description	Accepted Input Values
Style	emxTable.jsp	Determines the cascading style sheet to use for the table page.	list (default)—For table pages displayed in the content frame of emxNavigator.jsp. By default, the standard emxUIList.css in Chapter 8 style sheet is used. The style sheet to use is defined in this emxSystem.properties key: emxNavigator.UITable.Style.List = styles/emxUIList.css dialog—For table pages that display in popup windows, such as search result pages. By default, uses the emxUISearch.css in Chapter 8 style sheet. The style sheet to use is defined in: emxNavigator.UITable.Style.Dialog = styles/emxUISearch.css
subHeader	emxHistory.jsp	Only needed when using All Revisions mode. Determines whether to display Version or Revision for the SubHeader, which separates the history events for each revision.	Version—The History page looks in the emxFrameworkStringResource.properties file for value of the emxFramework.History.Version property. Revision—The History page looks in the emxFrameworkStringResource.properties file for value of the emxFramework.History.Revision property.
subHeader	emxTable.jsp	Creates a subHeader below the main header in the table header frame.	The value can be any static text or a string resource id. For example: subHeader= emxEngineeringCentral.Common.BOM Level subHeader=Bill of Material Level 1 The value can also include macros such as \$<type> \$<revision>.
SubmitURL	emxTable.jsp	Displays the Submit link in the table footer frame. The value can be any valid JSP page, which gets run upon clicking Submit. The label for the Submit link can be configured using the SubmitLabel parameter. If SubmitLabel is not passed in, the default label “Submit” is used.	Any JSP. The path can include any directory macro. SubmitURL=\${SUITE_DIR}/ emxBlank.jsp
SubmitLabel	emxTable.jsp	Defines the label for the Submit link.	Any static text or string resource id. SubmitLabel=emxEngineeringCentral.Button.Next

Parameter	Accepted by JSP	Description	Accepted Input Values
SuiteKey	emxHistory.jsp configurable pages, many standard pages, most custom pages	Determines which string resource property file to look in for the preFilter property key. This parameter is only used when the preFilter parameter is passed in as a resource ID. For example, SuiteKey=eServiceSuiteEngineeringCentral would be required for the preFilter parameter whose value is in the EngineeringCentral properties file, emxEngineeringCentral.properties.	eServiceSuiteMatrixCentral, eServiceSuiteInfoCentral, eServiceSuiteTeamCentral, eServiceSuiteEngineeringCentral, eServiceSuiteProgramCentral, or any other suite name.
	emxTypeChooser.jsp	Determines which application's properties file to look in for the key specified in the InclusionList or ExclusionList parameter. If the SuiteKey is not passed in, emxSystem.properties is used.	
table	emxTable.jsp	Specifies the table object to use for presenting this targeted page. The table object has all the information needed to display the table, including the columns to present.	Name of table admin object. For example: table=SCSBuyerDesk table=ENCParts
TipPage	emxForm.jsp	Specifies the Web page (html or jsp) that should be launched when the user clicks one of the page toolbar buttons in the form header frame.	Name of a custom URL page. emxForm.jsp?form=ENCPart&TipPage=../myapplication/showMyTipPage.jsp?
TipPage	emxTable.jsp emxPortal.jsp configurable pages, many standard pages, most custom pages	Specifies whether the page should include the Tip Page tool and call a specific html or jsp when a user clicks the tool. If this setting is not included, the Tip tool is not included on the page.	Name of a custom html or JSP page, including any path. The starting point for the directory reference is the content directory. For example, if you want to call an html file in ematrix/doc/customcentral and the content directory is ematrix/customcentral, you would add this parameter to the jsp: TipPage=../doc/customcentral/tippage.html
toolbar	emxForm.jsp emxPortal.jsp	Specifies the menu admin object that represents the Actions menu, which appears in the page header.	Name of menu admin object defined to represent the Actions menu.
TransactionType	emxTable.jsp	Controls whether the table query is run within an Update transaction.	read (default)—The table query is not run within an Update transaction. update—The table query (inquiry or JPO) will be run within an Update transaction.

Parameter	Accepted by JSP	Description	Accepted Input Values
treeLabel	emxTree.jsp	Use to override the default label for a tree, which is defined in the “Label” parameter of the tree menu admin object. When this parameter is available to the emxTree.jsp page, the page uses this value as the Label for tree menu base node. This parameter also overrides any JPO specified for the label in the settings Label Program and Label Function.	emxTree.jsp?treeLabel=custom Part Label&objectId=3434.345.4564.7755
treeMenu	emxTree.jsp	Use to override the standard tree menu for the object’s type (or any alternate tree defined by an application). The standard tree menu for an object is named using the symbolic name of the type. For example, the standard tree menu object for quality part plans is type_QualityPartPlan. For more information on calling specific trees, see Calling Specific Trees for an Object Type in Chapter 8. <i>Using the custom tree menu by overriding the system-defined tree for any object type is NOT RECOMMENDED.</i>	Name of a menu object for a tree: customPartTree

Parameters Automatically Passed to URLs

When a user chooses a command—such as a menu option or tree category—the system automatically appends one or more parameters to the URL specified in the href parameter for the object. The target URL can use these parameters if needed or not. The configurable pages and many standard pages use one or more of these parameters and most custom pages will also. For information about which parameters are used by a JSP, refer to the section that describes the JSP in [Chapter 8, Configuring the User Interface](#).

Parameter	Description	Values Sent
CurrencyConverter	Passes the value entered for the Currency Converter setting. If True, the target page should include the Currency Converter tool. If the setting is not included, the value is assumed to be false and the target page does not display the Currency Converter tool.	Value specified in Currency Converter setting: True False
emxSuiteDirectory	The name of the directory for the application under the ematrix directory.	engineeringcentral
HelpMarker	Passes the help marker text specified for the Help Marker setting.	Value specified in Help Marker which is a help marker for the page.
jsTreeID	The ID of the existing tree object, which is required to update the tree with “insert” mode. - When the tree is constructed for the first time, the “jsTreeID” parameter will not have any valid value. - Whenever a category within the tree is clicked, the system passes the current “jsTreeID” to that page and if that page needs to update the tree, it must call emxTree.jsp with the same “jsTreeID” value.	Active NodeId, for example, root_7 or root_7_0_6
objectId	The business object ID for the selected object. JSPs use the objectId to display information for that object. For example, emxTree.jsp uses it to construct the tree for that object type. If the object is of type “ECR”, the tree object of menu type “type_ECR” will be constructed.	Business object ID, for example, 46697.12656.52860.3882
parentOID	Passed to toolbars if there is a valid objectId passed in to emxTable.jsp (in case of tree calling the table page).	Business object ID, for example, 46697.12656.52860.3882
PrinterFriendly	Passes the value specified for the Printer Friendly setting. JSPs that use the PrinterFriendly parameter, such as emxTable.jsp and emxForm.jsp, show the Printer Friendly tool when the setting is True and hide it when False. If the setting is not included, emxTable.jsp and emxForm.jsp show the tool by default. Users can click the tool to get a version of the current page that can be printed with the browser’s Print button.	Value specified in Printer Friendly setting: True False
relId	The relationship ID for the selected relationship. JSPs use the relId to display information for that relationship.	

Parameter	Description	Values Sent
StringResourceFileId	The name of the String Resource file for the application.	emxFrameworkStringResource
suiteKey	Passes the property name from emxSystem.properties that maps to the Registered Suite setting's value. For example, if the Registered Suite is EngineeringCentral, then the suiteKey value is ServiceSuiteEngineeringCentral.	Name of property in emxSystem.properties that maps to Registered Suite setting value.
TipPage	Passes the value specified for the Tip Page setting. If the setting includes a URL, the target page includes the Tip Page tool and the tool calls the URL when a user clicks it. If the setting is not included, the target page does not display the Tip Page tool.	Value specified in Tip Page setting which is the name of a custom URL page.

C

Appendix C: Conversion Routines

Introduction

This section lists automatic and manual data conversions that are run during framework installation. Each section contains an explanation of conversions for each version.

Automatic conversion routines are run for each version that has been released since the installed version. For example, if you are installing version 10.5 and the currently installed version is 10.0.0.0, conversion routines are run for 10.0.0.0, 10.0.1.0, 10.0.2.0 and 10.5.

Manual conversion routines do not need to be run in all circumstances. See the notes for the manual conversion routines to determine if you need to run them.

The conversion routines are divided into these sections:

[Automatic Data Conversions](#)

[Manual Data Conversions](#)

Automatic Data Conversions

This section lists data conversions that are run automatically during framework installation.

AEF 10.0.1.0 Installation

Process Policy and Type (#273087)

This conversion is only used when installing 10.0.1.0 over 10.0.0.2. If you didn't install AEF 10.0.0.2, the conversion doesn't run. The conversion rolls back changes to the Process policy and type that were made in 10.0.0.2. It removes additional states added to the policy so the policy has only the Exists state as before. The conversion also removes the Process type from the *to* end of the Build Specification and Product Specification relationships.

Cage Code and DUNS Number Attribute Data Type Conversion

The Cage Code and DUNS Number attributes are assigned to the Organization type and all its derived types. Previously, these attributes were of type Integer so they could only store integer values. In Europe, Cage Codes and DUNS Numbers often have alphabetical characters. This conversion routine changes the attribute types from integer to string.

Matrix doesn't allow changes to an attribute type once it is created. To make this change, the following steps are performed.

1. Create new attribute named Cage Code New and DUNS Number New of type string and add them to type Organization.
2. Query all the instances of Organization type and its derived types and copy attribute values from Cage Code to Cage Code New and similarly from DUNS Number to DUNS Number New.
3. Remove attribute Cage Code and DUNS Number from type Organization.
4. Rename attribute Cage Code New to Cage Code and DUNS Number New to DUNS Number.'

AEF 10.0.0.0 Installation

The AEF installation will run conversions for the following:

- [*Sourcing Conversion to Change Policy for Supplier ECRs*](#)
- [*Sourcing Conversion for New Vaulting Strategy*](#)
- [*Engineering Central Conversion for Member Relationship and Project Roles*](#)
- [*Route Node ID Conversion*](#)

Depending on how you migrate your applications to V10, it may be necessary to run the first two conversions listed above again. These conversion programs can be downloaded from CustomerLink, in the V10 AEF Download section (<https://customerlink.matrixone.com/custlink/v10.jsp>), and will be made available as part of an upcoming V10 point release.

Sourcing Conversion to Change Policy for Supplier ECRs

This conversion routine lets implementations that have existing supplier-created ECRs take advantage of the MatrixOne Sourcing Central 10.0.0.0 ECR enhancements. When Engineering Central is also installed, these enhancements enable the buyer to connect a supplier-submitted ECR to a product line and submit it for further evaluation. For more information about ECR enhancements, see the Sourcing Central readme and User Guide.

This script runs automatically when framework 10 is installed. However, if Sourcing Central v9.5 continues to run after framework v10 is installed, this script will need to be run again before v10 Sourcing Central can be used so all ECRs are migrated to the correct policies. A script that can be run multiple times and instructions for running this script will be made available on CustomerLink, in the V10 AEF Download section (<https://customerlink.matrixone.com/custlink/v10.jsp>).

The new ECR features require that supplier ECRs have a new policy, called Sourcing ECR, while the ECR is being worked on by the supplier and buyer. After the framework installation program installs schema, this conversion routine runs and finds all ECRs created by suppliers for an RFQ's quotation line item. These are ECRs connected to a Supplier Line Item object with the ECR Reference relationship. The conversion routine acts based on the ECR's state:

If the ECR is in this state:	The conversion routine:
Create The supplier hasn't yet submitted the ECR to the buyer for evaluation.	Change the policy from ECR Standard to Sourcing ECR.
Submit The supplier has submitted the ECR to the buyer.	Change the policy from ECR Standard to Sourcing ECR. Change the ECR's vault and owner to the same as the RFQ.
Any other state The ECR is beyond the Submit state, which means the buyer is no longer working on it. The existing policy, ECR Standard, can remain.	Does not change the ECR and presents a message.

Here are the messages that can result from this conversion routine.

Condition	Warning/Error Message Example
If multiple or no RFQ is connected to ECR	WARNING: Could not run conversion on ECR XXX multiple or no RFQs references found.
If ECR is beyond Submit state	WARNING: Could not run conversion on ECR XXX ECR is beyond state Submit.
Any MQL error.	ERROR: Error: #1900068: modify business object failed System Error: #1: ORA-00942: table or view does not exist

Sourcing Conversion for New Vaulting Strategy

With Sourcing Central 10.0.0.0's new vaulting strategy, objects owned by a company may be distributed across several vaults and several companies may share a vault. To identify which company owns an object, a new attribute called Primary Key stores a unique number for each company. This attribute has been added to the Company type and all Sourcing types, such as RFQs, RFQ Quotations, etc. Additionally, all the employees of a company have a property on their person admin object with the name "Company Key." The value of this property is the same unique company ID stored in the Primary Key attribute for the person's company.

To ensure that buyers see RFQ quotations returned for their RFQs only (and not for another company's RFQs) and that suppliers can only see RFQs sent to them, an attribute called Secondary Keys has been added to RFQ and RFQ Quotation types. For an RFQ, this attribute contains the company ID for all suppliers who have quotations for the RFQ. For an RFQ Quotation, the attribute holds the company ID for the buyer company that owns the RFQ.

Additionally, Holder objects have been created for Package, RFQ, and Currency Conversion objects. A Holder object for each object type is connected to the company and every object of that type is connected to the corresponding Holder object. The name of Currency Conversion objects is prefixed with the company ID.

While setting the unique keys may take some time, this migration can occur without affecting a V9.5x implementation. Therefore, a customer can keep running V9.5x while doing this data migration. However, if the migration is performed and there is continued use of Sourcing Central 9.5x, customers will need to re-run this migration to get all of their Sourcing data to have the correct unique keys. This script will be run automatically when the framework is installed. If Sourcing, Team, or Program Central v9.5 continue to run after v10 schema has been installed, this script will need to be run again before v10 Sourcing, Team, or Program Central can be used. A script that can be run multiple times and instructions for running this script will be made available on CustomerLink, in the V10 AEF Download section (<https://customerlink.matrixone.com/custlink/v10.jsp>).

After the framework installation program installs schema, this conversion routine runs and performs these steps for every company in the database:

1. Assign a unique number, which we'll call the company ID, using the Primary Key attribute for the company object.
2. Find all its employees and add the Company Key property to each employee's person admin object. In the value for the property, enter the company ID.
3. Find all the following objects in the company's vault.
RFQ Template, RFQ, RFQ Quotation, Line Item, Supplier Line Item, ECR, Supplier Line Item Price Period, Line Item Template, Attribute Group, Location, Business Unit, Organization List, Package, Buyer Desk, Currency Conversion
4. For all objects except Currency Conversions, set the Primary Key attribute for each object to the company ID.
5. Create a Holder object for RFQ, Package, and Currency Conversion objects and connect each Holder to the Company object with the RFQ Holder, Package Holder, and Currency Conversion Holder relationship. The revision for each object is the company ID.

6. Connect each RFQ, Package, and Currency Conversion object to the corresponding Holder object with the Company RFQ, Company Package, and Currency Conversion Holder relationships. For each Currency Conversion object, prefix the company ID in front of the name.
7. For each RFQ Quotation, set the Secondary Keys attribute to the company ID for the buyer company that owns the RFQ.
8. For each RFQ that has quotations, set the Secondary Keys attribute to a ~ separated list of company IDs for all suppliers who have a quotation for the RFQ.

Here are the messages that can result from this conversion routine.

Condition	Warning/Error Message Example
When looking for employees of a company (Step 2 above), the Person business object is found but corresponding person admin object not found.	WARNING:> Could not find Person 'P1'.
When attempting to get Primary Key for RFQ Quotation (Step 5), Primary Key not set.	WARNING:> Attribute Primary Key is not set for RFQ Quotation XXXX
Any MQL error.	ERROR: Error: #1900068: modify business object failed System Error: #1: ORA-00942: table or view does not exist

Engineering Central Conversion for Member Relationship and Project Roles

In MatrixOne Engineering Central 10.0.0.0, a person's access to CAD Models, CAD Drawings, Drawing Prints, Parts, ECOs, and ECRs depends on the person:

- being a member of an organization that has responsibility for the object
In terms of schema, this means the Person is connected to a Company with a new relationship called Member and that Company is connected to the object for which access is needed with the Design Responsibility relationship.
- belonging to the role specified in the filter expression for the policy that governs the object
A person's roles for a particular organization are stored in a new attribute called Project Role on the Member relationship.

This conversion routine adds the Member relationship and populates the Project Role attribute for existing persons. After the framework installation program installs schema, the conversion routine runs.

For each Person business object connected to a Company object with the Employee relationship, get all roles assigned to the corresponding person admin object. Connect the Person business object to the Company with the Member relationship. Populate the Project Role attribute on the Member relationship with a list of all the person's roles (the string is formed using the symbolic name of the roles and roles are separated with ~).

Here are the messages that can result from this conversion routine.

Condition	Warning/Error Message Example
Admin person object not found for Person business object connected to Company with Employee relationship.	WARNING:>Could not find Person P1
Person admin object is assigned to an unregistered role.	WARNING:>Role R1 is not registered
Any MQL error.	ERROR: Error: #1900068: modify business object failed System Error: #1: ORA-00942: table or view does not exist

Route Node ID Conversion

This conversion routine accommodates a change in route processing that allows the Route Node relationship ID to change without causing problems in route and task functions.

After the framework installation program installs schema, this conversion routine runs and performs these steps for every route in the database:

1. Get all Route Node relationships and attributes on the relationships.
2. Get all Inbox Tasks and attributes.
3. For each Route Node relationship, get the relationship ID and enter it in the Route Node ID attribute on the relationship.
4. For each Inbox Task:
 - a) Set empty list of matched relationship ids.
 - b) Get attribute values on the Inbox Task.
 - c) For each Route Node connection, get attribute values on the Route Node relationship. If attribute values on Route Node relationship are equal to the attribute values on Inbox Task, append the Route Node relationship ID to the list of matched relationship ids.
 - d) If there are no matched relationship IDs, log a warning message about wrong connection id stored in Inbox Task.

If there is more than 1 matched relationship ids, log a warning message about unresolved situation.

If there is exactly 1, then set Route Node ID attribute to the matched relationship ID.

Here are the messages that can result from this conversion routine.

Condition	Warning/Error Message Example
When Inbox Task Route Node ID does not match with any of the Route Node relationships	WARNING: eServiceConversion10-0-0-0CorrectRouteNodeId.tcl" Inbox Task with id = 10.10.10.10 is storing wrong Route Node Connection Id"
When Inbox Task Route Node ID matches with more than 1 Route Node relationship.	WARNING: eServiceConversion10-0-0-0CorrectRouteNodeId.tcl" Inbox Task with id = 10.10.10.10 is storing wrong Route Node Connection Id" It should store one of the following Route Node connection Ids" 11.11.11.11 12.12.12.1.2 13.12.12.12
Any MQL error.	ERROR: Error: #1900068: modify business object failed System Error: #1: ORA-00942: table or view does not exist

AEF 9.5.2.1 Installation

Change Values for Allow User Edit to Symbolic Name of Role

This conversion routine for Sourcing Central data changes the value for the Allow User Edit attribute to the symbolic name for the role instead of just the role name. For example, if the value is Buyer, the routine changes it to role_Buyer.

Give Company Name Group Accesses to Employee Role

This conversion finds all policies with accesses for the group Company Name. For each state that gives access to the Company Name group, the conversion gives the Employee role the same accesses.

Project WBS Renumbering Conversion

All existing projects that have a WBS structure defined need to have their WBS sequence numbers re-generated in order for PMC 9.5.1.0 + to display a valid value in the WBS Sequence attribute. Previous versions of PMC did not manage the WBS Sequence number but rather a sequence number. In the new release of PMC, WBS numbers are maintained and generated in the User Interface. For this reason, existing customers that use previous releases of PMC must be upgraded to the new WBS sequence logic.

AEF 9.5.1.3 Installation

Project Visibility and Security

This conversion routine for Project Management Central data creates a Project Access List object for each project and its subtasks. This new object is linked to the project using one relationship and to its subtasks using another. The new object also maintains all the grants for the project based on the current project members.

AEF 9.5.1.2 Installation

Correct Circuit Card Part Typo

This conversion routine for Engineering Central corrects a typo in the name and symbolic name for the Circuit Card Part type. The routine renames the part from Circut Card Part to Circuit Card Part and makes the same change to the symbolic name.

If you are using this admin type in your legacy data and have any code that references either the admin type name or its symbolic name, update the code to reflect the modified, corrected name. If you are using this admin type only in sample data and not in legacy data, no corrections are necessary.

AEF 9.5.1.1 Installation

Remove Allow User Edit from Template Attributes

This conversion routine for Sourcing Central data removes the Allow User Edit attribute from the list of template attributes for RFQs and quotations. This attribute is now a system mandatory attribute and therefore cannot be selected for a RFQ template.

AEF 9.5.1.0 Installation

Rename Administrative Objects

This conversion routine renames the Company Representative relationship to Organization Representative. The symbolic name remains relationship_CompanyRepresentative.

It also renames the Workspace Vaults relationship to Data Vaults. The original name of this relationship was Project Vaults so the symbolic name is still relationship_ProjectVaults.

Copy Disposition Attributes

This conversion routine is for Engineering Central and performs the following:

- Copies the disposition and change description attribute information from the Request Part Revision relationship to the New Revision relationship for the *next* revision of the part.
- Copies the disposition and change description attribute information from the Request Part Obsolescence relationship to the Make Obsolete relationship of the *same* revision of the part.
- Copies any attribute data that is common on both of the relationships that the data is being copied from and to.

Messages

This conversion routine changes the way the subject for a message is stored for messages created from Program Central. The routine searches for messages whose Subject attribute is blank and copies the name of the message object to the Subject attribute. It also copies the revision of the message object to the name and makes the revision blank.

Project to Project Space

This Program Central conversion script changes projects created in Program Central to Project Spaces and connects project members directly to the Project Space. This new design supports the new security model.

The routine searches for all business objects whose type is type_Project and that are connected to a company with the relationship Company Project. It changes the type to type_ProjectSpace and the policy to policy_ProjectSpace.

It then expands the Project Members relationship, selecting the attributes Project Access and Project Role found on the Project Member business object. For each Person business ID returned, it connects the Person to the Project Space using the relationship Member. It fills the Member relationship attributes using the attributes Project Access and Project Role. Finally, it deletes the business id of the Project Member business object.

Raised Against ECR

This conversion routine for Engineering Central adds the Raised Against ECR relationship to connect ECRs and the objects they were created for. For each ECR in the database, the routine expands on the “from” side and “Request*” relationships. For each Item related to the ECR, it creates a new relationship Raised Against ECR relationship.

System Mandatory Attribute

This conversion routine for Sourcing Central removes System Mandatory attribute names from the Company Buyer Attributes, Company Supplier Attributes, and Attribute Filter attributes. Related to incident 241661.

AEF 9.5.0.0 Installation

Folders and subfolders were previously called categories and subcategories in the user interface. They are represented by Workspace Vault objects in the database, which were previously called Project Vaults. Workspaces were previously called Projects in both the user interface and in the database. Discussions and their replies are represented by Thread and Message objects.

Convert Route Accesses

This conversion routine is needed to change the way accesses for routes are assigned in the database and only changes routes created from Team Central. Accesses used to be stored on the Member Folder Access connection and are now assigned using grants. For each Route that is connected to a Project Folder, the routine:

1. Gets the route's Project Folder.
2. Gets the Project Folder's accesses from the Member Folder Access connection.

3. Grants access to its members based on their original access to the Project Folder.
4. Connects the routed objects directly to the Route objects.
5. Removes the Project Folder objects.

For information on how Team Central access groups (Read, Read Write, Add, Remove, Add Remove) map to Matrix accesses, see the *Team Central Administrator's Guide*.

Grant Top Level Folder Accesses to Subfolders and Discussions

This Team Central conversion routine grants the accesses assigned for top-level folders to subfolders and discussions for content in folders/subfolders. Previously accesses could only be assigned to top level folders and these accesses applied to subfolders and discussions. Now accesses can be assigned to subfolders and discussions separately.

For each top-level folder, the routine:

1. Gets all its subfolders and discussions for content in the folder and its subfolders.
2. Gets the top-level folder's accesses from the Member Vault Access connection.
3. For discussions, grants Read and Checkout access only. Otherwise, grants the accesses defined by the member's access to the top-level folder.

For information on how Team Central access groups (Read, Read Write, Add, Remove, Add Remove) map to Matrix accesses, see the *Team Central Administrator's Guide*.

Grant Workspace Member Accesses

This conversion routine, which is for Team Central data, grants to each workspace member Read access to the Workspace object.

Grant Team Lead Accesses

This Team Central conversion routine is needed to change the way accesses for Team Leads are assigned in the database.

For each person designated as a Team Lead for each Workspace in the database, the routine:

1. Grants the person Team Lead accesses to the workspace.
2. Grants the person Team Lead access to the folders and subfolders.

For information on how Team Lead access maps to Matrix accesses, see the *Team Central Administrator's Guide*.

Convert Global Read Access

This conversion routine, for Team Central data, changes the way global read access is assigned in the database. For each top-level folder in a workspace, the routine:

1. Checks to see if the Project Security Approach attribute is set to globalread.

2. If so, it goes through all of the top-level folders, subfolders, and content items and grants read, checkout, toconnect, and todisconnect access to the Global User role. All users are assigned to this role.
3. For subfolders and folders, sets the Project Security Approach attribute to globalread.

Revise Line Items and Supplier Line Items with RTS

This conversion routine for Sourcing Central data creates revisions for line items and supplier line items to match the Request to Supplier (RTS) revisions. Previously, when an RTS was revised (a new round was made), the system connected the new revision to the existing line item and supplier line item objects. Now the line item and supplier line item objects are revised along with the RTS and connected to the new RTS.

For every RTS that has a revision greater than 1, the conversion routine revises the connected line item and supplier line items. It copies any associated objects and relationships to the new revisions, connects the new revision to the new RTS revision, and connects the new line item revision to the new supplier line item revision.

Include Required Status for RTS Template Attributes

This conversion routine for Sourcing Central data changes the format of attributes stored in the Attribute Filter and Quotation Attribute Filter attributes for Request to Supplier and RTS Quotation objects. The new format indicates whether the attribute is required. In Sourcing Central version 9.5.0.0, the Buyer Administrator can indicate whether a template attribute is required or not.

Previously, the formats were stored with the format `ATTRIBUTE_NAME~ATTRIBUTE_NAME`. For example, `Purchase Decision~Default Responsibility`. Now the attributes are stored using symbolic names, using `~R` to indicate the attribute is required, and using a semicolon to separate attributes: `SYMBOLIC_NAME~R;SYMBOLIC_NAME~R`. The conversion routine assumes all existing attributes are required and therefore replaces each “~” with “~R;” For example, the attributes listed above would be converted to `attribute_PurchaseDecision~R;attribute_DefaultResponsibility~R;`

Grant Access to Unmanaged Documents

Unmanaged documents are documents attached to objects in Sourcing Central—such as to an RTS, package, quotation, part, or ECR—but that are not in a Team Central workspace folder. This conversion is needed to work with the changes made to the Document policy for the new Team Central access model. The routine grants the following accesses to the Buyer and Supplier roles for all Document objects that are connected to Sourcing Central objects and that are not in Workspace folders.

Show, Read, Checkout, Toconnect, Todisconnect

Make Business Unit Locations also Locations for Company

This conversion routine ensures that all locations for a business unit are also locations for the parent company. For each Company object, the routine:

1. Gets its locations and business units.

2. For each business unit, gets its locations.
3. For each business unit location that is not in the company's location list, connects the Location object to the Company object.

Assign Company Representatives to Role

In previous versions of Team Central, Company Representatives were designated in the schema using the Company Representative relationship between the person and company business objects. This conversion routine finds all person business objects connected with this relationship and assigns the Company Representative role to the corresponding person admin objects.

Populate Originator Attribute Trigger

This data conversion is needed because the TCL trigger program for populating the originator attribute has been replaced with a Java Program Object (JPO) trigger program. The program name has been changed from eServiceSetOriginator_if.tcl to eServiceSetOriginator_if, so the eService Trigger Manager Parameters objects need to call the new program name.

The TCL conversion routine performs these steps:

1. Fires a temp query for all eService Trigger Manager Parameters objects with the value of the eService Program Name attribute set to eServiceSetOriginator_if.tcl.
2. For each returned object, changes the value of the eService Program name attribute from eServiceSetOriginator_if.tcl to emxcommonSetOriginator_if.

Rename Admin Objects

This conversion routine renames the administrative objects listed in the section “Changes to Administrative Object Names” in the *Application Exchange Framework Guide*. The symbolic names do not change so programs that call these objects must use the symbolic name that reflects the original name.

Change Name of Trigger Manager

This conversion routine changes the name of the trigger manager program from eServiceTriggerManager.tcl to emxTriggerManager.

Add Show Access to All Policies

This conversion routine gives Show access to all users who have Read access for every state in every policy.

Manual Data Conversions

This section lists data conversions that are not run automatically during framework installation, but should be run manually only under noted circumstances.

AEF 10.5 Installation

Vaulted Documents Rev2

All instances of the Vaulted Documents Rev2 relationship are converted to Vaulted Documents.

This conversion routine should be run only while upgrading from pre 10-0-0-0 AEF version to 10-5 or post 10-5 version. It should not be run if AEF is already upgraded up to 10-0-0-0 or later.

To run the conversion routine

1. Start MQL.
2. Set context to a super user.
3. Execute the following command:

```
exec program emxCommonCreateObjectIdList.tcl "type_Route"  
"<OBJECTS_PER_TRANSACTION >" "<WORKING_DIR>";
```

This program is used to split all the Routes in the database to smaller chunks.

Each chunk is equal to OBJECTS_PER_TRANSACTION.

It creates RouteList files in WORKING_DIR.

Each file lists Route IDs equal to OBJECTS_PER_TRANSACTION.

The conversion routine reads each RouteList file and applies the conversion on routes listed in the file. Transaction boundaries are applied across Routes listed in each of these files. Example RouteList files will be named as:

RouteList1

RouteList2

RouteList3

.

.

.

RouteListn

4. Execute the following command to run the conversion routine:

```
exec program emxCommonConversion10-0-0-0CorrectRouteNodeId.tcl  
"<WORKING_DIR >" "<LOG_FILE>" "<ROUTE_FILE_BOUNDARY>";
```

where

WORKING_DIR = Directory where the RouteList files are generated.

LOG_FILE = File in which conversion routine will dump messages. Whole path should be specified.

ROUTE_FILE_BOUNDARY = If this input is not set then conversion routine will go through all the RouteList files. This variable gives the option of specifying the range of RouteList files to work. For example: 1,10...11,20...21,21...

5. Repeat steps 3 and 4 to cover all the RouteList files if ROUTE_FILE_BOUNDARY is used.

AEF 10.0.0.0 Installation

Changing the Format to Display Person Names

Framework-based applications can display person names using the Full Name field on the person admin object. In framework v10, this property in emxSystem.properties has been added to define how the Full Name field should be populated:

```
emxFramework.FullName.Format = <Last Name>, <First Name>
```

The system populates Full Name based on this format when a person's name is added, edited, or when the conversion utility described below is run. You can change the value for this property to change how the Full Name field is populated and thereby change how person names are displayed.

Note that the Full Name field and therefore this property only impacts applications that use the Full Name field when displaying person names. Version 10 applications have not fully adopted this display method but it is available for future use.

The property accepts these three sub-keys: <First Name>, <Last Name>, and <User Name>. For example, you could change the format to <First Name> <Last Name> or <User Name>. The default format is <Last Name>, <First Name>. All three sub-keys are filled in when a person is added to an application. First Name and Last Name are attributes for a Person type and the User Name is the name of the Person business and admin object.

Prior to version 10, the framework and applications didn't populate the Full Name field. Therefore, the field is empty for person admin objects added while running a pre-version 10 framework and application.

When a person is added in a v10 or greater application, the Full Name field is populated based on the format defined in the emxFramework.FullName.Format property. Whenever a value that makes up the Full Name format is changed for a person, a trigger program updates the Full Name field to match. For example, if the Full Name format includes <Last Name> and a person's last name is changed, the trigger program updates the Full Name field to match the new last name.

If the Full Name field is not populated for a person admin object and a version 10 and higher application displays names using the Full Name field, the person's username is displayed instead. Framework version 10 includes a conversion utility that populates the Full Name field for all person admin objects based on the name format you specify. You should run this utility if you are using an application that displays names using Full Name and:

- After upgrading to Version 10 framework and applications. The utility will populate the Full Name field for person admin objects that don't have a Full Name and make all populated Full Name fields have a consistent format.
- After changing the Full Name format.
If you decide to change the Full Name format, you can run the utility to convert all Full Name fields to the format you specify.

To run the Full Name conversion utility

1. Make sure you have installed framework 10 or higher.
2. Using the emxFramework.FullName.Format property in emxSystem.properties, make sure you have defined the Full Name format that you want the applications to use when a new person is added or a name is changed.

3. Start MQL and run these commands:

```
set context person [SUPER USER];  
exec program emxFullNameUtility.tcl [LOG FILE] [FULL NAME  
FORMAT]
```

Where the arguments in brackets are specified as described in the table below. An example command for running the utility is:

```
exec program emxFullNameUtility.tcl  
"PATH\emxPersonNameUtility.log" <Last Name>, <First Name>
```

Argument	Input Values
LOG FILE	The path and name of the log file the utility should print messages to. For example: PATH\emxPersonNameUtility.log If this input is not specified, then it defaults to standard console output.
FULL NAME FORMAT	Format to be used to form the full name. This input parameter should be same as the value of emxFramework.FullName.Format property in emxSystem.properties file. The system uses this property to fill in the Full Name for person admin objects when a new person is added or a name is changed. The argument accepts these three sub keys: <First Name>, <Last Name> and <User Name>. For example, the format could be any of the following: <Last Name>, <First Name> <First Name> <Last Name> <User Name> If this input is not specified, it defaults to <Last Name>, <First Name>.

Error and Warning Messages

Here is a description of the errors and warnings that can appear in the log file.

Condition	Warning/Error Message Example
Person admin object not found for a corresponding person business object.	WARNING:Could not find Person P1.
Last Name or First Name attributes are blank and they are referenced in full name format input parameter.	WARNING:Attribute 'Last Name' is not set for Person P1. <i>Or:</i> WARNING:Attribute 'First Name' is not set for Person P1. <i>Or:</i> WARNING:Attribute 'Last Name' and 'First Name' are not set for Person P1.
Any MQL error.	ERROR: Error: #1900068: modify business object failed System Error: #1: ORA-00942: table or view does not exist

Running the EBOM History Conversion Utility

MatrixOne Engineering Central 10.0.0.0 contains a feature that shows historical information about EBOM structures in the EBOM report. To obtain this historical information for existing data, you must run a conversion utility that is installed with framework version 10.0.0.0.

Running the conversion utility is optional. If it is not run, you will not get EBOM history for parts that were created before AEF and EC version 10.0.0.0 were installed. The Start and End Effectivity Dates will be blank for these old EBOM connections.

The conversion utility performs these tasks:

- Set the Start Effectivity Date attribute on the EBOM relationship to the release date of the component part or assembly part, whichever one is later.
- For all previous revisions of component parts, check if the revision was connected to an assembly by an EBOM relationship. If it was, connect the parts with the EBOM History relationship.
- Populate all the attributes on the created EBOM History relationship.
- Populate the Start and End Effectivity Date attributes on the EBOM History relationship to the release date of the assembly or component parts, whichever one is later.

Run this conversion utility after installing framework and Engineering Central 10.0.0.0. If you have a lot of part data to convert, the conversion may take a long time, so you may want to run the conversion in batches. For example, you can convert a portion of the database on one weekend and then another portion the next weekend and so on until done. When you run the utility, you specify which parts to convert, as described below.

To run the conversion utility

1. Make sure you have installed framework and Engineering Central 10.0.0.0.
2. Using Business Modeler or MQL, make sure the MatrixIniDefaults program contains the setting:

```
MX_NORMAL_DATETIME_FORMAT=moy/dom/yr4 h12:min:sec mer
```

3. Start MQL and run these commands:

```
set context to super user;  
exec program emxEBOMHistoryConversionUtility.tcl [PartList]  
[TransactionLimit] [LogFile]  
[SetEndEbomEffectivityForObsoletePart] [StartEffectivityDate]
```

Where the arguments in brackets are specified as described in the table below. An example command for running the utility is:


```
exec program emxEBOMHistoryConversionUtility.tcl "Database" 100
"PATH\emxEBOMHistoryConversionUtility.log" TRUE
```

Argument	Input Values
PartList	This input specifies where to get the list of parts to process. There are three options: - Database—The conversion utility queries the database to get all the released parts. - PATH/QueryList.txt—The utility gets the part list from a text file named QueryList.txt. The file contains the name of query objects to execute to get the part list. You must create these query objects in the database. The format of the text file must be: <pre>Query1 Query2 Query3 . . . Queryn</pre> - PATH/PartList.txt—The utility gets the part list from a simple text file named PartList.txt, which contains Type Name Revisions of all the Released parts that need to be processed. The format of file must be: <pre>Part Part1 1 Hardware Part Part2 2 . . . Electric Part Partn n</pre> Each of these TNRs are separated by . The PartList.txt is useful when there are many parts in the database. Converting all of them may take long so you might want to break large lists of parts into smaller chunks and store them in the text files. Then you can schedule conversions as needed.
TransactionLimit	The number of parts to process in one transaction. If the whole conversion is put in one transaction and the database contains a lot of parts, the database rollback segment may reach its limit, resulting in failure of the conversion program. This argument can be set to one of the following three values: - Number of parts to process in one transaction. 0—No transaction boundary required. 1—Process all the parts in one transaction.
LogFile	The path and name of the log file the utility should print messages to. For example: PATH\emxEBOMHistoryConversionUtility.log

Argument	Input Values
SetEndEbomEffectivityForObsoletePart	This input argument is a flag, which can be set to TRUE or FALSE. If TRUE and a component part is in the Obsolete state, the End Effectivity Date attribute on the EBOM relationship connecting the assembly and the obsolete component is set to the date on which it was made obsolete. If FALSE, the End Effectivity Date is not populated.
StartEffectivityDate	This input should only be used when Release Date information about components and assemblies are not available; for example, when assemblies and components are imported from a non-eMatrix database. If release dates are not available, there is no way to get EBOM History. But by populating Start and End Effectivity Date attributes on the EBOM relationship, EBOM History will be generated from that date forward. Specify the date and time using the following format only. mo/yr4 h12:min:sec mer This format results in a date and time such as 12/31/2000 12:50:50 PM. If Start Effectivity Date input is provided: 1. The Start Effectivity Date attribute for all EBOM connections is populated with the specified date. 2. If an Assembly is connected to a component part that is made obsolete and if SetEndEbomEffectivityForObsoletePart input is TRUE, then End Effectivity Date attribute on EBOM relationship is set to the Start Effectivity input date + 1 day.

Log File

The beginning of the log file lists the input method for getting the part list, the number of transaction boundaries used, and the path and name of the log file. Here is a description of the messages that can appear in the log file.

Condition	Warning/Error Message Example
Component Part not released.	WARNING:Part Component 1 is not in Release state skipping it.
Previous revision of component not released.	WARNING:Part Component 1 is not in Release state skipping it.
If previous revision of component part is already connected to assembly with EBOM History connection.	WARNING: Part Assembly 1 is already connected to Part Component 1 with relationship EBOM History. Attributes on relationship are: Component Location: abc Find Number: 1 Quantity: 4 Reference Designator: abc Usage: abc
Parsing a part list from input text file and the part does not exist.	WARNING:Part Assembly 1 does not exists
While parsing input file, Query specified in the input Query List file does not exist.	WARNING:Query Q1 does not exists
Any MQL error.	ERROR: Error: #1900068: modify business object failed System Error: #1: ORA-00942: table or view does not exist

Index

A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

Symbols

- `${COMMON_DIR}` 490
- `${COMPONENT_DIR}` 490
- `${NAME}` 490
- `${REVISION}` 490
- `${ROOT_DIR}` 490
- `${SUITE_DIR}` 490
- `${TABLE_SELECTED_COUNT}` macro 491
- `${TYPE}` 490
- + sign in tree category 561

Numerics

- 2D Viewable type 173
- 2D/3D view properties 466
- 2D/3D viewer. See viewer.
- 3D Viewable type 173

A

- Abatement Plan attribute 73
- Absence End Date attribute 73
- Absence Of attribute 73
- Absence Start Date attribute 73
- access
 - admin objects used in filter expressions 66
 - to an application 492
 - to cost 111
 - to dynamic UI components 492
 - to form fields 632
 - to form row 492
 - to menu command 492

- to parts 111
- to price 111
- to supplier 111
- to table column 492
- to table columns 493, 578
- to tabs 708
- to toolbar item 492, 715
- to tree categories 549, 688
- to tree category 492
- Access Expression setting 791
- Access Function setting 791
- Access Grantor role 166
- Access Map setting
 - deprecated 791
- Access Map setting
 - deprecated 792
- Access Mask setting 792
- Access parameter 784
- Access Program Method setting
 - deprecated 791, 792
- Access Program setting 792
- Access Type attribute 73
 - used in expression access 67
- accesses for policies 207
- accessing MatrixOne Applications externally 750
- Accounting Clerk role 166
- Accounts Receivable Department group 125
- Action Taken attribute 73
- Action Type setting 792
- Actions menu 128, 515
- Active Version relationship 134
- Actual Benefit attribute 73
- Actual Build Date attribute 73
- Actual Completion Date attribute 73
- Actual Cost attribute 73

- Actual Effort attribute 73
- Actual End Date attribute 73
- Actual Net Benefit attribute 73
- Actual Start Date attribute 74
- Actual Time attribute 74
- Add To Collection command 121
- Additional Object relationship 410
- additional type 410
- address
 - Address attribute 74
 - Address1 attribute 74
 - Address2 attribute 74
 - billing 76
 - shipping 111
- Address attribute 74
- Address1 attribute 74
- Address2 attribute 74
- ADHOC CATEGORY
 - type
 - deprecated 65
- ADHOC CATEGORY type 173
- admin tools 122
- Admin Tools menu 128
- Admin Type setting 793
- Administration Manager
 - using to add your company 372
- Administration Manager role 130, 166, 374
- Administration User person 130
- administrative objects
 - choosing between existing and framework's 385
 - common to all applications 56
 - format for string resource IDs 448
 - getting name of
 - Java 776, 777
 - Tcl 777
 - getting symbolic name of
 - Java 777
 - Tcl 778
 - names of 24
 - prefixed with "eService" 24
 - properties of 381
 - refer to using symbolic name 379
 - renaming 388
 - translating 447
- Advanced Quality Engineer role 166
- AEF Reload Cache Toolbar command 466
- AEFAddToCollection command 121
- AEFAdminTools menu 128
- AEFAppDownloadPackage command 121
- AEFAppMenuToolbar command 121
- AEFAttachmentsTreeCategory command 121
- AEFChangePasswordToolbar command 121
- AEFCollectionDelete command 121
- AEFCollectionItems command 121
- AEFCollectionItemsDelete command 121
- AEFCollections command 121
- AEFCollectionsAddExisting command 121
- AEFCollectionsCreateNew command 121
- AEFCollectionsEditDetails command 121
- AEFCollectionsMenu menu 128
- AEFConversionPreference command 121
- AEFExportTablePreference command 121
- AEFGeneralSearch command 121
- AEFGlobalSearch menu 128
- AEFGlobalToolbar 128
- AEFHelpAboutToolbar command 121
- AEFHistory command 122, 727
- AEFHistoryAllRevisions command 122, 727
- AEFHistoryAllVersions command 122, 727
- AEFHomePagePreference command 122
- AEFHomeToolbar command 122
- AEFIconMailToolbar command 122
- AEFLanguagePreference command 122
- AEFLifecycle command 122
- AEFLifecycleDemote command 122
- AEFLifecycleMenu 128
- AEFLifecyclePromote command 122
- AEFLogoutToolbar 462
- AEFLogoutToolbar command 122
- AEFMonitor command 122
- AEFMQL command 122
- AEFPageHistoryToolbar command 123
- AEFPaginationPreference command 123
- AEFPowerView menu 120
- AEFPowerViewActionsToolbar menu 128
- AEFPowerViewChannel menu 120
- AEFPowerViewHome command 123
- AEFPowerViewToolbar menu 128
- AEFPreferenceToolbar command 123
- AEFReloadCacheToolbar command 123
- AEFReviseSearch command 123
- AEFSaveAsSearch command 123
- AEFSavedSearches command 123
- AEFSaveSearch command 123
- AEFSearchActionsToolbar menu 128
- AEFSearchResultToolbar menu 128
- AEFSearchToolbar menu 128
- AEFSearchTypes menu 128
- AEFSeparator command 123
- AEFSystemData command 123
- AEFTimeZonePreference command 123
- Affected Code relationship 134
 - deprecated 65

- Agreed Unit Price attribute 74
- Allocation Responsibility relationship 134
- Allow Delegation attribute 74
- Allow Duplicates attribute 74
- Allow Manual Edit setting 793
- Allow User Edit attribute 74
- Also see arguments.
- Alt parameter 784
- alternate OID
 - form field 655
- Alternate OID expression setting 793
- Alternate Policy expression setting 793
- Alternate relationship 134
- alternate tree 559
- Alternate Type expression setting 793
- alternate type icon
 - form field 655
- Annual Part Quantity attribute 74
- API
 - for navigation tree 538
- App Download Package command 121
- App Menu command 121
- Appearance attribute 74
- AppendParameters parameter 557, 818
- applet
 - file limit 470
- application
 - Also see MatrixOne Applications and suite.
 - data models for 29
- Application Exchange Framework
 - associations 72
 - attributes 73
 - commands 121
 - components of 14
 - configuring 379
 - formats 124
 - groups 125
 - how related to applications 14
 - menus 128
 - naming 24
 - overview of 14
 - persons 130
 - programs 132
 - relationships 132
 - roles 166
 - stores 172
 - types 173
 - vaults 204
- Application Exchange Framework Guide 20
- Application Part relationship 134
- Application part type 173
- application properties 482
- application property 381
- ApplicationStartPage property 480
- Applied Against relationship 134
 - deprecated 65
- Applied Markup relationship 134
- Applies To parameter 784
- Approval Status attribute 74
- approvals for promotion and tasks 440
- Approved Date attribute 74
- Approver attribute 74
- Approvers Responsibility attribute 75
- Architecture type 173
- arguments
 - for
 - eServicecommonWzrdListFromCommand.tcl 404
 - for Trigger Manager 417
 - modifying for triggers 425
 - referring to admin objects in 379
- Arguments parameter 609, 784
- Assembly Technical Specifications type 173
- Assessment Comments attribute 75
- Assessment Status attribute 75
- Assessment type 173
- Assigned Build Part relationship 134
- Assigned Buyer relationship
 - described 134
 - used in expression access 67
- Assigned Documents relationship 134
- Assigned Feature relationship 135
 - used in expression access 67
- Assigned Incident relationship 135
 - used in expression access 67
- Assigned Issue relationship 135
 - used in expression access 67
- Assigned Meetings relationship 135
- Assigned Member relationship 135
 - used in expression access 68
- Assigned Requirement relationship 135
 - used in expression access 68
- Assigned Risks relationship 135
- Assigned Sales Engineer relationship 135
- Assigned Tasks relationship 135
 - used in expression access 68
- Assigned Test Case relationship 135
 - used in expression access 68
- Assigned To Buyers Desk relationship 136
- Assigned To Specification Office relationship 136
 - used in expression access 68
- Assigned Use Case relationship 136
- Assignee Role attribute 75
- Assignee Set Due Date attribute 75

- Associated Drawing relationship 136
- associations
 - descriptions of 72
- Attachments Tree Category command 121
- Attribute Filter attribute 75
- Attribute Group
 - relationship 136
 - type 173
- Attribute Group Supplier relationship 136
- Attribute Group Template relationship 136
- Attribute Group Type attribute 75
- attribute ranges
 - formats for string resource IDs 448
 - specified by program 403
 - specified manually 405
- attributes
 - customizing range values 403
 - descriptions of 73
 - for relationships 132
 - for types 173
 - range values specified by program 403
 - range values specified manually 405
- authentication 462
- Author role 166
- Authorized Temporary Specification attribute 75
- Authorized Temporary Specification
 - relationship 136
- auto filter 581, 794
- Auto Filter setting 794
- automatic data conversions 838
- automatic name
 - configuring 413
 - data model 409
 - how it works 412
 - JSP code 415
 - overview 409
 - program 411
 - Tcl code 415
- automation 765
- autoName function 415
- Available On relationship 136
 - renamed 62
- Average attribute 75
- average of column values 604
- Award Status attribute 75
- Awarded Line Item relationship 136

B

- background for login page 461
- banner 497
- banner properties 480

- Base Price attribute 75
- Baseline Current End Date attribute 76
- Baseline Current Start Date attribute 76
- Baseline Initial End Date attribute 76
- Baseline Initial Start Date attribute 76
- Baseline Log relationship 136
- Baseline Log type 174
- Baseline Text attribute 76
- Baseline type 174
- Benefit Interval attribute 76
- Benefit Interval End Date attribute 76
- Benefit Interval Start Date attribute 76
- Benefit Item Interval relationship 136
- Benefit Item type 174
- Billing Address attribute 76
- Bolt Part type
 - described 174
 - in Part inheritance model 29, 31
- Book type 174
- Bookshelf type 174
- Boolean Compatibility Rule
 - relationship 136
 - type 174
- BOS relationship 136
- Bounding Box attribute 76
- Bracket Part type
 - described 174
 - in Part inheritance model 29, 31
- Branch To attribute 76
- Brava viewer. See viewer.
- browser language setting 450
- Budgetary Estimate type 174
- Bug Impact Analysis relationship 137
 - deprecated 65
- Bug type 174
- Build Management group 125
- Build Satisfied relationship 137
- Build Satisfies relationship 137
- Build Serial Number attribute 76
- Build Source Code relationship 137
- Build Specification relationship 137
- Build Specification type 174
- Builds type 174
- Business Benefit attribute 76
- Business Goal Project Space relationship 137
- Business Goal type 175
- Business Manager role 166
- Business Modeler
 - adding person 373
 - changing Company Name 372
- business objects
 - automatic naming 409, 413

- Company Name 120, 176
- eService Number Generator 120
- eService Object Generator 120
- eService Trigger Program Parameters 120, 417
- for your company 371
- installed with framework 15, 120
- Test Everything person 120
- Business Unit Employee relationship 137
 - used in expression access 68
- Business Unit Owns relationship 137
 - used in expression access 68
- Business Unit Project relationship 137
- Business Unit Role relationship 137
- Business Unit type 175
- business unit. *See* organization.
- BusinessObject.getSelectedBusinessObjectData
 - method 605
- Buyer Administrator role 166
- Buyer Desk
 - type 175
- Buyer Desk Bid List relationship
 - described 138
 - used in expression access 68
- Buyer For relationship 138
- Buyer role 166

C

- cache
 - getting admin name from 776
 - getting symbolic name from 777
- CAD Drawing
 - type
 - described 175
 - in Engineering Central data model 30
- CAD Model
 - type
 - described 175
 - in Engineering Central data model 30
- CAD SubComponent relationship 138
- Cage Code attribute 76
- calculation settings 604
- CancelButton parameter 611, 818
- CancelLabel parameter 611, 819
- Candidate Incident relationship 138
 - deprecated 65
- Candidate Item relationship 138
- Candidate Request relationship
 - deprecated 65
- Capability relationship 138
- Capability Status attribute 76

- Capacitor Part type
 - described 175
 - in Part inheritance model 29, 31
- Capacity attribute 76
- Capital Expenditure Part type
 - described 175
 - in Part inheritance model 29, 31
- cardinality 27
- CAS Number attribute 77
- cascading style sheets 497
- categories. *See* navigation trees.
- categories. *See* navigation trees.
- Category of Change attribute 77
- Certification
 - relationship 138
 - type 175
- Change Administration Property wizard 389
- Change attributes
 - general description of 87
 - reason for 102
 - specific description of 112
- Change Board attribute 77, 403
- Change Boards group 125
- Change Management type 175
- Change Password command 121, 741
- Change State Property wizard 396
- changing onscreen text 406
- changing password 460
- channel
 - controlling the layout 701
 - controlling the layout of tabs 702
- Channel parameter 785
- channel tabs
 - portal page 698
- character, special 470
- Characteristic relationship 138
- Characteristic Specification relationship 138
- Characteristic type 175
- check boxes for table columns 594
- Checkin Reason attribute 77
- checkin/checkout
 - file limit 470
 - Java plug-in 470
 - timeout 470
 - use applet 470
- Chemical Characteristic type 175
- Child Markup relationship 138
- child type. *See* type inheritance.
- Circuit Card Part type
 - described 176
 - in Part inheritance model 29, 31
- Cirut Card Part type

- renamed 63
- City attribute 77
- Claimed Attribute List attribute 77
- Claimed SCO relationship 138
- Classification attribute
 - renamed 61
- clone rules, listed 134
- Closed By attribute 77
- Closed Date attribute 77
- Code parameter 609, 785
- Co-Dependent relationship 138
- Collaboration Partner relationship 139
- Collaboration Request relationship 139
- Collaboration Request Type attribute 77
- Collection Delete command 121
- Collection Items command 121
- Collection Items Delete command 121
- Collections Add Existing command 121
- Collections command 121
- Collections Create New command 121
- Collections Edit Details command 121
- Collections menu 128
- Color attribute 77
- colors 497
- Cols setting 794
- column
 - calculation settings 604
 - calculations in editable table 602
 - count of items 602
 - defining in editable table 597
- Column Icon setting 794
- Column Type setting 795
- column. *See* table and table page.
- combo box 657
- CommandName parameter 508, 819
- commands
 - descriptions of 121
 - internationalizing 488
- Comment attributes
 - dated and author identified 77
 - general, entered by user 77
 - reviewer 106
 - state 113
- Comments attribute 77
- Commitment Level attribute 78
- Committed Item relationship 139
- Common Access Grantor person 130
- common directory 490
- common schema. *See* Application Exchange Framework.
- COMMON_DIR macro 490
- CommonDirectory 479

- company
 - adding employees 373
 - adding your own 371
 - Also *see* organization.
 - business object 371
 - file store 426
 - notifications specific for 407
- Company Attributes attribute 78
- Company Buyer Attributes attribute 78
- Company Contact role 166
- Company Currency Conversion relationship 139
- Company Department relationship 139
- Company Document relationship
 - used in expression access 69
- Company Documents relationship 139
- Company ECR relationship 139
- Company Financial Categories relationship 139
- Company Name
 - business object 120, 176, 371
 - group 125, 372
- Company Package relationship 139
- Company Product Line relationship 139
- Company Product relationship 139
 - used in expression access 69
- Company Program relationship 139
- Company Project relationship 139
- Company Project Templates relationship 139
- Company Report relationship 139
- Company Representative
 - relationship
 - described 152
 - renamed 62
 - role 166
- Company RFQ relationship 139
- Company Role relationship 140
- Company Route Templates relationship 140
- Company Specifications Templates
 - relationship 140
- Company Supplier Attributes attribute 78
- Company Teams relationship 140
- Company type 176
- company. *See* organization.
- Comparison Operator attribute 78
- Compatibility Rule. *See* Boolean Compatibility Rule or Option Compatibility Rule.
- Competitor relationship 140
- Component Engineer role 166
- Component Location attribute 78
- Component Substitution relationship 140
- COMPONENT_DIR macro 490
- ComponentDirectory 479
- Conference Call Access Code attribute 78

- Conference Call Number attribute 78
- Configurable Feature
 - type 176
- Configurable Feature type
 - renamed 63
- configurable form. *See* form.
- Configurable Product
 - type 176
- Configurable Product policy
 - deprecated 64
- Configuration File
 - relationship 140
 - type 176
- configuring
 - login page 459
 - onscreen text 406
 - overview 379
 - system properties 464
- Confirm Message setting 491, 795
- Connector Part type
 - described 176
 - in Part inheritance model 29, 31
- Constraint Date attribute 78
- Constraints attribute 78
- Contact Us page 461
- Container
 - policy
 - deprecated 64
- Container Cost attribute 78
- Contains
 - relationship
 - deprecated 65
- Contains relationship 140
- ContentPage parameter 507, 819
- ContentPage property 480
- Context Assembly relationship 140
- Context Organization relationship 140
- Contextual Error attribute 78
- Controlled Design Release
 - policy
 - deprecated 64
- Controlled Production Release
 - policy
 - deprecated 64
- controlling access in dynamic user interface 492
- Conversion Preference command 121
- conversion preferences 717, 720
- conversion routines
 - automatic 838
 - manual 849
- ConvertCurrencyTag 719
- CoOwned List type 176
- CoOwned relationship 140
 - used in expression access 69
- Co-Owner attribute
 - renamed 61
- Co-Owners attribute 78
- CoOwns relationship 141
 - used in expression access 69
- Corporate person 130
- cost
 - access to 111
 - assessment 79
 - container 78
 - estimated 85
 - lifecycle 91
 - relative 103
 - target 114
 - tooling 115
 - total 116
- Cost Assessment attribute 79
- Cost Interval attribute 79
- Cost Interval End Date attribute 79
- Cost Interval Start Date attribute 79
- Cost Item Interval relationship 141
- Cost Item type 177
- Count attribute 79
- count for tree category 548
- Country attribute 79
- Create Folder attribute 79
- Create Route attribute 79
- Create Vault attribute 79
- CreateAdditional parameter 412
- Created On attribute 79
- css files 497
- Currency attribute 79
- currency conversion 717, 720
- currency conversion data model 57
- Currency Conversion Holder relationship 141
- Currency Conversion type 177
- Currency Converter setting 796
- Currency Expression setting 796
- currency fields 471
- CurrencyConverter parameter 834
- Current Route Node attribute 79
- Current Version attribute 80
- custom applications
 - naming 511
- custom filter 621
- Custom Item relationship 141
- custom navigation tree 559
- custom sorting 624
- Customer
 - relationship 141

- role 166
- Customer Design relationship 141
- Customer Representative role 166
- Customer Requirement Specification type 177
- Customer Specification type 177

D

- data conversions
 - automatic 838
 - manual 849
- Data Document
 - type 177
- data model
 - automatic naming 409
 - common to all applications 56
 - currency conversion 57
 - effort 56
 - Engineering Change application 30
 - how to read 27
 - issues 55
 - MatrixOne Document Central 29
 - MatrixOne Engineering Central 30
 - MatrixOne Product Central 34
 - MatrixOne Program Central 38
 - MatrixOne Sourcing Central 41
 - MatrixOne Specification Central 46
 - MatrixOne Supplier Central 47
 - MatrixOne Team Central 50
 - messages 57
 - Profile Manager application 48, 49
 - publish subscribe 58
 - routes 53
 - subscriptions 58
 - viewers 57
- Data Vaults relationship 141
 - used in expression access 69
- database 379
- date
 - absence end 73
 - absence start 73
 - actual completion 73
 - actual end 73
 - actual start 74
 - constraint 78
 - created 79
 - current baseline end 76
 - current baseline start 76
 - deleted 80
 - effective 82
 - effectivity 82
 - end effectivity 83
 - escalation 83
 - estimated end 85
 - estimated start 85
 - initial baseline end 76
 - initial baseline start 76
 - internationalizing 446
 - last login 90
 - meeting start 94
 - of last legacy update 90
 - production end 99
 - production start 99
 - promised delivery 101
 - quote requested by 102
 - release 103
 - requested delivery 104
 - required availability 104
 - requirement 80
 - scheduled completion 109
 - start effectivity 112, 113
 - submitted 113
 - task actual finish 114
 - task actual start 114
 - task estimated finish 114
 - task estimated start 114
 - version 118
- Date Deleted attribute 80
- Date Format setting 797
- Date Offset From attribute 80
- Date Requirement attribute 80
- dates
 - in form field 660
- DateTime.DisplayFormat 474
- DateTime.DisplayTime 474
- Day Delta attribute 80
- debugging internationalization 446
- decimal symbol 471
- Default Category setting 796
- Default Line Item Attribute Group relationship 141
- Default Line Item UDA Miscellaneous relationship 141
- Default Line Item UDA Non-recurring Cost relationship 142
- Default Line Item UDA Recurring Cost relationship 142
- Default Policy attribute 80
- Default Region relationship 142
- Default Responses attribute 80
- Default Responsibility attribute 80
- default search vaults 377
- Default Selection attribute 80
- Default setting 796
- Default Specification Office relationship 142

- default tree 559
- Default Tree menu 128
- DefaultCategory parameter 819
- DefaultTree.TreeName property 481
- defaultType 560
- defaultType property 481
- Defect Definition attribute 80
- Defect type 177
- Defects Per Million Opportunities attribute 80
- Defects Per Unit attribute 80
- Deleted Subtasks relationship 142
- Deliverable
 - type 177
- Deliverables relationship 142
 - deprecated 65
- delivery date 104
- Density attribute 80
- Department Employee relationship 142
- Department type 177
- Dependency relationship 142
- Dependency Type attribute 80
- deprecated methods
 - for navigation trees 542
- deprecated schema 64
- Derived Component relationship 142
- derived from. *See* type inheritance.
- Derived Viewable relationship 142
- Description attribute 80
- Design Engineer role 167
- Design Engineering group 125
- Design Group Owner attribute 80
- Design Process type 177
- Design Responsibility relationship
 - described 142
 - used in expression access 69
- Design Specification type 177
- Designated User attribute 81, 404
- details tree. *See* navigation trees.
- Development Manager relationship 142
- Diameter attribute 81
- Difficulty attribute 81
- DigitSeparator 471
- Directed By relationship 142
- Directed Item relationship 143
- Directed Supplier
 - type 177
- Directed To relationship 143
- directory
 - for suite 482
 - properties 479
- directory macros 490
- disableAllExports 468
- Display Time setting 797
- displayAdminName property 450
- Disposition attributes
 - Field Return 81
 - In Field 81
 - In Process 81
 - In Stock 81
 - On Order 82
- Disposition Comments attribute 82
- Distribution Available On relationship 143
- Distribution File
 - type 178
- Distribution File relationship 143
- Distribution Groups group 125
- Distribution List relationship 143
- Distribution Media relationship 143
- Division
 - attribute 82
 - relationship 143
- Document
 - type
 - described 178
 - in Team Central data model 50
- Document Classification attribute 82
- DOCUMENT CLASSIFICATION type 178
- Document Control Manager role 167
- Document Control role 167
- Document Holder relationship 143
 - used in expression access 69
- Document Management Administrator role 167
- Document Management type 178
- Document Part type
 - described 178
 - in Part inheritance model 29, 31
- Document Sheet type 178
- Document Sheets relationship 143
- Document Structure relationship 143
- Document Version relationship 143
- documentation
 - for applications 18
 - installing internationalized 454
- Documentation group 126
- DOCUMENTS type 178
- Drawing Print
 - type
 - described 178
 - in Engineering Central data model 30
- Due Date Offset attribute 82
- DUNS Number attribute 82
- Duplicate
 - relationship 143
 - type

- described 178
- Duplicate relationship
 - deprecated 65
- dynamic user interface
 - accessing externally 750
 - banner image 497
 - building menus 515
 - components of 485
 - controlling access 492
 - expressions 488
 - macros 488
 - menus 513
 - naming conventions 486
 - navigation trees 526
 - overview 485
 - standard commands 740
 - style sheets 497
 - toolbar 513
 - types of menus 513

E

- EBOM expansion 597
- EBOM History relationship 144
- EBOM ID attribute 82
- EBOM level information 597
- EBOM Markup
 - relationship 144
 - type
 - described 178
 - in Engineering Central data model 30
- EBOM relationship 143
- ECAD Model type 179
- ECO Change Request Input relationship 144
- ECO type
 - described 179
 - in Engineering Central data model 30
- ECR Chairman role 167
- ECR Coordinator role 167
- ECR Evaluator
 - attribute 82, 404
 - role 167
- ECR Holder relationship 144
- ECR Main Product Affected relationship 144
- ECR Originator Company relationship 144
- ECR Originator Organization attribute 82
 - deprecated 64
- ECR Reference relationship 144
- ECR Supporting Document
 - relationship 144
- ECR type
 - described 179

- in Engineering Central data model 30
- Edit link on form page 647
- Edit mode for form 627
- Editable setting 797
- editable tables 575
- editDataForm form 629
- editFooter form 629
- editLink parameter 611, 644, 819
- Effective Date attribute 82
- Effective Date Expression setting 797
- Effectivity Date attribute 82
- Effort
 - data model 56
 - Resolve Effort attribute 105
- Effort Classification attribute 82
- Effort type 179
- EINECS 82
- EINECS Number attribute 82
- Electrical Part type
 - described 179
 - in Part inheritance model 29, 31
- email
 - attribute 82
- Email Address attribute 82
- email notification properties 471
- email, accessing from 750
- ematrix.common.disableAllExports 468
- ematrix.common.icon.spreadsheet 468
- ematrix.sso.logout.url 463
- eMatrixDateFormat 475, 477
- eMatrixInputDateFormat 475, 477
- Employee
 - adding 373
 - relationship 144
 - role 168
- Employee relationship
 - used in expression access 69
- Employee Representative relationship 144
- emxCommonAppInitialization.jsp 473
- emxCommonBaseComparator program 624
- emxcommonSessionInfo program 439
- emxcommonSetOriginator_if trigger program 764
- emxComponentsRoutes.InboxTask.ExternalAuthentication 440
- emxComponentsRoutes.InboxTask.ExternalAuthenticationURL 440
- emxCreatePersonBusObject 473
- emxForm.jsp
 - example URLs 631
 - URL parameters 644
- emxFrameowrk.LifeCycle.ApprovalPasswordConfirmation 440

- emxFramework.Applet.MaximumNoFilesLimit 470
- emxFramework.DateTime.DisplayFormat 474
- emxFramework.DateTime.DisplayTime 474
- emxFramework.DecimalSymbol 471
- emxFramework.DefaultSearchVaults 377, 465
- emxFramework.DigitSeparator 471
- emxFramework.EnableURLNotification 471
- emxFramework.FilterParameterPattern 468
- emxFramework.FolderContentTypesThatTypesRequireGrants property 469
- emxFramework.FormatsWithViewerPreference 466
- emxFramework.FullName.Format 468
- emxFramework.GlobalSearch.Types 465
- emxFramework.HelpDirectory 454
- emxFramework.History.SpecialActionType 469
- emxFramework.IconMail.FindSelection 472
- emxFramework.IconMail.ShowSelects 472
- emxFramework.IncludeSubTypesForGrants property 469
- emxFramework.Javascript.BadChars 471, 817
- emxFramework.Javascript.NameBadChars 471, 817
- emxFramework.Javascript.RestrictedBadChars 817
- emxFramework.LifeCycle.ApprovalPasswordConfirmation 467
- emxFramework.LifeCycle.ExternalAuthentication 440
- emxFramework.LifeCycle.ExternalAuthenticationURL 440
- emxFramework.PaginationRange 481
- emxFramework.Preferences.ExportFormat.Choices 725
- emxFramework.Preferences.ExportFormat.Default 725
- emxFramework.Preferences.FieldSeparator.Choices 725
- emxFramework.Preferences.FieldSeparator.Default 725
- emxFramework.Preferences.FieldValueSeparator.Delimiter 726
- emxFramework.Preferences.Language.Choices 724
- emxFramework.Preferences.Language.Default 724
- emxFramework.Preferences.RemoveCarriageReturns.Choices 725
- emxFramework.Preferences.RemoveCarriageReturns.Default 725
- emxFramework.Routes.EnableFDA 440
- emxFramework.Search.DefaultVaultPreference 464
- emxFramework.Search.OnSelect.Action 466
- emxFramework.Search.UpperQueryLimit 464
- emxFramework.ServerTimeoutInSec 470
- emxFramework.ShowLanguageForFileCheckin 468
- emxFramework.ShowMassUpdate 482
- emxFramework.smallIcon 481, 560
- emxFramework.smallIcon.defaultType 481, 560
- emxFramework.Timing.LogFile 482
- emxFramework.UIForm.ValidationFile 669
- emxFramework.UIStructureTree.ShowUponOpen 480
- emxFramework.UITree.PreV10AppsSupport 481
- emxFramework.Unix.JavaPlugInURL 470
- emxFramework.UseApplet 470
- emxFramework.Viewer.ServletPreFix 466
- emxFramework.ViewerExtensions 431, 466
- emxFrameworkStringResource.properties 406
- emxGetPropertyFromAdminName 778
- emxGetPropertyFromStateName 778
- emxHistory.jsp 727
- emxI18NMethods.jsp 450
- emxLifecycleSSOLogout.fcc 440
- emxLogin.jsp 459, 462
- emxLogin.properties 460
- emxLogout.jsp 462
- emxMQLNotice.jsp 776
- emxNavigator.DefaultTree.TreeName 481, 559
- emxNavigator.Home.ContentPage 480
- emxNavigator.Home.PLMLink 480
- emxNavigator.jsp 459
 - accessing externally 750
 - parameters passed to 507
- emxNavigator.UICache.ComponentAge 466
- emxNavigator.UIHeader.LeftImage 480
- emxNavigator.UIHeader.RightImage 480
- emxNavigator.UITable.HeaderRepeat 481
- emxNavigator.UITable.Pagination.RememberSelection 482
- emxNavigator.UITable.Style.Dialog 481, 615, 831
- emxNavigator.UITable.Style.List 481, 615, 831
- emxPortal.jsp 699
 - URL parameters 709, 710
- emxSafetyVault 473
- emxSortNumericAlphaLarger program 626
- emxSortNumericAlphaSmaller program 626
- emxSuiteDirectory parameter 820, 834
- emxSystem.pageHistory.limit 467
- emxSystem.properties
 - configuring 2D/3D view properties 466

- configuring application properties 482
- configuring applications 464
- configuring banner properties 480
- configuring date/time format properties 474
- configuring directory properties 479
- configuring email notification properties 471
- configuring Engineering Central properties 473
- configuring form properties 482
- configuring IconMail properties 472
- configuring initialization properties 473
- configuring miscellaneous properties 466
- configuring resource file properties 479
- configuring search properties 464
- configuring special character properties 470
- configuring start/home page properties 480
- configuring table properties 481
- configuring tree properties 480
- file management property
 - for file management 470
- emxTable.jsp
 - creates table page 571
 - parameters for 611
- emxTableForm 566
- emxTableRowId 566, 589
- emxTaskSSOLogout.fcc 441
- emxTree.jsp
 - hooks navigation trees to apps 535
 - parameters for 556
- emxTreeAlternateMenuName 560
- emxTriggerManager program 133, 417
- emxTypeChooser.jsp 732
- emxUIChooser.css 497
- emxUIDialog.css 498
- emxUIForm.css 499
- emxUILifeCycle.css 499
- emxUIList.css 500
- emxUINavigator.css 501
- emxUIProperties.css 501, 502
- emxUISearch.css 503
- emxUIToolbar.css 503
- emxUITree.css 504
- emxUIWizard.css 505
- emxUIWorkflow.css 506
- EnableURLNotification 471
- End Effectivity attribute 83
- End Effectivity Date attribute 83
- End State attribute 83
- Engineering Central properties 473
- Engineering Change application 30
- Entered Name attribute 83
- Erratum type 179
- Error Category attribute 83
- error handling for JSPs 747
- Error Message attribute 83
- Error Navigation attribute 83
- Error Report attribute 83
- Escalation Date attribute 83
- Escalation Recipients attribute 83
- Escalation Required attribute 83
- eService Additional Object relationship
 - attributes 411
 - described 144
- eService Administration vault 204
- eService Change Administration Property wizard 205
- eService Change State Property wizard 205
- eService Connect Relation attribute 83
- eService Constructor Arguments attribute 83
- eService Language wizard 205
- eService Method Name attribute 83, 424
- eService Name Prefix attribute 84
- eService Name Suffix attribute 84
- eService Next Number attribute 84
- eService Number Generator
 - relationship 144
 - type
 - attributes 411
 - described 409
 - summary 179
- eService Object Generator
 - type
 - attributes 411
 - described 409
 - summary 179
- eService Processing Time Limit attribute 84
- eService Production vault 204
- eService Program Argument attribute 84, 423
- eService Program Argument Desc attribute 84, 424
- eService Program Name attribute 84, 423
- eService Register Application wizard 205
- eService Register Feature wizard 205
- eService Register Page Access wizard 205
- eService Register Suite wizard 205
- eService Register Viewer wizard 205
- eService Retry Count attribute 84
- eService Retry Delay attribute 84
- eService Safety Policy attribute 84
- eService Safety Vault attribute 84
- eService Sample vault 120, 204
- eService Sequence Number attribute 85, 423
- eService Target States attribute 85, 424
- eService Trigger Program Parameters
 - creating object 422

- for Set Originator attribute program 764
- how used 417
- installed with framework 764
- processed by Trigger Manager 420
- type 180
- eServiceCheckPageAccess.tcl program 132
- eServicecommonCheckRelState_if.tcl trigger program 761
- eServicecommonGetViewers.tcl program 132
- eServicecommonObjectGenerator.tcl program 411
- eServicecommonPreviousRevisionPromotion_if.tcl trigger program 758
- eServicecommonRelativeFloatAction_if.tcl trigger program 763
- eServicecommonRequiredConnection.tcl trigger program 758
- eServicecommonRevisionGenerator.tcl program 132
- eServicecommonShadowAgent.tcl 773
- eServicecommonTrigcRequiredFormat_if.tcl trigger program 760
- eServicecommonWzrdListFromCommand.tcl program 132, 404
- eServiceCreateList.tcl program 132
- eServiceDocumentCentral.General.FDACompliant Signatures 440
- eServiceEngineeringCentral.emxCreatePersonBus Object 473
- eServiceEngineeringCentral.emxSafetyVault 473
- eServiceFramework.FormatsWithViewerPreference 432
- eServiceGetCurrentSchemaName 777
- eServiceGetUserApps.tcl program 132
- eServiceGetUserFeatures.tcl program 132
- eServiceGetUserSuites.tcl program 132
- eServiceListSchemaNames.tcl program 132
- eServiceNumberGenerator procedure 415
- eServiceRegistry.tcl program 132
- eServiceSchemaVariableMapping.tcl
 - described 132
 - getting admin object name 777
 - getting symbolic names 778
 - reassigning symbolic names 389
 - stores symbolic names 379
- eServiceSUITE_NAME.ApplicationPropertyFile 482
- eServiceSUITE_NAME.ApplicationStartPage 483
- eServiceSuiteFramework.ApplicationStartPage 480
- eServiceSuiteFramework.CommonDirectory 479
- eServiceSuiteFramework.ComponentDirectory 479

- eServiceSuiteFramework.Directory 479
- eServiceSuiteFramework.InitializationFile 473
- eServiceSuiteFramework.RootDirectory 479
- eServiceSuiteFramework.StringResourceId 479
- eServiceSuiteFramework.UIForm.ValidationFile 669
- eServiceSuiteFramework.URL 480
- eServiceSuites.DateFormat 475, 477
- eServiceSuiteSUITE_NAME.Directory 482
- eServiceSystemInformation.tcl 133
- eServiceUtil.jsp 415
- eServiceValidRevisionChange_if.tcl trigger program 763
- Estimate Schedule Impact attribute 85
- Estimated Benefit attribute 85
- Estimated Completion Date attribute 85
- Estimated Cost attribute 85
- Estimated Effort attribute 85
- Estimated End Date attribute 85
- Estimated Max attribute 85
- Estimated Min attribute 85
- Estimated Net Benefit attribute 85
- Estimated Schedule Impact attribute 85
- Estimated Start Date attribute 85
- Event
 - type
 - described 180
 - in Team Central data model 51
- Event Post Processing menu 128
- Event Type attribute 86
- Exchange User role 168
- ExclusionList parameter 820
- execution sequence
 - attribute 85
 - modifying 425
- existing objects 385
- Expand Function setting 561, 797
- Expand Inquiry setting 550, 561, 798
- Expand Program setting 561, 799
- Expiration Date attribute 86
- expired password 460
- export format 724
- Export parameter 611, 820
- Export setting 799
- ExportFormat property 725
- ExportTablePreference command 121
- expression accesses 66
- Expression attribute 86
- Expression parameter 785
- expressions used in dynamic UI 488
- Extend Unit Price Formula attribute 86

- extension for files 87
- external access 750
- external authentication 462
 - and signature approvals 440
- External Program Lead role 168
- External Project Administrator role 168
- External Project Lead role 168
- External Project User role 168
- Extruded Part type
 - described 180
 - in Part inheritance model 29, 31

F

- Family Member relationship 144
- Fastener Part type
 - described 180
 - in Part inheritance model 29, 31
- Fax Number attribute 86, 95
- FDA signature approvals 440
- FDACompliantSignatures 440
- Feature Breakdown relationship 144
- Feature Category attribute 86
- Feature Classification attribute 86
- Feature Design Specification relationship 145
- Feature Functional Specification relationship 145
- Feature List From relationship 145
 - used in expression access 69
- Feature List relationship 145
- Feature List To relationship 145
 - used in expression access 69
- Feature List type 180
- Feature Requirement Specification type 180
- Feature Rule
 - relationship 145
 - type 180
- Feature Selection Type attribute 87
- Feature Specification type 180
- Feature Task Breakdown relationship 145
- Feature Test Case relationship 145
- Feature Test Specification relationship 145
- Feature type 180
- Feature Use Case relationship 145
- Features Specification relationship 145
- Features type 180
- field
 - parts in 81
 - parts returned from 81
- Field Column Headers setting 800
- Field Row Headers setting 800
- Field Size setting 800
- Field Table Columns setting 800

- Field Table Rows setting 800
- Field Type setting
 - for form object fields 801
 - for table object columns 800
- fieldNameActual parameter 821
- fieldNameDisplay parameter 821
- file
 - install log file
 - lists name collisions 24
 - upload limit 470
- File Collaboration Server 438
- File Contents attribute 87
- File Extension attribute 87
- File Format
 - type 181
- File Site attribute 87
- file store for company 426
- File Store Symbolic Name attribute 87
- filter
 - admin objects used in policy filters 66
 - custom for table 621
 - tool 581
- Filter tool 794
- FilterFramePage parameter 611, 820
- FilterFrameSize parameter 611, 820
- FilterParameterPattern 468
- Final Package Review state
 - renamed 62
- Final Quotation type 181
- Finance Comments attribute 87
- Finance Status attribute 87
- Financial Benefit Category type 181
- Financial Category Administrator role 168
- Financial Cost Category type 181
- Financial Item type 181
- Financial Items relationship 145
- Financial Reviewer role 168
- Financial Sub Categories relationship 145
- Financials type 181
- Find Number attribute 87
- Find Type attribute 87
- FindSelection 472
- First Name attribute 87
- Fixed In relationship 145
 - deprecated 65
- Fixed Resource type 181
- Folder Access attribute 87
- Folder Document relationship
 - deprecated 65
 - described 146
- folder grants 469

- FolderContentTypesThatTypesRequireGrants
 - property 469
- fonts 497
- form 566
 - See also* form page.
 - committing changes 670
 - decision and options for fields 650
 - defining fields 651
 - field
 - alternate OID 655
 - alternate type icon 655
 - arranged as table 663
 - combo box 657
 - dates 660
 - definingfield. *See* form and form page.
 - embedded configurable table 665
 - group 662
 - hyperlink and type icon 654
 - hyperlinked data 655
 - hyperlinked image 656
 - JPO interface 670
 - popup range helper 666
 - radio buttons 659
 - section header 661
 - section separator 661
 - select expressions 651
 - validating data 482, 668
 - values from program 653
 - JPO for field values 674
 - parameters for 632
 - properties 482
 - settings for 635
 - using Action menus with 647
- form page
 - See also* form.
 - committing changes 670
 - creating 630
 - decision and options for fields 650
 - edit and refresh 648
 - Edit link 647
 - Edit mode 627
 - html components 629
 - linking to app 630
 - links 628
 - mechanisms that define 629
 - preventing multiple subbits 670
 - refreshing after edit 670
 - two modes 627
 - URL for 631
 - View mode 627
- form parameter 644, 821
- format for names 468

- format for string resource IDs 448
- Format parameter 610, 786
- format setting 802
- formats
 - assigning to viewers 430
 - in framework 124
- formEditDisplay frame 629
- formEditFooter frame 629
- formEditHeader frame 629
- formEditHidden frame 629
- formHeader parameter 644, 821
- formName parameter 822
- Formula Characteristic type 181
- Formula Technical Specifications type 182
- formViewDisplay frame 629
- formViewHeader frame 629
- formViewHidden frame 629
- Found In relationship 146
 - deprecated 65
- Frame Part type
 - described 182
 - in Part inheritance model 29, 31
- frameName 822
- frameName parameter 822
- framework. *See* Application Exchange Framework.
- framework.properties 463
- frmFormView form 629
- FTP Directory attribute 87
- FTP Host attribute 87
- FullName.Format 468
- function for table column 592
- function setting 802
- Functional Specification type 182
- Further Chemical Physical Characteristic type 182

G

- GBOM From relationship 146
- GBOM relationship 146
- GBOM To relationship 146
- GBOM type 182
- General Description of Change attribute 87
- General menu 128
- General Requirements attribute 88
- General Search command 121
- Generic 182
- Generic Characteristic type 182
- Generic Document type 182
- generic format 124
- getAliasForAdmin 777
- getSchemaProperty 776
- getSelectBusinessObjectData method 605

- Global Attribute List attribute 88
- Global Read attribute 88
 - used in expression access 67
- global search menu 128
- Global Template Administrator role 168
- global toolbar
 - overview
 - toolbar
 - global
 - overview 515
- global toolbar menu 128
- Global User role 168
- Goal attribute 88
- Governed By relationship 146
 - used in expression access 69
- Governing ECO attribute 88
- Governing Project relationship 146
- grants for folder 469
- group
 - configuring 375
 - list of 125
- Group Count setting 802
- Group Header setting 596, 803
- grouped fields 662
- grouping table columns 596
- guide, how to use 18

H

- hacker protection 468
- Hardware Build type 182
- Hardware Part type
 - described 182
 - in Part inheritance model 29, 31
- Hardware Product type 182
- Hardware Vendor attribute 88
- Has Books relationship 146
- Has Bookshelves relationship 146
- Has Documents relationship 146
 - used in expression access 70
- header 497
- Header parameter
 - for emxHistory.jsp 822
- header parameter 612, 693, 710, 822
 - for emxPortal.jsp 822
- header repeat 481
- headerRepeat parameter 612, 822
- Heading parameter 786
- Headquarters Site attribute 88
- Height attribute 88
- Height parameter 786
- help 21

- Help About command 121
- Help Marker setting 803
- HelpMarker parameter 612, 644, 693, 710, 822, 834
- Hidden Attribute List attribute 88
- hiddenFrame frame 587
- Hide Label setting 803
- Historical Version relationship 146
- History All Revisions command 122
- History All Versions command 122
- history and User Agent 131
- History command 122
- History commands 727
- History page 727
- History.SpecialActionType 469
- HistoryMode parameter 823
- Holder type 182
- Home command 122
- home page
 - changing onscreen text 406
 - PLM link 480
 - properties 480
- home page preference 726
- Home Page Preference command 122
- Home Phone Number attribute 88
- Home.ContentPage 480
- Host Meetings attribute 88
- Hours Delta attribute 88
- HPGL format 124
- Href parameter 787
- html components
 - of form page 629
- html form 566
- HTTP Request Holder relationship 146
- HTTP Request Holder type 183
- HTTP Request relationship 146
- HTTP Request type 183
- hyperlink for form field 654
- hyperlinked image 656

I

- i18n 443
- icon
 - column type 582, 795
 - for table column 581, 586, 642, 794, 812
 - for table columns 594
 - for toolbar tools 803
 - for tree category 552
 - type's tree menu 560
- Icon Mail
 - attribute 88

- menu 129
- Icon parameter 787
- IconMail command 122
- IconMail properties 472
- IconMailLanguageLocale property 456
- IconMailLanguagePreference property 456, 724
- IGES format 124
- image
 - configuring images 497
 - for contact page 461
 - for toolbar tools 803
 - for tree category 552
 - login page 461
- Image format 124
- Image Horizontal Size attribute 88
- Image relationship 147
- Image setting 803
- Image type 183
- Image UOM attribute 89
- Image Vertical Size attribute 89
- Impact
 - attribute 89
- Impact Analysis Effort attribute 89
- Impact Analysis relationship 147
 - used in expression access 70
- Impact Analysis type 183
- Implementation Effort attribute 89
- Inbox Task
 - type
 - described 184
 - in Team Central data model 50
- inbox task
 - automation for 768, 769
- Incident
 - type 184
- Incident Disposition attribute 89
- Incident Document relationship 147
 - deprecated 65
- Incident Entry Clerk role 168
- Incident Fixed In relationship 147
- Incident Found In relationship 147
- Incident Owned By relationship 147
- Incident Owned By relationship, deprecated 65
- Incident Submitted By relationship 147
- Incident Validation relationship 147
- include page for login page 461
- Includes Supplier Part relationship 147
- IncludeSubTypesForGrants property 469
- InclusionList parameter 823
- Incompatible relationship 147
- Info Central application
 - does not require framework 14

- independent of data 25
- Initial Package Review state
 - renamed 62
- Initial Quotation type 184
- Initial Resource attribute 89
- initialization properties 473
- InitializationFile 473
- Initiated Template Project relationship 147
- Initiating Route Template relationship 147
- Input Type setting 804
- inquiry
 - parameters for 609
 - used to build table 571
 - using for expanding tree category 561
- inquiry parameter 612, 823
- inquiry setting 804
- inquiryLabel parameter 612, 824
- install log files
 - lists name collisions 24
- installed date property 381
- installedViewer property 428
- installer property 381
- installFramework.log
 - lists name collisions 24
- Instance Of relationship 147
- Instance Parameters attribute 89
- Internal Design relationship 147
- internationalizing
 - commands 488
 - date and time 446
 - debugging 446
 - documentation 454
 - MatrixOne Applications 443
 - menus 488
 - schema names 447
 - system notifications 455
- Interval Date attribute 89
- Interval Item Data type 184
- Is Template attribute 89
- Is Version attribute 89
- Is Version Object attribute 89
- Issue
 - relationship 147
 - type 185
- Issue Category attribute 89
- Issue Category Classification relationship 147
- Issue Category type 185
- Issue Classification attribute 90
- Issue Classification type 185
- issue data model 55
- Issued Date attribute 90
- Items parameter 788

Items To Change relationship 147
Items To Obsolete relationship 148

J

Java plug-in for applet 470
Java. See JSP or JPO programs.
JavaScript validation
 custom 482, 669
 standard 668
JPO
 and trigger manager 420
 constructor arguments for 83
 for form field values 653
 for getting field range values 673
 for getting field values 672
 for getting list of objects 618
 for sorting a table 624
 for structure tree 563
 for table column 590, 605
 for updating field values 674
 function for table column 592
 getting names of admin objects 776, 777
 getting symbolic names of admin objects 777
 guidelines for writing 617
 interface for form fields 670
 method name 83
 program for table column 585, 641, 808, 809
 sample for controlling access 495
 sample for form field values 674
 to update column values 620
 to update table values 601
JSP programs
 and object generator 412, 414, 415
 changing onscreen text 406
 error handling 747
 getting names of admin objects 776, 777
 getting symbolic names of admin objects 777
 implementing schema translations in 449
 internationalizing 443
 referring to objects in 379
 sample for custom table filter 622
 sample of internationalized 445
jsTreeID parameter 558, 824, 834
JT format 124

K

Keywords attribute 90
Kind of Product attribute 90

L

Label attribute 90
Label Function setting 805
Label parameter 788, 789
Label Program setting 805
Label setting 805
Lag Time attribute 90
language
 internationalizing applications 443
 introducing new for applications 444
 preferred for notifications 723
 system notifications 455
 translating schema names 447
Language attribute 90
Language Preference command 122
Last Legacy Update attribute 90
Last Login Date attribute 90
Last Name attribute 90
Latest Version relationship 148
lead time
 production 112
 prototype 112
 tooling 112
Lead Time Assessment attribute 90
Lead Time attribute 90
Lead Time Production attribute 90
Lead Time Prototype attribute 91
Lead Time Tooling attribute 91
Ledger Account Number attribute 91
Left Expression attribute 91
Left Expression relationship 148
LeftImage 480
Legacy attribute 91
Legacy System Identifier attribute 91
Legacy System Source attribute 91
legend for data models 27
Length attribute 91
Library type 185
Life Cycle Cost Issues attribute 91
lifecycle
 inbox task 768, 769
 organization 769
 organizations 770
 person 770
 routes 766
 signature approvals 440
 task 768, 769
 tasks 766
Lifecycle command 122
Lifecycle Demote command 122
lifecycle menu 128

- Lifecycle Promote command 122
- LifeCycle.ApprovalPasswordConfirmation 467
- Limited Author role 168
- Line Item
 - relationship
 - described 148
 - type
 - described 185
- Line Item Object relationship 148
 - used in expression access 70
- Line Item relationship
 - used in expression access 70
- Line Item Roll-up Calculation Formulas
 - attribute 91
- Line Item Roll-up Formulas attribute 91
- Line Item Split relationship
 - described 148
 - used in expression access 70
- Line Item Supplier Exception relationship 148
- Line Item Template
 - type 185
- Line Item Template Scope attribute 91
- Link URL
 - attribute 91
 - relationship 148
- Link URL relationship
 - used in expression access 70
- List Member relationship 148
- List Price attribute 91
- listHidden frame 587
- Local Attribute List attribute 91
- Local Template relationship 149
- Location
 - type 185
- Location Code attribute 92
- Location Equivalent relationship 149
- Location Equivalent type 185
- Location Preference attribute 92
- Location Status attribute 92
- log file
 - timing 482
- log files
 - for installed versions 133
 - lists name collisions 24
- login 462
 - background 461
 - image 461
 - include page 461
 - page 459
 - requirements for 460
 - servlet 461
 - target page 461

- login page 406
- Login Type attribute 92, 462
- Logout command 122, 740
- logout command 462
- logout tool 462
- lookup methodology 392
- Lot Effectivity attribute 92
- Lower Limit attribute 92
- Lower Specification Limit attribute 92

M

- Machined Part type
 - described 186
 - in Part inheritance model 29, 31
- macro
 - for common directory 490
 - for directory names 490
 - for number of selected rows in table 491
 - for object name 490
 - for object revision 490
 - for object type 490
 - for root directory 490
 - for select expressions 490
 - for suite directory 490
- Make Obsolete relationship 149
- make/buy code
 - production 99
 - service 110
- manual data conversions 849
- Manual Section relationship 149
- Manual Title attribute 92
- manual, how to use 18
- Manufacturability attribute 92
- Manufacturer Equivalent History relationship 149
- Manufacturer Equivalent relationship 149
- Manufacturing Assessment type 186
- Manufacturing Engineer role 168
- Manufacturing Process type 186
- Manufacturing Responsibility relationship 149
 - used in expression access 70
- Manufacturing Site attribute 92
- Marketing Feature attribute 92
- Marketing Manager role 168
- Marketing Name attribute 92
- Marketing Text attribute 92
- Markup
 - format 124
 - relationship 149
 - type
 - described 186
 - in Engineering Central data model 30

- Markup Author attribute 92
- Markup Base Filename attribute 92
- Markup Base Format attribute 93
- Markup Connection ID attribute 93
- Markup Function attribute 93
- Markup Tool attribute 93
- mass updates for editable tables 576
- massUpdate parameter 612, 824
- Material Category attribute 93
- Material Limitation attribute 93
- Matrix Navigator
 - accessing admin wizards 205
 - adding person 374
 - changing Company Name 371
 - configuring autonaming 413
 - configuring triggers 422
 - connecting employee to company 375
 - registering admin objects 382
 - registering viewers 428
- MatrixIniDefaults program 133
- MatrixOne Applications
 - data models for 29
 - documentation 18
 - how related to framework 14
 - internationalizing 443
 - items in 16
 - list of 14
- MatrixOne applications
 - accessing externally 750
 - configuring 379
- MatrixOne Document Central
 - data model 29
- MatrixOne Engineering Central
 - data model 30
- MatrixOne Product Central
 - data model 34
- MatrixOne Program Central
 - data model 38
- MatrixOne Sourcing Central
 - data model 41
- MatrixOne Specification Central
 - data model 46
- MatrixOne Supplier Central
 - data model 47
- MatrixOne Team Central
 - data model 50
- Max Palette Height attribute 93
- maximum column value 604
- Maximum Length setting
 - for form object field 805
 - for toolbar menu object 805
- Maximum Packaging Height attribute 93
- Maximum Quantity attribute 93
- MBOM relationship 149
- MCAD Assembly Instance type 186
- MCAD Assembly type 186
- MCAD Component Instance type 186
- MCAD Component type 186
- MCAD Model type 186
- MCAD Parametrized Assembly Model type 186
- MCAD Parametrized Component Model type 187
- MCAD Parametrized Model type 187
- Mean attribute 93
- Measure of Success attribute 93
- Mechanical Part type
 - described 187
 - in Part inheritance model 29, 31
- Media
 - type 187
- median column value 605
- Meeting
 - type
 - described 187
 - in Team Central data model 51
- Meeting Attachment relationship
 - used in expression access 70
- Meeting Attachments relationship 150
- Meeting Attendee attribute 93
- Meeting Content relationship 150
- Meeting Duration attribute 93
- Meeting ID attribute 93
- Meeting Key attribute 93
- Meeting Location attribute 94
- Meeting Owner attribute 94
- Meeting Password attribute 94
- Meeting Site ID attribute 94
- Meeting Sitename attribute 94
- Meeting Start Date Time attribute 94
- Meeting Username attribute 94
- Member Folder Access relationship
 - deprecated 65
 - described 150
- Member Folder relationship
 - deprecated 65
 - described 150
- Member List relationship 150
- Member List type 187
- Member relationship 150
- Member Route relationship 150
- Member Task relationship 150
- Member Topic Access relationship 150
- membership agreement 462
- menu mode 750
- MenuName parameter 508, 824

- menus
 - building in dynamic UI 515
 - installed with framework 128
 - internationalizing 488
 - types of 513
- MenuTab parameter 824
- Message
 - data model 57
 - relationship 150
 - type
 - described 187
 - in Team Central data model 50
- Message Attachment relationship
 - used in expression access 70
- Message Attachments relationship 151
- Message relationship
 - used in expression access 70
- Message URL Label setting 806
- messages for company 407
- method
 - for refreshing table 588
- Metric Integer attributes 94
- Metric Real attributes 94
- Metric Source attribute 94
- Microbiological Characteristic type 187
- Middle Name attribute 94
- minimum column value 605
- Minimum Quantity attribute 94
- modality 585, 641, 806
- mode parameter 558, 645
 - for emxForm.jsp 825
 - for emxNavigator.jsp 750, 825
 - for emxTree.jsp 825
- Model Product relationship 151
- Model type 187
- Model Type attribute 94
- Molded Part type
 - described 187
 - in Part inheritance model 29, 31
- Monitor command 122
- Move Files to Version attribute 94
- MQL command 122
- MQL notice 776
- MQL Script attribute 94
- MQL scripting 379
- MSDS format 124
- MX_TRIGGER_RECURSION_DETECTION 417
- My Desk menu 129, 514

N

- name format 468
- name macro 490
- Name parameter 789
- names
 - administrative objects 776, 777
 - automatic for business objects 409
 - changing administrative object 388
 - changing automatic name 413
 - collisions 380, 385
 - company name 176
 - conventions
 - dynamic UI components 486
 - custom applications 511
 - do not change program names 388
 - number in 409
 - of applications programs 421
 - prefix for automatic 84
 - states 384, 396
 - suffix for automatic 84
 - symbolic name of admin objects 777
 - trigger programs 421
- navigation 83
- navigation trees
 - + sign 561
 - API for 538
 - building 530, 712
 - calling alternate for a type 559
 - calling custom 559
 - configuring dynamic expand for category 561
 - configuring structure tree 532
 - count for category 548
 - default tree 559
 - deprecated methods 542
 - introduction to building 526
 - modes 528
 - overview 526
 - parameters for 543
 - parameters for categories 547, 688
 - settings for categories 550, 689
 - sub-tree 532
- Navigator page
 - accessing externally 750
 - configuring 507
 - defining default home page 507
- New Part/Part Revision relationship 151
- New Specification/Specification Revision
 - relationship 151
- Next Message attribute 95
- No Bid Reason attribute 95
- Notes attribute 95

- notice, MQL 776
- NotificationAgent 471
- NotificationLanguages 455, 472
- notifications
 - customizing for company 407
 - internationalizing 455
- Nowrap setting 806
- Number Generator type 409
- number in names 409
- Nut Part type
 - described 187
 - in Part inheritance model 29, 31

O

- object generator
 - attributes 411
 - calling from JSP program 415
 - calling from Tcl program 415
 - parameters 411
- Object Generator program. See eServicecom-monObjectGenerator.tcl program
- Object Generator type 409
- Object List Map 597
- Object PowerView
 - linking page to MatrixOne Applications 701
- Object Route relationship 151
- object tree. *See* navigation trees.
- object type. *See* types.
- objectBased parameter 613, 825
- ObjectGeneratorName parameter 412
- ObjectGeneratorRevision parameter 412
- objectID parameter 645
- objectId parameter 508, 558, 589, 825, 834
- Objective attribute 95
- ObjectPolicy parameter 412
- objects. *See* administrative objects or business objects.
- ObserveHidden parameter 826
- Odour attribute 95
- online help 21
- On-line Meeting Instructions attribute 95
- On-line Meeting Provider attribute 95
- onscreen text 406
- Operational Definition attribute 95
- Opportunity attribute 95
- Option
 - Ordered Option attribute 95
 - type 188
- Option List relationship 151
- Option Rule
 - relationship 151
 - type 188
- Order Entry Clerk role 168
- Order Entry group 126
- Order Quantity attribute 95
- Order Recipient attribute 95
- Ordered Item relationship 151
- Ordered Options attribute 95
- Organization
 - automation for 769
 - type
 - described 188
 - in Team Central data model 50
- Organization Context relationship 151
- Organization Data Document relationship 151
- Organization Fax Number attribute 95
- Organization Goal relationship 151
- Organization ID attribute 95
- Organization List
 - type 188
- Organization List Item relationship 151
- Organization Location relationship 152
- Organization Manager role 169, 374
- Organization Name attribute 95
- Organization Phone Number attribute 95
- Organization Region relationship 152
- Organization Representative relationship 152
- Organization RTS Template relationship
 - described 152
 - renamed 62
- Organization Specification Office relationship 152
- Organization Template relationship 152
- organization, automation for 769
- organizations
 - policy and lifecycle 770
 - status 770
- Origin attribute 95
- original name property 381
- Originating Requirement relationship 152
- Originating Specification Template
 - relationship 152
- Origination attribute 96
- Originator attribute
 - described 96
 - program that sets 764
 - used in expression access 67
- Other Part type
 - described 188
 - in Part inheritance model 29, 31
- Other Requirements Affected attribute 96
- Other Tasks Affected attribute 96
- Out of Bounds attribute 96
- ownership 27

Owning Organization relationship 152

P

Package

triggers 770

type 188

Package Holder relationship 152

Package RFQ relationship 152

Package RTS relationship 152

renamed 62

Package Sent state

renamed 62

Packaging Characteristic type 188

Packaging Size attribute 96

Packaging Technical Specifications type 189

Packed On Date attribute 96

Packed Requirement attribute 96

Page History command 123

pageHistory.limit 467

Pager Number attribute 96

pagination controls 567

pagination parameter 613, 826

Pagination Preference command 123

PaginationRange 481

Palette Size attribute 96

Parallel Node Procession Rule attribute 96

parameters automatically passed to URLs 834

parameters for

emxTable.jsp 611

emxTree.jsp 556

forms 632

inquiry objects 609

navigation trees 543

portal menu objects 707

tables 578

tabs 707, 709

tree categories 547, 688

Parent Markup relationship 152

Parent Part ID attribute 96

Parent Route Node attribute 96

parent type. See type inheritance.

parentOID parameter 826, 834

Part

access to 111

Annual Part Quantity attribute 74

in field 81

in process 81

in stock 81

on order 82

returned 81

type

described 189

in Engineering Central data model 30

inheritance 29, 31

Part Classification attribute 97

Part Family

type 189

Part Family Base Number attribute 97

Part Family Coordinator role 169

Part Family Last Used Sequence attribute 97

Part Family Member relationship 152

Part Family Name Generator On attribute 97

Part Family Pattern Separator attribute 97

Part Family Prefix Pattern attribute 97

Part Family Reference Document relationship 153

Part Family Sequence Pattern attribute 97

Part Family Suffix Pattern attribute 97

Part Master

type 189

Part Quality Plan Template type 189

Part Quality Plan type 189

Part Revision relationship 153

Part Rule

relationship 153

type 189

Part Specification relationship 153

Pass Through Line Item relationship 153

Pass Through RFQ relationship 153

password

change 460

expiration 460

for shadow agents 439

verification 467

Pattern parameter 610, 789

PDF format 124

Pending Line Item Attribute Group

relationship 153

Pending Version relationship 153

Percent Complete attribute 97

Percent Concentration attribute 97

Percentage attribute 97

performance of table columns 605

Period attribute 97

Period Type attribute 98

Person

adding 373

administrative object type 130

for testing 130

type

described 190

in Team Central data model 50, 51

person

automation for 770

- person name format 468
- Personal Folder
 - policy
 - deprecated 64
 - type 190
 - deprecated 65
 - PH attribute 98
 - phone number
 - home 88
 - organization 95
 - pager 96
 - work 119
 - Physical Characteristic type 190
 - Plan
 - type 190
 - Plan Deliverable relationship 153
 - Plan Focus relationship 153
 - Plan Location relationship 153
 - Plan Manager relationship 153
 - Planned Benefit attribute 98
 - Planned Build Date attribute 98
 - Planned Cost attribute 98
 - Planned Feature relationship 153
 - Planned Net Benefit attribute 98
 - Plant ID attribute 98
 - Plant Representative role 169
 - Plant Scorecard
 - type 190
 - Platform attribute 98
 - PLM link 480
 - policy
 - accesses for 207
 - admin objects used in access expressions 66
 - changing state names 396
 - inbox task 768, 769
 - organization 769
 - person 770
 - states 207
 - task 768, 769
 - policy states
 - formats for string resource IDs 449
 - Popup Modal setting 806
 - popup range helper
 - field with 666
 - implementing 667
 - popup windows 585, 641, 806
 - portal page
 - building 703
 - channel tabs 698
 - linking 703
 - mechanisms that define 699
 - overview 698
 - parameters for 707
 - PowerView 696
 - SummaryView 697
 - portal parameter 693, 709, 752, 826
 - portalpage
 - PowerView 697
 - portalTable parameter 613, 710, 827
 - Postal Code attribute 98
 - PowerView
 - linking page to VCP 700
 - PowerView Actions menu 128
 - PowerView command 123
 - PowerView menu 128
 - PowerView portal page 696, 697
 - Precise BOM attribute 98
 - deprecated 64
 - Precise BOM relationship 153
 - PreExpand Category setting 561, 807
 - PreExpand Function setting 808
 - PreExpand setting 807
 - preference
 - currency conversion 717, 720
 - export table data format 724
 - home page 726
 - language 723
 - unit of measure 720
 - Preference command 123
 - preference_Command property 726
 - preference_FieldSeparator property 725
 - preference_Menu property 726
 - preference_RecordSeparator property 725
 - preference_RemoveCarriageReturns property 725
 - Preferences attribute 98
 - Preferences menu 129
 - Preferences.FieldValueSeparator.Delimiter 726
 - preferred file store for company 426
 - preferred viewer 432, 433
 - preFilter parameter 827
 - prefix
 - for automatic names 84, 413
 - pre-processing for tree category 561
 - pre-production database 379
 - price
 - access to 111
 - Agreed Unit Price attribute 74
 - Base Price attribute 75
 - Container Cost attribute 78
 - list 91
 - unit 116
 - Primary Contact attribute 98
 - Primary Image relationship 153
 - Primary Key attribute 99

- used in expression access 67
- Primary Process relationship 154
- primary type 410
- primary vault 378
- Printer Friendly setting 808
- PrinterFriendly parameter 613, 645, 828, 834
- Prioritization Benefit attribute 99
- Priority attribute 99
- Problem Statement attribute 99
- Process
 - type 190
- Process Qualification Status attribute 99
- Process Technical Specifications type 191
- Product
 - attribute 99
 - Sourcing Product attribute 112
 - type 191
- Product Build relationship 154
- Product Configuration
 - relationship 154
 - type 191
- Product Configuration Build relationship 154
- Product Documentation
 - type 191
- Product Features type 191
- Product Line
 - type
 - described 191
 - in Engineering Central data model 30
- Product Line Models relationship 154
- Product Management group 126
- Product Manager role 169
- Product Obsolescence Manager role 169
- Product Requirement relationship 154
- Product Requirement Specification type 191
- Product Specification relationship 154
- Product Specification type 191
- Product Support Clerk role 169
- Product type
 - deprecated 65
- Product Version relationship 154
- production
 - database 379
 - lead time 112
- Production End Date attribute 99
- Production Make Buy Code attribute 99
- Production Purpose attribute 99
- Production Start Date attribute 99
- Production Support group 126
- Products relationship 154
- Products type 192
- Profile Manager application

- data model 48, 49
- program argument
 - description of 84
 - for trigger program 84
- Program Lead role 169
- program parameter 614, 828
- Program Project relationship 154
- Program Rule Argument attribute 100
- Program Rule attribute 99
- Program setting 809
- Program type 192
- programLabel parameter 614, 828
- programs
 - Also see JPO.
 - described 132
 - do not change names 388, 421
 - emxTriggerManager 420
 - eServicecommonObjectGenerator.tcl 411
 - eServicecommonWzrdListFromCommand.tcl 404
 - for table columns 590
 - JSP 379
 - prefixes for names 421
 - referring to objects in 379
 - specifies range values 403
 - Tcl 379
 - that create objects 412, 414
 - trigger 421
- Project
 - policy
 - renamed 62
 - type
 - described 203
 - in Team Central data model 50
 - renamed 63
- Project Access attribute 100
- Project Access Key relationship 154
 - used in expression access 70
- Project Access List relationship 154
 - used in expression access 70
- Project Access List type 192
- Project Administrator role 169
- Project Assessment relationship 154
- Project Assessor role 169
- Project Concept
 - type 192
- Project Financial Item relationship 154
- Project Folder
 - policy
 - deprecated 64
 - type
 - deprecated 65

- described 192
 - in Team Central data model 50
- Project Folders relationship
 - deprecated 65
 - described 154
- Project Function attribute 100
- Project Lead role 169
- Project Management type 192
- Project Member
 - type
 - described 193
 - in Team Central data model 50
- Project Member Access attribute 100
- Project Members relationship 154
- Project Membership relationship 154
- Project Name attribute 100
- Project Originators group 126
- Project Question relationship 154
- Project Release
 - relationship 154
- Project Requirement relationship 154
- Project Role attribute 100
- Project Route relationship 155
- Project Routing Slip relationship 155
- Project Security Approach attribute 100
- Project Space
 - type 193
- Project Space Access Grantor person 130
- Project Task
 - relationship 155
- Project Task Actual Duration attribute 100
- Project Task Actual Finish Date attribute 101
- Project Task Actual Start Date attribute 101
- Project Task Duration attribute 101
- Project Task Finish Date attribute 101
- Project Task Resource attribute 101
- Project Task Start Date attribute 101
- Project Template
 - type 193
- Project Type attribute 101
- Project User group 126
- Project User role 169
- Project Vault
 - policy
 - renamed 62
 - type
 - described 203
 - in Team Central data model 50
 - renamed 63
- Project Vaults relationship
 - described 141
 - renamed 62

- Project Visibility attribute 101
- Promised Delivery Date attribute 101
- Properties Form Name attribute 101
- Properties Template Form Name attribute 101
- property
 - application 381
 - emxFrameworkStringResource.properties 406
 - for 2D/3D viewer 466
 - for application 482
 - for banner 480
 - for date/time format 474
 - for email notification 471
 - for Engineering Central 473
 - for form 482
 - for framework directories 479
 - for home page 480
 - for IconMail 472
 - for initialization 473
 - for internationalizing 443
 - for login page 460
 - for resource file 479
 - for search 464
 - for special characters 470
 - for start page 480
 - for symbolic names 379
 - for table 481
 - for tree 480
 - installed date 381
 - installedViewer 428
 - installer 381
 - onscreen text 406
 - original name 381
 - state names
 - customizing 396
 - described 384
 - transferring to another admin object 389
 - version 381
 - viewerTip 428
- PropertyFileAlias 473
- Proposed Markup relationship 155
- Prototype Capability attribute 101
- prototype, lead time 112
- Public Add association 72
- Public attribute 101
- Public Delete association 72
- Public Folder
 - policy
 - deprecated 64
 - type 193
 - deprecated 65
- Public Modify association 72
- Public Read association 72

- Publish relationship 155
- Publish Subscribe
 - relationship 155
 - type
 - described 193
 - in Team Central data model 51
- publish subscribe data model 58
- Pull Right setting 809
- Purchase Class
 - relationship 155
- Purchase Class Manager relationship 155
- Purchase Class Member relationship 155
- Purchase Class Scorecard
 - type 193
- Purchase Class type 193
- Purchase Decision attribute 101
- Purchase Order attribute 101
- Purchase Subclass relationship 155
- push and pop to Shadow Agent 773
- Pushed Subscription relationship 155

Q

- Quality Administrator role 169
- Quality Assessment attribute 101
- Quality Impact attribute 101
- Quality Manager role 169
- Quality Metric relationship 155
- Quality Metric type 194
- Quality Plan Part relationship 155
- Quality relationship 155
- Quality type 194
- Quality Type attribute 102
- quantity
 - Annual Part Quantity attribute 74
 - maximum 93
 - minimum 94
 - order 95
- Quantity attribute 102
- quantity fields 471
- Question
 - relationship 155
 - type 194
- quotation
 - triggers 771
- Quotation Header Roll-up Calculation Formulas
 - attribute 102
- Quotation Header Roll-up Formulas attribute 102
- Quotation Response Review attribute 102
- Quotation type 194
- Quote relationship 156
- Quote Requested by Date attribute 102

R

- radio buttons 659
- Raised Against ECR relationship 156
- Range Function setting 809
- Range Program setting
 - for form object fields 809
 - for table object columns 809
- range values
 - specified by program 403
 - specified manually 405
- Rate attribute 102
- Rate Period
 - relationship 156
 - type 194
- Raw Material Package attribute 102
- Raw Material Technical Specifications type 195
- Reason For Cancel attribute 102
- Reason For Change attribute 102
- Reason for Closure attribute 102
- Receiving Plant attribute 102
- Recipient attribute 102
- recipient of order 95
- Recommended Supplier relationship 156
- Recommended Suppliers relationship
 - used in expression access 70
- Recurrence Interval attribute 102
- Reference Designator attribute 102
- Reference Document relationship
 - described 156
 - used in expression access 70
- refreshTablePage method 588
- region 113
- Region attribute 103
- Region Owns relationship 156
- Region type 195
- Regional Attribute List attribute 103
- Regional Template relationship 156
- register here link 462
- Registered Suite setting 810
- RegisteredDirectory parameter 645, 829
- registering
 - group 375
 - objects 389
 - viewers 427, 428
- Regression attribute 103
- Regression Version attribute 103
- Related Concept Project relationship 156
- relationships
 - attributes for 132
 - clone rule 132
 - described 132

- for additional objects 83
 - revision rule 132
 - shown on data models 27
- Relative Cost attribute 103
- Relative Penalty attribute 103
- Relative Risk attribute 103
- Release attribute 103
- Release Date attribute 103
- Release Distribution Group attribute 103, 404
- Release Manager role 169
- Release Type attribute 103
- relID parameter 829
- relId parameter 589, 645, 834
- Reload Cache command 123, 466
- Reload Cache tool 166, 466
- ReloadOpener parameter 829
- rememberSelection parameter 614, 829
- RememberSelection property 482
- Renamed From attribute 103
- renaming
 - administrative objects 388
 - states 396
- Render Error Message attribute 103
- Render Language attribute 104
- Render Success Flag attribute 104
- Rendered File Name attribute 104
- Rendition In Process attribute 104
- Replacement Scope attribute 104
- Replaces Part relationship 156
- replacing objects 386
- Reply relationship 156
 - used in expression access 70
- Reporting Organization relationship 156
- Request Disposition attribute 104
- Request for Quote
 - relationship 157
 - triggers 771
 - type 195
- Request Impact Analysis relationship 157
 - deprecated 65
- Request Parameters attribute 104
- Request Part Obsolescence relationship 157
- Request Part Revision relationship 157
- Request Specification Revision relationship 157
- Request To Supplier
 - policy
 - renamed 62
 - type
 - renamed 63
- Request to Supplier
 - type 196
- Request type

- described 195
- Requested Delivery Date attribute 104
- Requested Work attribute 104
- Required Availability attribute 104
- Required setting 810
- Requirement attribute 105
- Requirement Baseline relationship 157
- Requirement Baseline type 195
- Requirement Breakdown relationship
 - renamed 62
- Requirement Category attribute 105
- Requirement Classification attribute 105
- Requirement Group relationship 157
- Requirement Importance attribute 105
- Requirement Manager role 169
- Requirement Satisfied By relationship 157
- Requirement Specification relationship 157
- Requirement Specification type 196
- Requirement type 195
- Requirement Use Case relationship 157
- Requirement Validation relationship 157
- requirements, other affected 96
- Requires relationship 157
- reserved prefixes, names, and characters 386
- Resistor Part type
 - described 196
 - in Part inheritance model 29, 31
- Resolution Recommendation attribute 105
- Resolution Statement attribute 105
- Resolve Effort attribute 105
- Resolved To relationship 157
- Resource Comments attribute 105
- resource file
 - naming convention 443
 - string naming 444
- resource file properties 479
- Resource ID attribute 105
- Resource Limit relationship 158
- Resource Maximum attribute 105
- Resource Minimum attribute 105
- Resource Operation attribute 105
- Resource Status attribute 105
- Resource Usage
 - attribute 105
 - relationship 158
- Resource Usage relationship 158
- resource, initial 89
- Responsibility attribute 106
- Responsible Design Engineer attribute 106, 404
- Responsible Manufacturing Engineer attribute 106
- Responsible Product Manager attribute 106, 404
- Responsible Sales Engineer relationship 158

- Restrict Members attribute 106
- Result attribute 106
- returned parts 81
- Review Comments Needed attribute 106
- Review Task attribute 106
- Reviewer role 169
- Reviewers Comments attribute 106
- Revise Search command 123
- revision macro 490
- revision rules, listed 134
- RFQ for Package Quotation relationship 158
- RFQ Holder relationship 158
- RFQ Line Item Template relationship 158
- RFQ Quotation
 - relationship 158
 - type 196
- RFQ quotation
 - triggers 771
- RFQ Supplier relationship 158
- RFQ Template
 - relationship 158
 - type 197
- RFQ Template Buyer Desk Association
 - relationship 158
- RFQ type 196
- Right Expression attribute 106
- Right Expression relationship 158
- RightImage 480
- Risk
 - relationship 159
 - type 197
- Risk Affected Items relationship 159
- Risk Comments attribute 107
- Risk Factor attribute 107
- Risk Impact attribute 107
- Risk Items relationship 159
- Risk Probability attribute 107
- Risk RPN relationship 159
- Risk RPN Value attribute 107
- Risk Status attribute 107
- Risk Type attribute 107
- Risk Visibility attribute 107
 - used in expression access 67
- role
 - inheritance 52
- Role Person relationship 159
- roles
 - all described 166
 - test persons assigned to 130
- root directory macro 490
- ROOT_DIR 490
- RootDirectory 479

- Route
 - type
 - described 197
 - in Engineering Central data model 30
 - in Team Central data model 50
- Route Access attribute 107
- Route Action attribute 108
- Route Base Policy attribute 108
- Route Base Purpose attribute 108
- Route Base State attribute 108
- Route Branch attribute 108
- Route Completion Action attribute 108
- route data model 53
- Route Delegation Grantor person 130
- Route Instruction attribute 108
- Route Node ID attribute 108
- Route Node relationship 159
- Route Scope relationship 159
- Route Sequence attribute 108
- Route Status attribute 108
- route task
 - signature approvals 440
- Route Task relationship 160
- Route Task User attribute 108
- Route Task User Company attribute 109
- Route Task User type 197
- Route Template
 - type
 - described 197
 - in Team Central data model 50
- Route Templates relationship 160
 - used in expression access 70
- routing 765
- Row Select setting 810
- row. *See* form and form page.
- Rows setting 811
- RPN
 - type 198
- RSA 462
- RTS Cancelled policy, renamed 62
- RTS for Package Quotation relationship 158
 - renamed 62
- RTS History Version policy, renamed 62
- RTS ID attribute 109
- RTS Part relationship
 - renamed 62
- RTS Pending Version policy, renamed 62
- RTS Quotation
 - policy
 - renamed 62
 - relationship 158
 - type 196

- renamed 63
- RTS Quotation Archive type 198
- RTS Quotation relationship, renamed 62
- RTS Removed Supplier relationship 160
- RTS Supplier Part
 - policy, renamed 62
 - type, renamed 63
- RTS Supplier relationship 158
 - renamed 62
- RTS Template
 - policy
 - renamed 62
 - relationship, renamed 158
 - type, renamed 63
- RTS Template relationship
 - renamed 62
- Rule Extension
 - relationship 160
 - type 198
- Rule Type attribute 109
- RuleExtension ID attribute 109
- running MQL command 122

S

- Safety and Regulatory Characteristic type 198
- Safety Description attribute 109
- Safety Symbol attribute 109
- Safety Warning attribute 109
- Sales Engineer role 169
- Sales Order
 - relationship 160
 - type 198
- sample
 - data 120, 130
 - groups 125
 - users 130
- Sample Change Board group 126
- Sample Distribution Group group 126
- Save As command 123
- Save Search command 123
- Saved Searches command 123
- Schedule Comments attribute 109
- Schedule Duration Unit attribute 109
- schedule impact 85
- Schedule Status attribute 109
- Scheduled Completion Date attribute 109
- schema names 447
- schema scripts 379
- SCO type 198
- Scope attribute 109
- Score By relationship 160
- Score For relationship 160
- Scorecard
 - type 198
- Screw Part type
 - described 198
 - in Part inheritance model 29, 31
- search
 - default vaults 377
 - popup 466
 - properties 464
 - types 465
 - valut criteria 377
- Search Actions menu 128
- Search menu 128
- Search Result menu 128
- search results limit 464
- Search Type attribute 109
- Search types menu 128
- Secondary Keys attribute 110
 - used in expression access 67
- secondary vault 378
- Secondary Vaults attribute 110
- Section Access attribute 110
- section header 661
- Section Level setting 811
- section separator 661
- Section Type attribute 110
- secure ID
 - displaying field on login page 462
 - integrating with RSA 462
 - requiring user to enter 92
- security 468
- Security Classification attribute 110
 - used in expression access 67
- select expression macros 490
- select expressions
 - for dynamic UI 488
 - for form fields 651
 - for table columns 590
- Selectable in Preferences setting 811
- SelectAbstractTypes parameter 829
- Selected Options relationship 160
- selection parameter 615, 830
- selection type 87
- SelectType parameter 830
- Send Reminder attribute 110
- Senior Design Engineer role 170
- Senior Manufacturing Engineer role 170
- Senior Quality Engineer role 170
- separator
 - for table columns 596
- Separator command 123

- sequence for trigger programs 85
- Sequence Order attribute 110
- Serial Effectivity attribute 110
- ServerTimOutInSec 470
- Service Make Buy Code attribute 110
- Service Product type 198
- settings 791
- settings for
 - forms 635
 - structure tree categories 555
 - tables 580
 - tabs 709
 - tree categories 550, 689
- Severity attribute 110
- shadow agent
 - password 439
- Shadow Agent group
 - assigned All access for every policy 207
 - described 126
 - push and pop to 773
 - User Agent assigned to 131
- Shape Limitation attribute 110
- Share Types attribute 110
- Sheet Number attribute 110
- Shelf Life attribute 111
- Ship To attribute 111
- Shipping Address attribute 111
- Shipping Clerk role 170
- Shipping Department group 126
- Ships To relationship 160
- Show Alternate Icon setting 811
- Show Clear Button setting 811
- Show Sub Components attribute 111
- Show Target Cost attribute 111
- Show This Supplier Price to Other Suppliers
 - attribute 111
- Show This Supplier to Other Suppliers
 - attribute 111
- Show Type Icon setting 812
- ShowFilterAction parameter 830
- ShowFilterTextBox parameter 830
- ShowIcons parameter 830
- ShowLangaugeForFileCheckin 468
- ShowSelects 472
- Sigma attribute 111
- signature approvals 440
- Similar Part relationship 160
- Single Signon 462
- site
 - headquarters 88
 - manufacturing 92
- Size Limitation attribute 111

- Sketch type
 - described 198
 - in Engineering Central data model 30
- smallIcon 560
- smallicon property 481
- Software Architecture
 - relationship 160
- Software Baseline relationship 160
- Software Build
 - relationship 160
 - type 198
- Software CCB Coordinator role 170
- Software Change Board group 126
- Software Dependency relationship 160
- Software Distribution
 - relationship 160
- Software Distribution type
 - renamed 63
- Software Engineer role 170
- Software Engineering group 126
- Software Engineering Manager role 170
- Software Feature
 - type 199
- Software Part type
 - described 199
 - in Part inheritance model 29, 31
- Software Platform
 - type 199
- Software Product type 199
- Software Project Manager role 170
- Software Quality Engineer role 170
- Software Quality Engineering group 127
- Software Quality Manager role 170
- Software Release
 - type 199
- Software Release Engineer role 170
- Software Release Engineering group 127
- Software Release Manager role 170
- Software Requirement Specification
 - relationship 161
- Software Requirement Specification relationship
 - deprecated 65
- Software Requirement Specification type
 - renamed 63
- Software Specification type 199
- Software Use Case
 - policy
 - renamed 62
- Software User Requirement policy
 - deprecated 64
- Software Vendor attribute 111
- Sort Program setting 624, 812

- Sort Type setting 812
- Sortable setting 812
- sortColumnName parameter 615, 830
- sortColumnName setting 813
- sortDirection parameter 615, 830
- sortDirection setting 813
- sorting a table 624
- sorttype parameter 624, 790
- Source Code Directory type 199
- Source Directory type 199
- Source File
 - type 199
- Source File relationship 161
- Source Qualification Status attribute 111
- Source Selection Status attribute 111
- Sourcing Central
 - currency conversion 717
- Sourcing Product attribute 112
- Sourcing Spares attribute 112
- Spare Part
 - attribute 112
 - relationship 161
- spares 112
- Spatial Location attribute 112
- Specific Description of Change attribute 112
- Specification Category attribute 112
- Specification Manager role 170
- Specification Office Manager role 170
- Specification Office type 200
- Specification Restricted User role 170
- Specification Section relationship 161
- Specification Section type 200
- Specification Template Holder type 200
- Specification Template relationship 161
- Specification type 199
- Specification User role 170
- Specification View Form Name attribute 112
- Spicer viewer. See viewer.
- Sponsoring Customer attribute 112
- spreadsheet property 468
- SSO 462
 - and signature approvals 440
- standard commands 740
- Standard Component relationship 161
- Standard Deviation attribute 112
- Standard Item relationship 161
- Standard Lead Time Production attribute 112
- Standard Lead Time Prototype attribute 112
- Standard Lead Time Tooling attribute 112
- Standards Manual
 - type 200
- Start Effectivity attribute 112
- Start Effectivity Date attribute 113
- start page properties 480
- Start State attribute 113
- State Comments attribute 113
- State/Region attribute 113
- states
 - changing names of 396
 - how the system identifies state names 384
- status
 - organizations 770
- Status attribute 113
- Steps to Reproduce attribute 113
- Storage Conditions attribute 113
- store, file store for company 426
- stores 172
- String Resource files
 - formats for IDs 448
 - using for internationalization 443
 - using to change onscreen text 406
- StringResourceFileId 479, 488
- StringResourceFileId parameter 830, 835
- Structure Function setting 813
- Structure Inquiry setting 813
- Structure Menu setting 813
- Structure Navigator 480, 529, 532
- Structure Program setting 813
- structure tree
 - API for 538
 - configuring 532
 - described 529
 - JPO program 563
 - settings for categories 555
 - shown 529
- Style parameter 615, 831
- style sheet
 - for table pages 615, 831
- style sheets 497
- Sub Componets Level attribute 113
- Sub Directories relationship 161
- Sub Goal relationship 161
- Sub Product Lines relationship 161
- Sub Region relationship 161
- Sub Requirement relationship 161
- Sub Route relationship 161
- Sub Route Visibility attribute 113
- Sub Test Case relationship 161
- Sub Topics relationship 161
- Sub Use Case relationship 161
- Sub Vaults relationship
 - used in expression access 70
- subHeader parameter
 - for emxHistory.jsp 831

- for emxTable.jsp 831
- subHeader parameter for portal 710
- subHeader parameter for table 616
- Subject attribute 113
- sub-menu 532
- Submit setting 813
- SubmitLabel parameter 616, 831
- Submitted Date attribute 113
- SubmitURL parameter 616, 831
- Subscribed Item relationship 162
- Subscribed Person relationship 162
- Subscription Level attribute 113
- subscriptions data model 58
- Subsidiary relationship 162
- Substitute
 - relationship 162
 - type
 - described 200
 - in Engineering Central data model 30
- Substituted Component relationship 162
- Subtask relationship 162
- sub-tree 532
- suffix for automatic names 84, 413
- suite
 - Also see application.
 - directory 482
- suite directory macro 490
- SUITE_DIR 490
- SuiteKey parameter 832
- suiteKey parameter 835
- sum of column values 604
- SummaryView portal page 697
- Supplier
 - relationship 162
 - role 170
- Supplier Buyer Line Item relationship 162
 - used in expression access 70
- Supplier Classification attribute 113
- Supplier Development Manager role 170
- Supplier Development Plan
 - type 200
- Supplier Engineer role 171
- Supplier Line Item
 - relationship 162
 - type 200
- Supplier Line Item Price Period
 - relationship 162
 - type 200
- Supplier Line Item relationship
 - used in expression access 70
- Supplier Part Request relationship 162
- Supplier Quality Engineer role 171

- Supplier Quote Analyst role 171
- Supplier relationship
 - used in expression access 70
- Supplier Representative role 171
- Supplier Response
 - attribute 113
 - relationship 162
- Supplier Scorecard
 - type 200
- supplier, access to 111
- Supplies relationship 163
- Supply Response relationship
 - used in expression access 70
- Supply Responsibility relationship 163
 - used in expression access 70
- Supported File Format relationship 163
- Supported Organization relationship 163
- Suspend Version attribute 113
- Symbol attribute 114
- symbolic name
 - allows name changes 380
 - benefits of using 380
 - do not change 379
 - facilitates upgrades 380
 - for states 384
 - getting admin object name from 776, 777
 - getting for admin object 777
 - JSP programs 379
 - reassigning 389
 - resolves name collisions 380
 - stored as properties 379
 - syntax for 379
 - Tcl programs 379
 - use to refer to admin objects 379
 - used to identify objects 379
- Synopsis attribute 114
- System Conversion Manager role
 - assigned All access for every policy 207
 - Corporate person is assigned to 130
 - described 171
- System Data command 123
- System Engineer role 171
- system notifications 471
- system notifications. See notifications.
- System Requirement Specification type 200
- System Specification type 200
- System Transition Manager role
 - assigned All access for every policy 207
 - described 171

T

table

See also table page and inquiry.

body definition 566

check boxes for columns 594

column calculation settings 604

column calculations 602

configuring editable 575

count of items in column 602

custom filter 621

defining columns 589

editable

adding to application 577

defining 577

defining columns 597

mass updates 576

embedding in form 665

filtering 581, 794

getting object Id from 589

grouping columns 596

header repeat 481

icons for columns 594

improving performance of 605

JPO for column 605

JPO for getting objects 618

parameters for 578

programs for column values 590

properties 481

refreshing to update 588

remembering selections 482

select expressions 590

selected count macro 491

settings for 580

sorting 624

special information 597

timing log 482

updating with JPO 601

Table Name attribute 114

table page

See also table.

building 571

how linked to application 571

mechanisms that define 570

overview 566

pagination 567

specific to business object 567

style sheet 615, 831

table parameter 616, 832

table selected count macro 491

table setting 814

tabs

controlling the layout in channel 702

parameters for 707, 709

settings for 709

Target Cost attribute 114

Target Location setting 814

target page after logging in 461

Task

other affected 96

relationship 163

type 200

task

automation for 768, 769

signature approvals 440

Task Actual Duration attribute 114

Task Actual Finish Date attribute 114

Task Actual Start Date attribute 114

Task Comments Needed attribute 114

Task Deliverable relationship 163

Task Edit Setting attribute 114

Task Escalation Message

relationship 163

type 200

Task Estimated Duration attribute 114

Task Estimated Duration Maximum attribute 114

Task Estimated Duration Minimum attribute 114

Task Estimated Finish Date attribute 114

Task Estimated Start Date attribute 114

Task Management type 201

Task Requirement attribute 114

Task Sub Route relationship 163

Task Transfer attribute 115

Task WBS attribute 115

Tcl programs

calls object generator 412, 414, 415

getting names of admin objects 777

getting symbolic names of admin objects 778

Team Membership relationship 163

Technical Feature attribute 115

Technical Specification relationship 163

Technical Specifications type 201

Technical Writer role 171

Template Administrator role 171

Template Header Administrator role 171

Template Level attribute 115

Template Task attribute 115

Test Case

type 201

test database 379

Test Everything person

administrative user 130

business object 120

Test parameter 790

- test persons 130
- Test Specification type 201
- Testing Process type
 - described 202
- Testing Responsibility relationship 163
 - used in expression access 71
- text
 - baseline 76
- Thread
 - relationship
 - described 164
 - type
 - described 202
 - in Team Central data model 50
- Thread relationship
 - used in expression access 71
- TIF format 124
- time
 - actual 74
 - format 474
 - incident occurred 115
 - internationalizing 446
 - lead 90
 - limit for automatic names 84
 - meeting start 94
- Time of Occurrence attribute 115
- Time Zone Preference command 123
- timeout 470
- timing 482
- tip for viewer 428
- Tip Page setting 815
- TipPage parameter 616, 645, 693, 710, 832, 835
- Title attribute 115
- token authentication 462
- Tolerance attribute 115
- Tolerance Lower attribute 115
- Tolerance Upper attribute 115
- toolbar
 - Change Password tool 741
 - Logout tool 740
- Toolbar menu 129, 515
- toolbar parameter 644, 693, 709, 832
- Tooling Assessment attribute 115
- Tooling Cost attribute 115
- Tooling Part type
 - described 202
 - in Part inheritance model 29, 31
- tooling, lead time 112
- ToolTip for viewer 428
- Total Budget Cost attribute 116
- Total Cost attribute 116
- TransactionType parameter 616, 832

- Transistor Part type
 - described 202
 - in Part inheritance model 29, 31
- translating
 - commands 488
 - menus 488
 - schema names 447
- tree categories. *See* navigation trees.
- tree categories. *See* navigation trees.
- Tree menu 129
- tree mode 752
- tree properties 480
- Tree Scope ID setting 547, 816
- treeLabel parameter 558, 833
- treeMenu parameter 558, 559, 833
- trees. *See* navigation trees.
- trigger
 - adding 422
 - common triggers 765
 - modifying 425
 - packages 770
 - quotations 771
 - RFQ 771
- trigger event, adding trigger for 422
- Trigger Manager. *See* emxTriggerManager program
- trigger programs
 - arguments for 84
 - Check Relative State 761
 - description of arguments 84
 - do not change code 421
 - do not change file names 421
 - installed with applications 421
 - modifying argument values for 425
 - previous revision promotion 758
 - relative float action 763
 - required connection check 758
 - required file check 760
 - sequence 85
 - Set Originator attribute 764
 - utility programs described 757
 - valid revision change 763
- TXT format 124
- Type Chooser 732
- type icon for form field 654
- type inheritance
 - Part 29, 31
 - shown with data models 27
- type macro 490
- types
 - attributes for 173
 - described 173

ownership 27
shown on data models 27

U

UDA
 type 202
UDA Mandatory attribute 116
UDA Miscellaneous relationship 164
UDA Name attribute 116
UDA Non-Recurring Cost relationship 164
UI level 3. *See* Dynamic User Interface.
UIForm.ValidationFile property 482
UIHeader.LeftImage 480
UIHeader.RightImage 480
UIStructureTree.ShowUponOpen 480
Unit Conversion type 202
Unit of Measure attribute 116
unit of measure conversion 720
Unit Price attribute 116
Unit Price Calculation Formula attribute 116
Unit Price Formula attribute 116
Units attribute 116
Unmanaged Document Grantor person 131
unregister objects 389
UOM Expression setting 816
Update Function setting 816
Update Program setting 817
upgrades 380, 381
Upper Limit attribute 116
Upper Specification Limit attribute 116
URL
 property 480
 type 202
URL parameters 818
Usage attribute 117
Use Case Estimated Duration attribute 117
Use Case Specification type 202
Use Case type 202
Use Case Validation relationship 164
User Agent person 131
User Manual type 202
User Requirement
 relationship 164
User Requirement Classification attribute 117
User Requirement Importance attribute 117
User Requirement type 202
User Symbolic Name attribute 117
user. *See* person.
utility trigger programs 757

V

Validate setting 817
Validate Type setting 817
validating form field data 668
 using custom JavaScript 669
 using custom JavaScript method 482
 using standard JavaScript validation 668
Validation Comments attribute 117
Validation Effort attribute 117
Validation Status attribute 117
Value (Real) attribute 117
Value (String) attribute 117
Value attribute 117
Value Type attribute 117
vault
 all described 204
 criteria 377
 primary 378
 safety 84
 secondary 378
Vault Access attribute 118
Vaulted Documents relationship
 described 165
 renamed 62
Vaulted Documents Rev2 relationship
 described 164
 used in expression access 71
Vaulted Objects relationship 165
 used in expression access 71
Vaulted Objects Rev2 relationship 164
vendor
 hardware 88
 software 111
version
 attribute 118
 property 381
 regression attribute 103
Version Date attribute 118
Version Document
 type 203
version log 133
Version relationship 165
 used in expression access 71
View mode for form 627
Viewable Base Filename attribute 118
Viewable Base Format attribute 118
Viewable relationship 165
Viewable State attribute 118
Viewable Subcomponent relationship 165
Viewable type 203
viewer

- assigning formats 430
- data model 57
- how viewers are registered 427
- integrating with 427
- program for getting viewers for format 132
- registering 428
- tip 428
- user preference 432, 433
- ViewerExtensions 431, 466
- viewerTip property 428
- Viewing Tool attribute 118

X

- XML format 124

W

- Wall Thickness attribute 118
- Warning Description attribute 118
- Washer Part type
 - described 203
 - in Part inheritance model 29, 31
- Web Availability attribute 118
- web form. *See* form.
- Web site
 - accessing from 750
- Web Site attribute 118
- Weight attribute 118
- Weight Limitation attribute 118
- Where Used Component Reference attribute 118
- Width attribute 119
- Window Height setting 817
- Window Width setting 817
- wizards
 - Change Administration Property 389
 - Change State Property 396
- Work Around attribute 119
- Work Phone Number attribute 119
- Work Place relationship 165
- Workspace
 - type 203
- Workspace Access Grantor person 131
- Workspace Buyer Desk relationship 165
- Workspace Lead Grantor person 131
 - used in expression access 67
- Workspace Member Grantor person 131
- Workspace Template Member relationship 165
- Workspace Template relationship 165
- Workspace Template type 203
- Workspace Vault
 - type 203
- Workspace Vault grants 469
- Workspace Vaults relationship
 - described 141
 - renamed 62

