Administration Guide | PUBLIC
2019-09-20

Administration Guide
SAP Quotation and Underwriting for Insurance 1.1

Content

1 Introduction

1.1 About this Document

This document provides procedures for administrative activities for SAP Quotation and Underwriting for Insurance, including how to use the Administrative Console and how to develop, package, and deploy applications.

1.2 Audience

The intended audience for this document is solution architects, application integrators, and IT administrators. It's assumed that you have had at least an introduction to the fundamentals of FS-PRO through classroom training.

The content of this document assumes that you have a working knowledge of the SAP HANA XS Advanced application platform.

1.3 Substitution Variable Placeholders

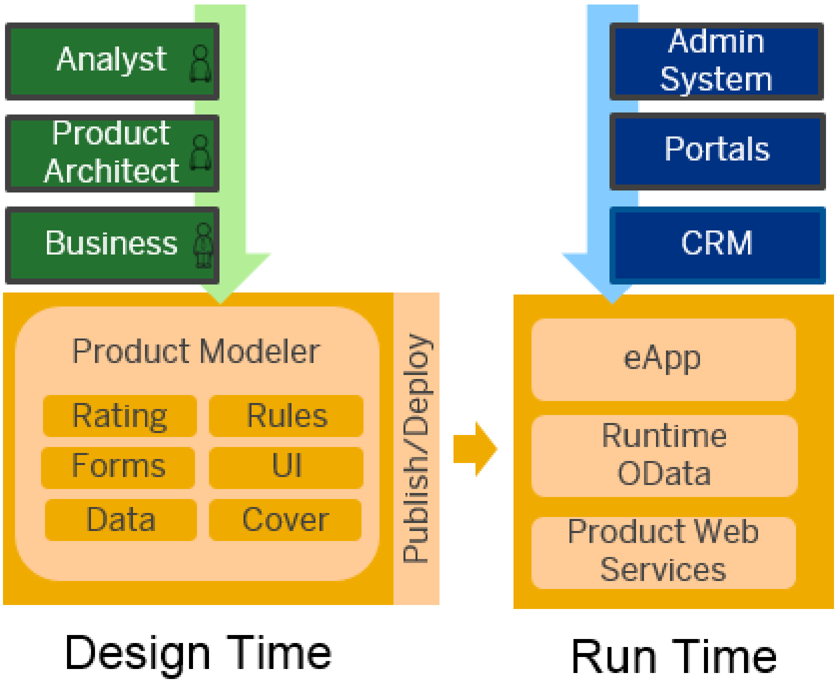
The following table defines the substitution variables used in this guide. Gather the information listed prior to beginning.

Substitution Variable	Description
`<pro_admin_user>`	The name of the administrative user for FS-PRO and FS-QUO.
`<pro_designtime_app_url>`	The URL to the FS-PRO Design Time approuter application.
`<quo_runtime_app_url>`	The URL to the FS-QUO Runtime approuter application.

1.4 Servers in the FS-PRO Suite

The FS-PRO suite consists of several application servers, each of which connects with databases and other application servers in particular ways.

The following figure provides a deployment view of the FS-PRO suite and shows how the Product Modeler server interacts with the other servers within the entire FS-PRO suite.



The Product Modeler Application

Products are created in the Product Modeler. A product contains coverages and components that store limits, rating formulas, and other rules. A product can be created from the ground up or from other products through the use of inheritance. After development and testing, product managers publish products to the FS-PRO Runtime Server and to the Product Repository in Product Services. The product is then available for use by applications developed in other enterprise applications.

The Product Modeler performs the following tasks:

- Supports the creation of a common, centralized repository of product definitions.
- Enables creation of template products and allows products to automatically inherit their attributes.
- Allows business analysts to add data elements to product definitions.
- Enables the creation of product rules and calculations.
- Updates the local cache that resides on each Product Services server, enhancing performance by storing a reduced copy of the product repository.
- Promotes a product, through the publishing process, to Product Services, making the product available to external systems.

The Product Modeler has its own database, and can publish to all the other servers in the suite.

The FS-QUO Server

FS-QUO provides an enterprise underwriting solution for PnC carriers.

FS-QUO performs the following tasks:

- Provides pre-built underwriting processes based on industry best practices.
- Supports policy lifecycle from quote to post-issue.
- Manages product definitions and business rules externally for speed and flexibility.
- Includes Underwriter's Work Area for full underwriting functionality.
- Provides pre-built management reports that can be easily extended or customized.

The FS-QUO server is a single application server that contains two functions:

- Design Time
- Runtime

They share a single database called the FS-QUO database. The FS-QUO server can be a standalone application server. However, typically, the Product Modeler publishes products to FS-QUO through internal application server communication.

The FS-QUO server is optional. If it's set up in your FS-PRO system, you can use it to design your own customized underwriter tools. Otherwise, you can use existing underwriter tools from a third-party system

The Product Services Server

In the FS-PRO suite, Product Services is the application server that hosts eApps and runtime OData services. eApps are smart web forms with questionnaires, validation, show/hide and calculation rules. OData is an industry and SAP standard HTTP-based ReST API for accessing data.

The figure above shows how the Product Services server interacts with the other servers within the entire FS-PRO suite.

Product Services performs the following tasks via SOAP APIs:

- Allows external application systems to reference product definitions and invoke product rule and rating execution.
- Manages the "packaging" of product rule and/or rating execution back to the external application system.

1.5 Client Requirements for FS-PRO

The FS-PRO client lets you access the Product Modeler and the Administrative Console. The client is browser-based.

For information about supported browsers, Client System Requirements.

1.6 Understanding the Role and Responsibilities of the Administrator

Each server has its own Administrative Console. As the administrator, you set up and configure the servers and their connections through each Administrative Console.

The FS-PRO administrator is responsible for installing, configuring, and maintaining the environment. Generally, this person is a senior developer or systems administrator, someone who has the necessary permissions to set up and connect servers and databases, and to grant network resource privileges.

1.7 Getting Started in the Administrative Console

1.7.1 Logging in to the Administrative Console

The first step in using the Administrative Console is to log in to the web application.

Prerequisites

You will need a user name and password. These are issued by the FS-PRO system administrator.

If you require a password change, contact your system administrator.

Context

To log in, perform the following steps:

Procedure

1. Open Internet Explorer and enter one of the following URLs in the web browser address field:
   - `<pro_designtime_app_url>`/csiroot/admin
   - `<quo_runtime_app_url>`/csiroot/admin

   > → Remember
   >
   > The Design Time and Runtime applications each have their own Administrative Console. Ensure that you are using the URL that connects to the appropriate Administrative Console for the tasks that you want to perform.
2. Press `Enter`.
   The *Logon* screen appears.
3. Enter the user name provided to you by your system administrator in the *Name* field.
4. Enter the password provided to you by your system administrator in the *Password* field.

   This field is case-sensitive.
5. Choose *Login*.
   The Administrative Console appears.

Results

You are logged in to the Administrative Console and ready to start work. Note that the features available to you are determined by permissions assigned to your userid. If you don't see a feature or option, consult your system administrator about changing your permissions.

1.7.2 Which Administrative Console am I Logged Into?

How do I figure out if I am logged in to the Design Time or the Runtime Administrative Console

Context

The Design Time and Runtime applications each have their own Administrative Console. Ensure that you are using the URL that connects to the appropriate Administrative Console for the tasks that you want to perform.

If you are working with more than one Administrative Console at a time, you can easily verify that you are accessing the Administrative Console that you intend.

To identify the Administrative Console that you are using,

Procedure

1. Launch the Administrative Console.
2. Look in the lower left corner of the browser; the URL of the current application appears in the browser frame.

Related Information

Changing an Association Between a Product and a Remote Server [page 79]
Adding an Association Between a Product and a Remote Server [page 80]
Viewing Information about your System [page 65]
Working with System and Application Files [page 65]
Enabling Log Files [page 69]
Viewing Log Files [page 70]
Searching for Objects [page 71]
Working with the Application Cache [page 72]
Working with Function Libraries [page 76]
Synchronizing the Database to all Flowstores [page 82]
Synchronizing the Database to Flowstores for a Product [page 83]
Resetting the Class Loader [page 77]
Testing a Connection Before Publishing [page 78]
Manually Deploying a Product [page 84]
Viewing Reference Metadata [page 94]
Creating a New Configuration Entry for a Server [page 102]
Creating a New Configuration Entry for an Application [page 103]
Renaming a Configuration Entry for a Server [page 104]
Renaming a Configuration Entry for an Application [page 104]
Deleting a Configuration Entry [page 105]
Creating a Configuration Variable for a Server [page 106]
Creating a Configuration Variable for an Application [page 106]
Editing a Configuration Variable [page 107]
Overriding Configuration Variables [page 108]

2 Application User and Authorization Management

Authorization and authentication for FS-QUO users is managed using the mechanisms provided by your platform application. For more information about the authorization and authentication tools and policies, see the SAP HANA XS Advanced environment documentation.

Users for FS-QUO will be created on the Identity Provider (IdP) level. Roles, role collections and user assignments are maintained by the security management cockpit provided by SAP HANA XS Advanced.

In FS-QUO, the scopes (permissions) have been predefined by the assigned application and is presented in a role template. The role templates can be grouped into a role collection based on different permission requirements. Users can be granted a proper permission set by assigning them to one or more role collections.

Note that there are three FS-PRO features that have additional user management requirements:

Product OData Services	You can create a set of Fiori applications that run on top of the FS-PRO Product Modeler. These apps provide a more business-friendly and customizable way to perform the functions available in the Product Modeler. They communicate with FS-PRO through an OData layer which can also be customized. User management for the Product OData Services requires a hybrid of the role collection assignment through user management and XML-based permission for more granular access on the service layer.
Operations APIs	Several administrative tasks are exposed as RESTful APIs allowing you to perform these tasks without a multi-step manual process. User management for these Operations APIs requires a hybrid of the role collection assignment through user management and XML-based permission for more granular access on the service layer..
Product Web Services	Product Web Services are used to access product content and run business rules through traditional SOAP web service on the FS-PRO Runtime instance. This feature doesn't use the role collection information from your platform application. Rather, user management is handled solely by the FS-PRO *Role Painter*.

2.1 Authorizations

Authorization for users in FS-QUO is managed via roles and role collections defined in your platform application.

The following functions need to be performed outside of FS-QUO:

- User management (creating, deleting, and modifying users)
- Application role collection management (creating, modifying, and deleting role collections)

- Assigning role-collections to users

The SAP HANA XS Advanced Administration and Monitoring Tools is used for authentication and authorization. For more information on managing user authorizations in SAP HANA XS Advanced, see the Authorization Management Tools topic in the *SAP HANA Security Guide*.

2.1.1 Resource URL Authorization

Authorization for users to access specific URLs which are related to specific functionalities is managed by the platform authorization tools.

FS-QUO and FS-PRO are shipped with a set of standard Role Templates and associated scopes, described in the table below. These Role Templates can be assigned to Role Collections, and Role Collections assigned to Users and Groups, using the platform authorization management tools.

Authorizations are checked on every HTTP request. The table lists the URL patterns that are authorized by each Role Template.

After authorization checks on the HTTP request URL, additional authorization checks on specific actions are performed in the business service layer inside the application server.

> ⓘ Note
>
> If you create your own role collections, remember to add it to the user. You will also need to assign the appropriate authorizations for the FS-IPW apps. For example, you will need to assign UW_User_RC to a user in order for them view submissions in the Underwriting Worklist. If you will be creating Underwriting Groups, then the groups created in the UW groups component will need a role collection defined.

Insufficient scopes for a user will result in their browser displaying a `403 Forbidden` error.

> ⓘ Note
>
> URL patterns that begin with `/A2A/` are intended for application-to-application usage, and have basic authentication. The URL patterns `/csiroot/productCatalog/` and `/csiroot/assembledComponentValueRow/` are also intended for A2A use and have basic authentication. All other URL patterns have UAA authentication, which includes OAuth, SAML, X.509 and form-based authentication.

> ⓘ Note
>
> All authenticated users are authorized to access the URL pattern `/sap/`, which is the URL for the Fiori Launch pad. ABAP authorization checks are then applied to determine which tiles are visible to the user.

2.1.1.1 FS-QUO and FS-IPW OData API Role Templates

Role Template Name: UW_User_RT

Scope & Permissions Granted	AccessQuo AccessEApp AccessUW uwSubmissionListQuery uwCaseMassReassign uwSimulate UWFullAccess AccessCustomServlet assignSubmission
Typical Usage	Access to apps
Applicable Instance	Runtime

URL Patterns	`/csiroot/eapp-runtime/`
	`/A2A/csiroot/ipw/quo/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/quoteSummary/odata.svc/`
	`/A2A/csiroot/ipw/uw/odata.svc/`
	`/A2A/csiroot/ipw/eApp/odata.svc/`
	`/csiroot/ipw/`

Role Template Name: DPP_Officer_RT

Scope & Permissions Granted	ManageDataPrivacy AccessEApp AccessQuo AccessUW AccessCustomServlet
Typical Usage	Access to DPP-related apps.
Applicable Instance	Runtime

URL Patterns	`/csiroot/dpp/`
	`/csiroot/eapp-runtime/`
	`/A2A/csiroot/ipw/quo/odata.svc/`

	`/A2A/csiroot/ipw/quo/web/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/quoteSummary/odata.svc/`
	`/A2A/csiroot/ipw/uw/odata.svc/`
	`/A2A/csiroot/ipw/eApp/odata.svc/`
	`/csiroot/ipw/`

Role Template Name: QUO_User_RT

Scope & Permissions Granted	AccessQuo AccessEApp AccessCustomServlet
Typical Usage	Access to apps, but no underwriting access.
Applicable Instance	Runtime

URL Patterns	`/csiroot/eapp-runtime/`
	`/A2A/csiroot/ipw/quo/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/quoteSummary/odata.svc/`
	`/A2A/csiroot/ipw/uw/odata.svc/`
	`/A2A/csiroot/ipw/eApp/odata.svc/`
	`/csiroot/ipw/`

2.1.1.2 Administration Role Templates

Role Template Name: Administrator_RT

Scope & Permissions Granted	ManageSystemConfigure	Manage system configuration
	ManageWebService	Manage Product Web Services (deploy Product Web Services, upload product)
	ManageGeneratedRuntimeContent	Manage generated Runtime content (synchronize flowstores, load cache)
	ADMINISTRATOR	Full access to all Administrative Console functionality
Typical Usage	Application super administrators	

Applicable Instance	Design Time Runtime

URL Patterns	`/csiroot/admin*`
	`/csiroot/psruntime/ws/`
	`/csiroot/ii/pc/jsp/enu/componentchooser/`

Role Template Name: Content_Admin_RT

Scope & Permissions Granted	ManageGeneratedRuntimeContent	Synchronized the deployed the product content to the Runtime application
Typical Usage	Product content administrators	
Applicable Instance	Design Time Runtime	
URL Patterns	`/csiroot/admin*` `/csiroot/ii/pc/jsp/enu/componentchooser/`	

Role Template Name: Webservice_Admin_RT

Scope & Permissions Granted	ManageWebService	Manage Product Web Services, upload and deploy product JAR files to SOAP web services
Typical Usage	Product Web Services administrators	
Applicable Instance	Runtime	
URL Patterns	`/csiroot/admin*` `/csiroot/psruntime/ws/`	

Role Template Name: System_Admin_RT

Scope & Permissions Granted	ManageSystemConfigure	Manage FS-QUO Design Time or Runtime system configuration
Typical Usage	System configuration administrators	
Applicable Instance	Design Time Runtime	

URL Patterns	`/csiroot/admin*`

Role Template Name: Product_Deployer_RT

Scope & Permissions Granted	ViewProduct	View, but not modify, product content
	DeployProduct	Remotely deploy products from Design Time to Runtime applications
	PRODUCT_DEPLOY_ROLE	Can access product deploy to Runtime transaction
Typical Usage	Users that need permission to deploy products to the Runtime application	
Applicable Instance	Runtime	

URL Patterns	`/csiroot/ii/pc/*`
	`/csiroot/pa/pc/`
	`/csiroot/pcportal/`
	`/csiroot/service/`
	`/csiroot/servlet/(.*)-service`
	`/csiroot/productCatalog/`
	`/csiroot/assembledComponentValueRow/`	

2.1.1.3 eApp Role Templates

Role Template Name: EApp_User_RT

Scope & Permissions Granted	AccessEApp	Access to the eApp runtime application
	AccessCustomServlet	Access to the custom servlet
Typical Usage	eApp users	
Applicable Instance	Runtime	

URL Patterns	`/A2A/csiroot/ipw/quo/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/odata.svc/`
	`/A2A/csiroot/ipw/quo/web/quoteSummary/odata.svc/`
	`/A2A/csiroot/ipw/uw/odata.svc/`
	`/A2A/csiroot/ipw/eApp/odata.svc/`
	`/csiroot/ipw/`	

2.1.1.4 Product Modeling Role Templates

Role Template Name: Product_Modeler_Admin_RT

Scope & Permissions Granted	ViewProduct	View, but not modify, product content
	ModifyProduct	Modify product content
	PublishProduct	Publish products
	DeployProduct	Deploy products
	ModifyProductTemplate	Modify product content (including product templates)
	SE_ADMIN_ROLE	Viewing objects, viewing and changing permissions other than Execute and reassigning the ownership of an object
Typical Usage	Product modelers	
Applicable Instance	Design Time	

URL Patterns	`/csiroot/ii/pc/`
	`/csiroot/pa/pc/`
	`/csiroot/pcportal/`
	`/csiroot/servlet/(.*)-service`
	`/csiroot/service/`
	`/csiroot/productCatalog/`
	`/csiroot/assembledComponentValueRow/`	

Role Template Name: Product_Modeler_Author_RT

Scope & Permissions Granted	ViewProduct	View, but not modify, product content
	ModifyProduct	Modify product content (not including product templates)
	PublishProduct	Publish products
	DeployProduct	Deploy products
	SE_AUTHOR_ROLE	Creating and maintaining objects
Typical Usage	Product modelers managing only marketable products (not product templates)	

> **Note**
>
> Users with this permission can access the product repository but not the System repository

Applicable Instance	Design Time

URL Patterns	`/csiroot/ii/pc/`
	`/csiroot/pa/pc/`
	`/csiroot/pcportal/`
	`/csiroot/servlet/(.*)-service`
	`/csiroot/service/`
	`/csiroot/productCatalog/`
	`/csiroot/assembledComponentValueRow/`

Role Template Name: Product_Viewer_RT

Scope & Permissions Granted	ViewProduct	View, but not modify, product content
Typical Usage	Read-only access to product configuration	
Applicable Instance	Design Time	

URL Patterns	`/csiroot/ii/pc/`
	`/csiroot/pa/pc/`
	`/csiroot/pcportal/`
	`/csiroot/servlet/(.*)-service`
	`/csiroot/productCatalog/`
	`/csiroot/assembledComponentValueRow/`	

Role Template Name: Product_Custom_View_Creator_RT

Scope & Permissions Granted	ViewProduct	View, but not modify, product content
	CUSTOM_VIEW_CREATOR_ROLE	Customize view to generate product report
Typical Usage	Product managers	
Applicable Instance	Design Time	

URL Patterns	`/csiroot/ii/pc/`
	`/csiroot/pa/pc/`
	`/csiroot/pcportal/`	

```
/csiroot/servlet/(.*)-service
/csiroot/productCatalog/
/csiroot/assembledComponentValueRow/
```

2.1.1.5 Operations APIs Role Templates

Role Template Name: Ops_User_RT

Scope & Permissions Granted	UpdateCSIHome	Update CSI_HOME files
	ImportObject	Import objects
	ExportObject	Export objects
	BuildObject	Build objects
	SyncObject	Synchronize objects
	PublishProduct	Publish products
	DeployProduct	Deploy products
	OPS_API_EXECUTION_ROLE	Access to all Operations APIs
	SE_ADMIN_ROLE	In FS-PRO, this role permits the following actions: • viewing objects • viewing and changing permissions other than Execute • re-assigning the ownership of an object
Typical Usage	Technical clients calling Ops API	
Applicable Instance	Design Time Runtime	

URL Patterns	`/csiroot/ii/pc/`
	`/csiroot/pa/pc/`
	`/csiroot/pcportal/`
	`/csiroot/servlet/(.*)-service`
	`/csiroot/service/`
	`/csiroot/productCatalog/`	

2.1.1.6 FS-PRO OData API Role Templates

These scopes are relevant for Technical clients calling FS-PRO OData services. All Role Templates in this section have permissions specified in `permissions_config.xml` in CSIHome.

Role Template Name: Product_OData_Deploy_RT

Scope & Permissions Granted	PRODUCT_ODATA_DEPLOY_ROLE	FS-PRO OData service: deploy activity-related transactions
Typical Usage	Managing scheduled activities Managing scheduled activities	
Applicable Instance	Design Time	
URL Patterns	`/csiroot/productCatalog/` `/csiroot/assembledComponentValueRow/`	

Role Template Name: Product_OData_ReadWrite_RT

Scope & Permissions Granted	PRODUCT_ODATA_READONLY_ROLE	FS-PRO OData service: read-only access to product or component data value rows
	PRODUCT_ODATA_READWRITE_ROLE	FS-PRO OData service: product or component data value row read and write actions
Typical Usage	Ability to create, update and delete data value rows Read-only access	
Applicable Instance	Design Time	
URL Patterns	`/csiroot/productCatalog/` `/csiroot/assembledComponentValueRow/`	

Role Template Name: Product_OData_ReadOnly_RT

Scope & Permissions Granted	PRODUCT_ODATA_READONLY_ROLE	FS-PRO OData service: Object-level read-only access rights for specified objects (individual products or components) that are defined

		in the Product Modeler. It includes `Read` permission. It will allow a user to read a data value row by conjunction with Product OData permissions. It should work in conjunction with other non-object level Product_OData_xxx permissions.

Typical Usage	Read-only access to product lists, data rows, and scheduled activities. Allows data searches
Applicable Instance	Design Time
URL Patterns	`/csiroot/productCatalog/` `/csiroot/assembledComponentValueRow/`

Role Template Name: Product_OData_Object_FullAccess_RT

Scope & Permissions Granted	PRODUCT_OBJECT_LEVEL_FULLACCESS_ROLE	FS-PRO OData service: Object-level full access rights for specified objects (individual products or components) that are defined in the Product Modeler. It includes `Read`, `Write` and `Full Control` which has permission to grant rights to other FS-PRO roles through the Product modeler UI. It will allow a user to modify a data value row by conjunction with Product OData permissions. It should work in conjunction with other non-object level Product_OData_xxx permissions.

Typical Usage	Bypasses the object-level permission defined in the Product Modeler permission settings. Allows full permission (read, write, full) to the object (products, components, etc).
Applicable Instance	Design Time
URL Patterns	Not applicable

Role Template Name: Product_OData_Object_WriteAccess_RT

Scope & Permissions Granted	PRODUCT_OBJECT_LEVEL_WRITEACCESS_ROLE	FS-PRO OData service: Object-level write and read access rights for specified objects (individual products or components) that are defined in the Product Modeler. It includes `Write` and `Read` permission. It will allow a user to modify a data value row by conjunction with Product OData permissions. It should work in conjunction with other non-object level Product_OData_xxx permissions.
Typical Usage	Bypasses the object-level permission defined in the Product Modeler permission settings. Allows write permission to objects (products, components, etc)	
Applicable Instance	Design Time	
URL Patterns	Not applicable	

Role Template Name: Product_Runtime_OData_RT

Scope & Permissions Granted	ExecuteProductRuntimeOData	FS-PRO Runtime OData service: execution
Typical Usage	Executes the Runtime OData services.	
Applicable Instance	Runtime	
URL Patterns	`/csiroot/productRuntime/` `/A2A/csiroot/productRuntime/`	

2.1.1.7 Product Web Services Runtime Role Templates

Role Template Name: Product_Web_Service_Runtime_RT

Scope & Permissions Granted	AccessProductWebServiceRuntime
Typical Usage	Access to Product Web Services in the Runtime application.

Applicable Instance	Runtime
URL Patterns	Not applicable

2.1.2 Data Visibility

User groups and role collections can also be used to control data visibility of submissions and tasks in the Quotation and Task worklists.

In the current implementation of data visibility, submissions and tasks are visible to their creator, as well as creator's peers. That is, anyone who is part of any of the user groups that the creator belongs to will see creator's submissions and tasks.

Example: Example user configuration:

There are two teams reporting to a manager. The agents within each team should see each other's submissions and tasks but should not see submissions and tasks of the other team. A manager of the two teams wants to see everyone's submissions. To achieve this data visibility the following user setup should be implemented:

	Role Collection	Users
Team A	TEAM_A_RC	Uw_A1, Uw_A2, Uw_A3,...
Team B	TEAM_B_RC	Uw_B1, Uw_B2, Uw_B3,...
Manager	TEAM_A_RC, TEAM_B_RC	Uw_Mgr1

> ⓘ Note
>
> Be aware that if there is a Role Collection which is assigned to all users then everyone will see everyone's submissions and tasks.

For more information about the current implementation of data visibility, see Understanding the Data Visibility Framework and Data Visibility Configuration.

2.1.3 Role Collections and Sample Products

While the FS-QUO sample products have been configured with authorizations assigned, additional role collections will need to be assigned for increased data visibility and feature use in the FS-IPW apps.

The following role collections need to be assigned to users:

Product	Required for Underwriting Worklist	Required for Reassign
Sample Household	QUO_UW_HO_LOB_PNC	SAP_UW_HOUSEHOLD
Sample Household Extended	QUO_UW_HO_LOB_PNC	SAP_UW_HOUSEHOLD
Life	UW_User_RC	SAP_UW_LIFE
Motor Vehicle	UW_User_RC	QUO_UW_HO_LOB_PNC

3 Product OData Services Development User and Authorization Management

You can create a set of Fiori applications that run on top of the FS-PRO Product Modeler. These apps provide a more business-friendly and customizable way to perform the functions available in the Product Modeler. They communicate with FS-PRO through an OData layer which can also be customized.

User management for the Product OData Services requires a hybrid of the role collection assignment through user management and XML-based permission for more granular access on the service layer.

You grant service layer permissions for roles in FS-PRO for the Product OData Services by editing a sample XML file which is installed with FS-PRO.

The file is named `permission_config.xml.template` and is included in the installation package. You can download this file from the application administrative console and customize it. Then save a copy as `permission_config.xml` with your own customized configuration or keep the default settings for OData permissions verification.

After the modification is complete, the `permission_config.xml` file needs to be uploaded back to same location as the `permission_config.xml.template` (as the system will only read the permission configuration from `permission_config.xml`, not the template file).

> **Note**
>
> No permissions will be granted without the `permission_config.xml` file.

For more information about the Product OData Services, see the Design Time Services Guide and the Runtime Services for Insurance Apps.

3.1 Defining Permissions for Product OData Services

This topic describes how to assign permissions to users and roles for the Product OData Services.

You grant service layer permissions for roles in FS-PRO for the Product OData Services by editing a sample XML file which is installed with FS-PRO.

To add, modify or remove permissions, you will need to download the file, edit it and then upload it to make the permissions available to FS-PRO and Product OData Services. You will have to upload the file again after every update.

The file allows you to assign permissions using the following methods:

Assign permissions by role	Access to a Product OData Service can be granted by role. Any user that has been assigned that role will be have the defined permission to the API. The roles must have already been defined in UAA before attempting to include them in the XML file.

Assign permissions to an individual user	Access to a Product OData Service can be granted to an individual user. This method allows different users to have different permissions. You can only assign additional permissions for a user, you can't reduce permissions that have already been defined for a user in UAA
Assign permissions to all users	You can define blanket access to an individual Product OData Service, making it available to all users. This is an efficient method to use if you want all users to have the same permissions.
Assign permissions by inheritance	When you designate a parent permission within the file, any child permissions included in the tag node will automatically inherit the permissions of the parent. This can reduce the time and effort required to maintain the permissions file, by reducing the file size and the number of nodes needed.

If a permission check has been configured in FS-PRO (for example, in a Product), it can be bypassed using the following permission options in the XML file:

Write*	The object level permission check will be omitted if the required object-level permission required for the check is Write and if the user has this permission.
FullAccess*	The object level permission check will be omitted if the object-level permission required for the check is FullAccess or Write and if the user has this permission.

3.1.1 Defining Role Permissions for Product OData Services

This topic describes the steps required to assign permissions to Product OData Services by role.

Prerequisites

Only users with sufficient permission to access the Design Time Administrative Console can define or modify permissions for Product OData Services.

The application roles (scopes) must have already been defined in UAA.

Context

Granting permissions for roles in FS-PRO for Product OData Services is accomplished by editing an XML file. A sample XML file called `permission_config.xml.template` is provided out-of-the-box to help you define the permissions. To add, modify or remove permissions, you will need to download the file, edit it and then upload it to make the permissions available to FS-PRO and Product OData Services. Note that the file is static; it is not dynamically connected to FS-PRO. You will have to upload the file again after every update.

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. Go to *System* > *Edit File System*.
3. Choose PA - Custom Integration Folder (Write) from the *Path* dropdown list.
4. Select the `permission` folder in the tree
5. Choose the `permission_config.xml.template` file from the resulting dialog.
   The file is downloaded.
6. Rename the file to be `permission_config.xml`.
7. Open the XML file in an editor.
8. Make your updates.

   > → Tip
   >
   > Adding comments to each node set will aid in keeping the XML file organized and easy to use.
9. Save your changes.
10. Upload the updated XML file to Design Time Administrative Console.
    a. Go to *System* > *Edit File System*.
    b. Choose PA - Custom Integration Folder (Write) from the *Path* dropdown list.
    c. Select the `permission` folder in the tree
    d. Right-click in the right pane and choose *Upload*.
    e. Browse and select the updated XML file.
    f. Choose *Upload*.

3.1.2 Defining Parent-Child Role Permissions for Product OData Services

This topic describes the steps required to define inherited permissions to Product OData Services.

Prerequisites

Only users with sufficient permission to access the Design Time Administrative Console can define or modify permissions for Product OData Services.

The application roles (scopes) must have already been defined in UAA.

Context

When you designate a parent permission within the file, any child permissions included in the tag node will automatically inherit the permissions of the parent. This can reduce the time and effort required to maintain the permissions file, by reducing the file size and the number of nodes needed.

Granting parent-child role permissions in FS-PRO for Product OData Services is accomplished by editing an XML file. A sample XML file called `permission_config.xml.template` is provided out-of-the-box to help you define the permissions. To add, modify or remove permissions, you will need to download the file, edit it and then upload it to make the permissions available to FS-PRO and Product OData Services. You will have to upload the file again after every update.

> ⚠ Caution
>
> Do not alter or edit the value of the application role or permission code in the XML file.

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. Go to *System* › *Edit File System*.
3. Choose PA - Custom Integration Folder (Write) from the *Path* dropdown list.
4. Select the `permission` folder in the tree
5. Choose the `permission_config.xml.template` file from the resulting dialog.
   The file is downloaded.
6. Rename the file to be `permission_config.xml`.
7. Open the XML in an editor.
8. Create a node in the file for each parent that you want to define.

   > → Tip
   >
   > Adding comments to each node set will aid in keeping the XML file organized and easy to use.

   a. Add a tag node to the file:

   ```
   <PermissionSets>
       <ParentPermission></ParentPermission>
       <ChildPermission></ChildPermission>
       <ChildPermission></ChildPermission>
       <ChildPermission></ChildPermission>
   </PermissionSets>
   ```

   b. Create as many `<ChildPermission>` tags as needed in the node.

> Note
>
> There can only be one `<ParentPermission>` tag defined in each `<PermissionSets>` node set. Custom `<ParentPermission>` sets can be created to regroup permission codes in a different group and assign to a specific role.

c. Enter the name of the service that you want to define as the parent in the `<ParentPermission>` tag.

   Any permissions that have been assigned to this service will automatically be inherited by any defined children.
d. Enter the name of the service that you want to define as the child in each `<ChildPermission>` tag.
e. Save your changes.

9. Upload the updated XML file to Design Time Administrative Console.
   a. Go to *System* > *Edit File System*.
   b. Choose PA - Custom Integration Folder (Write) from the *Path* dropdown list.
   c. Select the `permission` folder in the tree
   d. Right-click in the right pane and choose *Upload*.
   e. Browse and select the updated XML file.
   f. Choose *Upload*.

4 Operations APIs User and Authorization Management

Several administrative tasks are exposed as RESTful APIs allowing you to perform these tasks without going through multiple manual steps in the Administrative Console.

User management for these Operations APIs requires a hybrid of the role collection assignment through user management and XML-based permission for more granular access on the service layer.

You grant service layer permissions for roles in FS-PRO for the Operations APIs by editing a sample XML file which is installed with FS-PRO.

The file is named `permission_config.xml.template` and is included in the installation package. You can download this file from the application administrative console and customize it. Then save a copy as `permission_config.xml` with your own customized configuration or keep the default settings for Operations APIs permissions verification.

After the modification is complete, the `permission_config.xml` file needs to be uploaded back to same location as the `permission_config.xml.template` (as the system will only read the permission configuration from `permission_config.xml`, not the template file).

> ⓘ Note
>
> No permissions will be granted without the `permission_config.xml` file.

For more information about the Operations APIs, see Operations APIs.

4.1 Assigning Permissions in the Administrative Console

Out-of-the-box user called admin with the `ADMINISTRATOR` role is created in the Administrative Console's *Role Painter*. In addition, out-of-the-box, a permission XML is defined under the path: *User* > *Edit Permissions*. This permission XML gives `ADMINISTRATOR` role permission to access all areas in the Administrative Console. You are able to build on this xml to grant or deny access for different roles to different areas in the Administrative Console.

To modify permissions, you export the XML file, makes changes to it externally, and then import it back. When you upload the file, the system validates if the uploading XML is in a valid format.

- If the validation passes, the XML file is uploaded and the current XML file is updated and refreshed to the uploaded XML
- If the validation fails, the uploaded XML is rejected and the current XML file won't be updated

> ⓘ Note
>
> Changes to `permission_config.xml` will take effect only after a restart of the FS-PRO server.

According to permissions defined in the permission XML, Administrative Console menu items are granted the following permissions:

Deny
: No access

  Menu item is disabled.

Read
: Read access

  Menu item is accessible, but all editable fields, buttons, icons, links are disabled (except for search, refresh and test functions).

Update
: Update access

  Menu item is accessible and all fields are enabled for create, delete, update actions, but not execute action.

Execute
: Execution access

  Menu item is accessible and all fields are enabled for execute actions, but not create, delete, update actions.

> ⓘ Note
>
> You can specify a default permission (with the DefaultPermission tag) for those menu items not explicitly specified in the XML.

Administrative Console Guidelines for Roles and Permissions

You should define your own admin roles to grant or deny access for different functions depending on the requirements of your company.

The following guidelines apply when defining permissions in the XML:

- If a function is only applicable for read access, setting it to update access permission in XML has no effect – read access permission will be applied.
- If a role is granted no access permission to all areas, then any user belonging to the role shouldn't be able to login to Administrative Console.
- If a user belongs to multiple roles, and the roles have conflicting permissions, then will be granted the higher permission (Deny < Read < Update/Execute).
  For example, User belongs to RoleA and RoleB. RoleA has update permission to functionF. RoleB has no permission to functionF. User has update permission to functionF
- The roles specified in the XML must already exist in UAA.

4.1.1 Understanding the Permissions XML File

To define the permissions for your users, you need to assign them in an XML file.

Changes to the XML file will only take effect after a restart of the FS-PRO server.

The following table lists the tagging structure of the XML file:

Node Name		Description	Comments
`<PermissionConfig>`		Master node	Only one per XML file All other nodes must be child nodes of this
`<AllUsersPermissions>`		Default permissions for all users	Child of PermissionConfig
	`<Permission>`	Name of permission	Child of AllUsersPermission
`<PermissionSets>`		Parent node for permission set	Child of PermissionConfig
	`<ParentPermission>`	Name of permission set	Child of PermissionSets Only one per PermissionSets
	`<ChildPermission>`	Name of permission	Child of PermissionSets One or more per PermissionSets
`<RolePermissions>`		Parent node for role	Child of PermissionConfig
	`<Role>`	Name of role	Child of RolePermissions Only one per RolePermissions
	`<Permission>`	Name of permission	Child of RolePermissions One or more per RolePermissions
`<UserPermissions>`		Parent node for user permissions	Child of PermissionConfig
	`<UserLoginName>`	Login name (ID) of user	Child of UserPermissions Only one per UserPermissions
	`<Permission>`	Name of permission	Child of UserPermissions One or more per UserPermissions

There are four different permissions for menus and menu items:

- deny
- read

- update
- execute

It also allows you to specify a default permission for a certain role. For example, permission to menus and menu items for the ADMINISTRATIVE role as follows:

Out-of-the-box permission XML which grants permission to the ADMINISTRATIVE role to all areas in the Administrative Console

```
<xml version="1.0"?>
<Roles>
<Role ID="ADMINISTRATOR"> <!—For the ADMINISTRATOR-->
<DefaultPermission>
<Permissions&gt;UPDATE, EXECUTE&lt;/Permissions>
<!--Grant update and execute access for all unspecified menu items-->
</DefaultPermission>
</Role>
</Roles>
```

Out-of-the-box permission XML with ADMINISTRATOR access to all areas; Also assigns only the Deploy Product function and access to *Help* menu to the ProductDeploymentRole role

```
<?xml version="1.0"?>
<Roles>
<Role ID="ADMINISTRATOR"> <!--For ADMINISTRATOR-->
<DefaultPermission>
<Permissions>UPDATE, EXECUTE</Permissions>
<!--Grant update and execute access for all unspecified menu items-->
</DefaultPermission>
</Role>
<Role ID="ProductDeploymentRole"><!--For ProductDeploymentRole-->
<Object TYPE="MENU">
<Name>Deploy_Products</Name> <!--For the Deploy Product menu item under Product
Services-->
<Permissions>UPDATE, EXECUTE</Permissions> <!--Allows to update and execute-->
</Object>
<Object TYPE="MENU">
<Name>HELP</Name> <!--For all items under Help menu-->
<Permissions>READ</Permissions> <!--Allows to read-->
</Object>
<DefaultPermission>
<Permissions>DENY</Permissions> <!--No access to all unspecified menu items-->
</DefaultPermission>
</Role>
</Roles>
```

Description of the XML Tags

The list below provides a description of the XML tags

Role	Attribute: ID Name of the role to which permission will be assigned. Multiple Role tags can be used to define permission for many roles.
DefaultPermission	Element: Permissions Used to specify the default permission when not explicitly done.

	Default permission tag is optional, when not specified the default permission is DENY. EXECUTE — Menus and menu items not explicitly defined will be granted execution access. UPDATE — Menus and menu items not explicitly defined will be granted update access. READ — Menus and menu items not explicitly defined will be granted read access. DENY — Menus and menu items not explicitly defined will be denied access.
Object	Attribute: TYPE Type of the object we're defining permissions for. In the permission XML, it is always MENU. The TYPE attribute is optional.
Object	Element: Name Menu and menu item name to which permission will be assigned.
Object	Element: Permissions Permission to be granted. Content of Permissions tag is a comma separated list of permission names. EXECUTE — Execution access UPDATE — Write access READ — Read access DENY — No access

XML IDs

The table below provides list of Administrative Console functions for the Product Modeler and their names to be used with XML:

Function	Name in XML
System	SYSTEM
Edit Configuration Settings	Edit_Configuration_Settings
Edit File System	Edit_File_System
Enable Logs	Enable_Logs
Execute SQL Queries	Execute_SQL_Queries
Register Function Library	Register_Function_Library
Reset ClassLoader	Reset_ClassLoader

Function	Name in XML
Reload Configuration Settings	Reload_Configuration_Settings
View Logs	View_Logs
Product	PRODUCT
Clear Attributes Cache	Clear_Attributes_Cache
Delete Products	Delete_Products
Deploy Products	Deploy_Products
Distribute Products	Distribute_Products
Manage Product Web Services	Manage_Product_Web_Services
Search Objects	Search_Objects
Synchronize Product Flowstore	Synchronize_Product_Flowstore
View Primary Key Generator Cache	View_Primary_Key_Generator_Cache
View Product Cache	View_Product_Cache
View Reference Metadata	View_Reference_Metadata
Process	PROCESS
Clear Application Cache	Clear_Application_Cache
Synchronize Process Flowstore	Synchronize_Process_Flowstore
User	USER
Edit Permissions	Edit_Permissions
Edit Roles	Edit_Roles
Edit Users	Edit_Users
Help	HELP
Help	About

Permissions for a Menu as Opposed to Menu Items

The following guidelines apply:

- If a role has no access to a menu, and individual menu item access isn't specified, then the menu won't be displayed for a user belonging to the role.

- If a role is defined with access permission to a menu, and individual menu item access isn't specified, then all items under the menu have the same access permission.
- Permission to a menu item takes precedence over permission for its menu.
    - For example, if a role is given update access to a menu but read access to an item on the same menu, then the role has read access to the menu item, and update access to the rest of the menu items
    - For example, if a role is given read access to a menu but update access to an item on the same menu, then the role has update access to the menu item, and read access to the rest of the menu items

5 Product Web Services User and Authorization Management

Product Web Services are used for local development on the FS-PRO Runtime instance. User management and authorization for the services is defined via the *Role Painter* feature.

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

In the *Role Painter*, you define organizational hierarchies, define roles, and assign users to the appropriate organization level and role.

A master administrator with user name admin and password of password is created out-of-the box and has the ADMINISTRATOR role. The admin user has access to the *System* menu, *User* menu, and *Help* menu in order to setup the environment.

All users involved in business processes must be defined in Directory in order to have a user ID and a password. When implementing FS-PRO, your system administrator must define the organizational hierarchy, define roles and job titles, and assign users to the appropriate job title and authorization profile. These tasks are performed using *Role Painter*.

The organizational hierarchy consists of the following components:

Organization
: The organization and its main branch offices.

Organization Unit
: One of the major units within the organization—for example, a division, a department, or a business unit.

Job Title
: One of the positions within an organization unit. Because *Role Painter* requires each job title to specify the position that it reports to, a complete set of job titles describes the reporting structure within an organization unit.

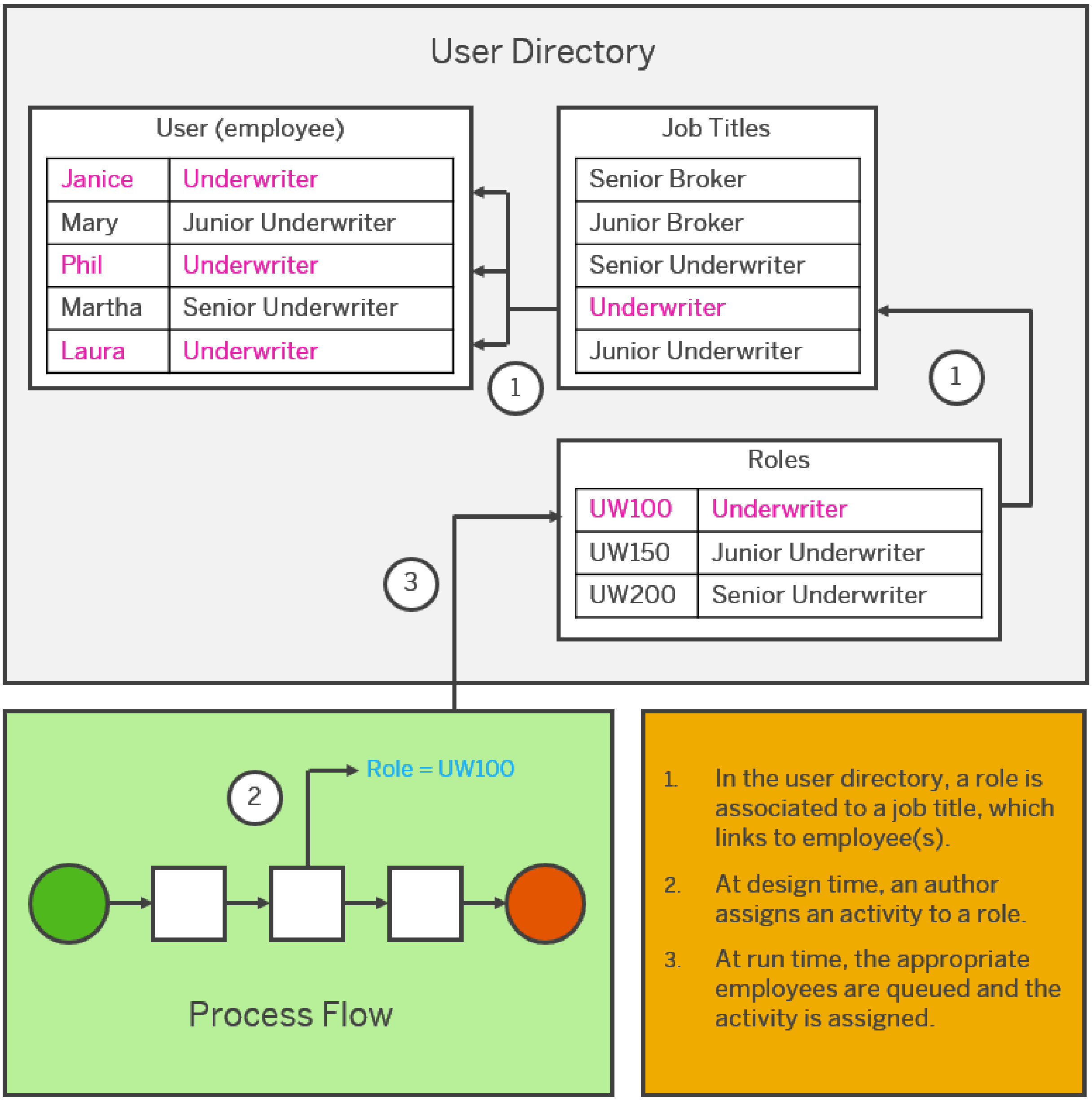
> ⓘ Note
>
> A job title must be unique within its organization unit.

User
: The information pertaining to an actual user, including contact details, access privileges, and authorization profile.

Role
: Every role is associated with a job title. And because every job title is associated with one-or-more users, as a workflow progresses the appropriate users are notified of their pending tasks.

> ⓘ Note
>
> If a user's job title changes, some or all of the user's roles may change as well.

The following figure illustrates how Directory and the Role data element link a process flow to people in an organization.



The *Role Painter* screen consists of several panels.

The hierarchy of the component panels goes from the *Organization* panel through the *Organization Units* and *Job Titles* to individual *Users* and their *Roles*. You can collapse and expand the component panels by selecting the *X* in the upper left corner of a panel.

The detail panel at the bottom of the screen contains information about the element that is currently selected in the upper panel.

For more information about the Product Web Services, see Working with Product Web Services.

Authorization Profiles

An authority profile is a custom object that defines a user's access rights and ability to view and modify data.

Authorization profiles allow you to control what an end user can do within the system at Runtime. In an authorization profile, you define custom attributes and values, which rules can process. For example, you can use an authorization profile to dynamically control the values that appear to different users who access a dropdown list.

A system administrator creates profiles using Authorization Services.

> ⓘ Note
>
> Authorization Profiles are no longer supported. Similar functionality is provided in scopes and role collections.

Related Information

Resource URL Authorization [page 14]

5.1 Accessing *Role Painter*

Users will also need to be defined for the Product Web Services used for local development on the FS-PRO Runtime instance. This is accomplished via the *Role Painter* feature.

Prerequisites

Only the system administrator can access *Role Painter*.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Log in to the Administrative Console,
2. Go to *User* > *Edit Roles*.

Results

The *Role Painter* screen appears.

5.2 Managing Organizations

The following procedure describes how to create an organization and branch organizations, and how to delete them.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Organization is the highest level of the organizational hierarchy. This level contains an entry for the head office and can include entries for other main offices. An organization can have several levels. You can use this feature to designate, for example, different geographical regions that are served by the organization.

Procedure

1. Log in to the Administrative Console,
2. Select *Edit Roles* from the menu.
   The *Role Painter* screen appears.
3. Determine the type of office that your are creating:
   - If you are creating a head office, right-click in the *Organization* section, and choose *New*.
   - If you are creating a branch office, right-click the higher level organization under which the branch office is to appear, and choose *New*.

   The Organization information fields appear in the detail panel.

4. Enter the name of the organization in the *Display Name* field.
5. Select the location with which the organization is associated from the *Location* dropdown list.
6. Save your changes.
7. You can delete only organizations that have no dependent entities, such as organization units.
    a. Right-click an item in the *Organization* panel to select the organization you want to delete and choose *Delete*.
    The organization information appears in the detail panel.
    b. Choose *Delete*.
    c. Confirm your decision.

5.3 Managing Organization Categories and Units

Organization units are the second level in the organizational hierarchy. These are specific divisions, departments, or business units within each organization that you have defined.

You can subdivide organization units; for example, you can create departments in business units.

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

5.3.1 Creating Organization Categories

This procedure describes how to add new organization categories.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

When creating an organization unit, you need to select an organization unit category. Typical categories would be Divisions or Business Units. If an appropriate category doesn't exist, you can create it. The categories appear in the detail panel and in the *Category* dropdown list.

Procedure

1. Right-click the organization unit, and choose *OrgCategory*.
   All the categories that have been defined appear in the detail panel.
2. Right-click in the *Categories* list and choose *New*.
3. Enter the organization unit label in the *Category Description* field.
4. Save your changes.

5.3.2 Deleting Organization Categories

This procedure describes how to delete an organization category.

Prerequisites

You can only delete categories that have no dependent organization units

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click an organization unit, and choose *OrgCategory*.
   All the categories that have been defined appear in the detail panel.
2. Right-click the category you want to delete, and choose *Delete*.
   The category information appears in the detail panel.
3. Choose *Delete*.
4. Confirm your decision.

5.3.3 Creating Organization Units

This procedure describes how to create organization units and child organization units.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Select the organization to which you want to add an organization unit in the *Organization* panel.
2. Right-click an organization unit in the *Organization Unit* panel and choose one of the following options:
    - Select *New* to add a new organization unit
    - Select *New Child* to add a child unit to the selected organization unit.

    The organization unit fields appear in the detail panel.
3. Enter the name of the organization unit in the *Display Name* field.
4. Select the category for the organization unit from the *Category*.
5. Select the job title of the person who heads the organization unit from the *Manager Job Title*.
6. Select the location of the organization unit from the *Location* .
7. Optionally, enter the cost center for the organization unit in the *Cost Center* field.
8. Save your changes.

5.3.4 Deleting Organization Units

This procedure describes how to delete an organization unit.

Prerequisites

You can delete only organization units that have no dependent entities, such as job titles.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

To delete an organization unit, perform the following steps:

Procedure

1. Select the unit you want to delete in the *Organization Unit* panel.
2. Right-click the unit and choose *Delete*.
3. Confirm your decision.

5.3.5 Moving Organization Units

You can move an organization unit to another organization or organization unit. In the latter case, the organization unit that is moved becomes a child of the target organization unit.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click the organization unit you want to move and choose *Move*.
   The organization unit move information appears in the detail panel.
2. Select the organization or organization unit into which you want to move the selected organization unit.
3. Choose *Move*.

5.4 Managing Locations

Locations are the physical sites of organizations or organization units. A location can include more than one organization or organization unit.

When you are creating an organization or organization unit, you need to select a location. If the correct location doesn't exist, you can create it. The *Locations* list appears in the detail panel.

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

5.4.1 Creating Locations

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click an organization or organization unit, and choose *List Location*.
   The currently defined locations appear in the detail panel.
2. Right-click a location, and choose *New*.
3. Enter the display name for the location in the *Location Name* field.
4. Enter the street address of the location in the *Address* field.
5. Enter the city of the location in the *City* field.
6. Select the country of the location from the *Country* dropdown list.
7. Select the state of the location from the *State* dropdown list.
8. Enter the postal or zip code of the location in the *Zip* field.
9. Save your changes.

5.4.2 Editing Locations

This procedure describes how to change the information for a location.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Select the location you want to change in the *Locations* list.
2. Choose *Edit Location*.
3. Edit the information.
4. Save your changes.

5.4.3 Deleting Locations

This procedure describes how to delete a location.

Prerequisites

You can delete a location only if it isn't referenced by an organization or organization unit.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click an organization or organization unit, and choose *List Locations*.
2. Select the location you want to delete in the *Locations* list
3. Right-click the location.
4. Choose *Delete*.
5. Confirm your decision.

5.5 Working with Job Titles

In Directory, a job title is a component of an organization unit. A job title must be unique within its organization unit, and must specify the positions that it reports to. After you have defined your organization, organization units, and locations, you need to identify all the job titles in each of the business units that you have defined.

When you define job titles, you are defining the reporting structure of the organization, because each job title reports to another job title. Therefore you begin defining job titles at the highest executive level of the organization and move down through the other levels.

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

5.5.1 Creating Job Titles

This procedure describes how to create a job title for an organization unit.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Select the organization in the *Organization* panel.
   The organization units belonging to the organization appear in the *Organization Units* panel.
2. Select the organization unit to which you want to add a job title.
3. In the Job Title panel, do one of the following:
   - Choose *Add New Jobtitle*.
   - If there are already job titles in the panel, right-click one, and choose *New*.
4. Enter a unique name for the job title in the *Display Name* field.
5. Select the job title to which this job title reports from the *Reports To* dropdown list.
6. Save your changes.

5.5.2 Editing Job Titles

This procedure describes how to update a job title.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

When you change a job title, the change applies to all users associated with that job title.

Procedure

1. Select the job title in the *Job Title* panel.
2. Right-click the job title and choose *Edit*.
3. Edit the information.
4. Save your changes.

5.5.3 Moving Job Titles

This procedure descibes how to move job titles to other organizations or organization units.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click the job title you want to move and choose *Move*.
2. Select an organization or unit in the detail panel,
   This is the organization or unit to which you want to move the selected job title.
3. Choose *Move*.

5.5.4 Deleting Job Titles

This procedure describes how to delete a job title.

Prerequisites

You can delete a job title only if there are no users or roles dependent on it.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click on a job title in the *Job Title* panel.
2. Choose *Delete*.
   The information about this job title appears in the detail panel.
3. Choose *Delete*.
4. Confirm your decision.

5.6 Users and Roles

After you identify the job titles used in all the units of the organization, you can associate users with job titles. The following sections describe how to associate existing users with job titles, create new users, remove users, and define user security.

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

To organize roles, you can create role categories. For example, an Administrator role category could contain roles such as Web Administrator and Applications Administrator. Roles can be shared among role categories. For example, Policy Administrator and Sales role categories can both contain the Insurance Applicant Reviewer role.

Roles are associated with job titles. Consequently, if a user's job title changes, assumes the roles associated with the new job title.

> → Remember
>
> The users and roles created in *Role Painter* are for use by Product Web Services on the Runtime instance only. They are not the same as the application roles and users created in SAP HANA XS Advanced.

5.6.1 Adding Existing Users to Job Titles

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

> ⚠ Caution
>
> When you add users to a job title, keep in mind that all the users with that job title must have the same roles. If some users with a particular job title have different roles, create another job title for them so that they can be assigned to their appropriate roles.

Procedure

1. Select the organization in the *Organization* panel.
   The units belonging to the organization appear in the *Organization Units* panel.
2. Select the organization unit that contains the job title to which you want to add a user.
3. Select the job title.
4. Right-click in the *Users* tab and choose *Add Existing User To Roles*.
   A list of available users appears in the detail panel.
5. Select the user.
6. Choose *View* to view the user's primary job title and then choose *Back* to return to the *Available Users* list.
7. Select Secondary from the *Job Type* dropdown list if this is the user's secondary job title. If this is the user's primary job title, accept the default value.

   A user may have one primary job title, and many secondary job titles.
8. Save your changes.

5.6.2 Adding New Users to Job Titles

After you identify the job titles used in all the units of the organization, you can associate users with job titles.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Select an organization in the *Organization* panel.

   This is the organization for which you want to add a new user to a job title.

   The units belonging to the organization appear in the *Organization Unit* panel.
2. Select the organization unit that contains the job title.
3. Select the job title in the *Job Title* panel.
4. Right-click in the *Users* tab and choose *Create New User*.
5. Complete the fields. Most are self-explanatory, other than these few:

Inactive	Indicates whether is currently inactive (inactive users can't be selected at runtime).
Job Type	Select Primary if this job title reflects the user's main function, otherwise select Secondary. A user may have one Primary job title and many Secondary job titles.
Authorization Profile	Enter the name of Authorization Services profile to which belongs. The system doesn't validate this field. If the profile name doesn't match the one in the UAS, at runtime the authorization procedure fails. ⓘ Note Authorization Profiles are no longer supported. Similar functionality is provided in scopes and role collections.

   > ⓘ Note
   >
   > When you create a user, the user receives a default password of `changeme`, and is prompted to change it when they log in for the first time.

6. Save your changes.

Next Steps

You can now configure the user's security settings and login preferences.

Related Information

Resource URL Authorization [page 14]

5.6.3 Removing Users from Job Titles

You can remove a user from a job title.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

The *Remove User From Role* option only removes the user from the role and doesn't delete the user from the system.

Procedure

1. Select the organization in the *Organization* panel.
   The units belonging to the organization appear in the *Organization Units* panel.
2. Select the organization unit that contains the job title to which you want to remove a user from a role.
3. Select the job title in the *Job Title* panel.
4. Select a user in the *Users* panel.
5. Right-click and choose *Remove User From Roles*.
6. Choose *Remove*.

   A confirmation message appears.
7. Choose *OK*.

5.6.4 Creating Role Categories

Role categories enable you to group similar roles.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Perform one of the following actions in the *Users* panel:
    - Right-click a user in the *Users* tab and choose *Role Category*.
    - Right-click a role in the *Roles* tab and choose *Role Category*.

    The *Role Category* list appears in the detail panel.
2. Right-click a category, and choose *New*.
3. Enter a unique name for the new category in the *Role Description* field.
4. Save your changes.

5.6.5 Creating Roles

This procedure describes how to add a role.

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

In process flows, roles link activities to the users who perform them. At Runtime, all the people who are assigned to the role are notified in the Runtime instance that there is an activity for them to perform.

Procedure

1. Choose the *Roles* tab in the *Users* pane.
2. Right-click, and choose *New*.
3. Enter a brief description of the role in the *Description* field.
4. Select a role category (if appropriate) from the *Category* list.
5. Save your changes.

Results

The role is added to the *Roles* tab.

5.6.6 Deleting Roles

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

Procedure

1. Right-click the role you want to delete, and choose *Delete*.
   The information about this role appears in the detail panel.
2. Choose *Delete*.
3. Confirm your decision.

5.6.7 Assigning Users to Roles

Context

> ⚠ Caution
>
> This section is provided solely for FS-QUO toolkit customizations based on legacy code, and for use in local development (non-SAP HANA XSA) runtimes. *Role Painter* authorizations, organizations, job titles, and profiles are no longer used by FS-QUO on SAP HANA XSA.

All the users listed on the *Users* tab for a particular job title have the same role or roles. If a user has a different role, you need to create another job title, so that you can assign that user to the appropriate roles.

Procedure

1. Click the *Roles* tab in the *Users* panel.
2. If the role already exists in the system, perform the following steps:
   a. Right-click a role and choose *Add Role*.
   b. From the *Available Roles* list, select the role you want to assign to the user.
   c. Choose *Add*.
3. If the role doesn't exist in the system, perform the following steps:
   a. Right-click and choose *New Role*.
   b. Enter the role for these users in the *Description* field, .
   c. Select the category to which the role belongs from the *Category* dropdown list.
      If the desired category doesn't exist, you can create one.
   d. Save your changes.

Results

The new role is added to the *Roles* tab. All the users listed in the *Users* tab now have this role.

5.7 Editing User Settings

Administrators can update passwords and clear sessions for for users.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

If a Product Web Services user forgets their password, administrator users can view it in the *Edit Users* screen. The passwords can also be reset when necessary.

> → Remember
>
> Passwords for SAP HANA XS Advanced cannot be changed from within the FS-PRO Administrative Console. They must be changed in the XS Advanced Administration and Monitoring tools.

Administrator users can also clear the Product Modeler sessions associated with a user. This can be necessary, for example, if your system uses the `UniqueSignOn` feature, which specifies whether more than one person can sign on simultaneously to a server using the same user name. If a certain user is still logged in, and a different user needs to log in at the same time, you may need to clear the sessions of the first user.

To reset a user password or clear a user session, perform the following steps:

Procedure

1. Go to *User* > *Edit Users*.
   The *Edit Users* screen appears.
2. Enter the first few letters of the user's name, and choose *Search*.
   A list appears.
3. Select the name in the list.
   Information about that user appears in the detail panel.
4. Enter a new password for this user, and choose the *Reset user's password* link.
5. If the screen indicates that this user has any open sessions, you can clear them by choosing *Clear Sessions* in the tree panel.

6 Working in the Administrative Console

You can use the to perform a wide variety of configuration and administrative tasks in FS-PRO.

To increase security and better keep track of user activities, it is now possible for a master administrator to create administrative roles, and associate individual Administrative Console function with roles. Direct access to individual functions is prohibited – you must first login to Administrative Console to access the functions. An Audit log is also available to keep track of user activities in the Administrative Console.

ⓘ Note

To perform any procedure related to the Administrative Console, you require the appropriate permissions.

Related Information

Configuring Servers [page 101]

6.1 Using the Menu Bar in the Administrative Console

The Administrative Console menu bar provides access to a wide variety of configuration and administrative tasks in . You require the appropriate permissions to access certain tasks.

The table below illustrates the Administrative Console menu bar for the Product Modeler. An x indicates a sub-menu is available and the options are shown separately for each sub-menu together with a description of the function of the item.

Login	System	Product	User	Help	Logout
	X	X	X	X	

The following sub-menus are accessed by selecting the title to display a sub-menu where applicable.

System Menu

The *System* menu is where you configure the data required to run the application

Sub-menu	Description
Edit Configuration Settings	Configure server and application settings.
Edit File System	Views and manages CSI_HOME files.
Enable Logs	Specifies content of log files.
Register Function Library	Views, adds, edits, or deletes function libraries.
Reset ClassLoader	Loads all classes defined in the class path to the application server memory.
Reload Configuration Settings	Apply changes in Config Manager.
View Logs	Views generated log files.
View Exceptions	View exception servlet.

Product Menu

The *Product* menu contains product-related functions.

Sub-menu	Description
Delete Products	Deletes product from single or multiple servers.
Deploy Products	Uploads a product JAR file to the current server.
Edit Product Deployments	Makes product available to external systems When you publish from the Product Modeler server.
Synchronize Product Flowstore	Converts product data definitions to database design for a particular product or a data definition model.
View Reference Metadata	Views reference products metadata.
Search Objects	Searches for rule and screen objects.
Manage Product Web Services	Uploads, downloads, deploys, and undeploys products in repository and Runtime.
Clear Attributes Cache	Clears attributes cache.
View Primary Key Generator Cache	Views primary Keys available for each database table.
View Product Cache	Adds, edits, and deletes product cache.

Sub-menu	Description
Clear Application Cache	Reloads, resets, clears or accesses application cache. A cache is a memory area where the system stores frequently-used data for rapid access. If you are receiving inconsistent results in FS-PRO, this may be caused by your cache containing out-of-date data. In this case, you should consider to first clear out the cache, and then to reload it.

Process Menu

The *Process* menu contains the following sub-menus.

Sub menu	Description
Clear Application Cache	Reloads, resets, clears or accesses application cache. A cache is a memory area where the system stores frequently-used data for rapid access. If you are receiving inconsistent results in FS-PRO, this may be caused by your cache containing out-of-date data. In this case, you should consider to first clear out the cache, and then to reload it.
Synchronize Process Flowstore	Compares the structure of your application to the database and synchronizes all flowstores.

User Menu

The *User* menu contains user account and user role functions. This menu is only applicable if you are using *Role Painter* to provide user management for Product Web Services. For all other features/functionality, user management is handled at the IdP level.

Sub-menu	Description
Edit Permissions	Defines permission to applications for user roles.
Edit Roles	Manages authentication and authorization, defines organizational hierarchies, roles, and assigns employees to the appropriate organization level and role.
Edit Users	Sets up users, changes passwords, and clears user sessions.

Help Menu

The *Help* menu contains the following sub-menu:

Sub-menu	Description
About	Views information about the system.

6.2 Tracking Activities in the Administrative Console Using a Log File

Activities in the Administrative Console are recorded in a log file. Access or read actions aren't recorded in the log file, but download actions are recorded.

When you makes changes in the Administrative Console, a record is created in the log file with the following information:

- Name of who made the changes
- Date and time of the changes
- Function that was changed, the old and new values, if applicable

Each log file record is in the following format: `[date and time of the change] [user name who made the change]: [function that is changed] - [user action] [message with detail of the change]`

6.2.1 Accessing the Log File

You can access the Administrative Console log file in one of two ways.

Procedure

1. Download it from the <CSI_HOME> environment log folder.
   For example, `\\herc07\was70_csihomes\csi\s13018\env\log\config_ changes.log`
2. Alternately, go to *System* > *View Logs* > *System* > *Log* > *System Logs* > *ConfigAdminLog* in the Administrative Console

6.2.2 Changing the Name of the Log File

You can change the name of the log file from the default name.

Procedure

1. Go to *Configuration* > *System* > *Log* > *System Logs*
2. Change the `ConfigAdminLog` config variable.

6.2.3 Disabling the Activity Log

You can disable the activity log in the Administrative Console.

Procedure

1. Go to *System* > *Edit Configuration Settings* > *Configuration* > *System* > *Log* > *System Logs* .
2. Change the `AdminAuditTrailMode` config variable value to False.

6.2.4 Archiving Log Files

The System will archive a log file larger than 10MB and appended with [#].log. For example, `config_changes_1.log` , `config_changes_2.log`

> **Note**
>
> You must back up and clean up log files to prevent them from taking up too much space.

6.3 Viewing Information about your System

You can view information about your system setup.

Procedure

1. Log in to the Administrative Console.
2. Verify that you are in the correct server.
3. Go to *Help* > *About*.

Results

A table containing information about your system appears.

6.4 Working with System and Application Files

You can perform tasks with files stored in the file system. You can view the file system from the *System* menu.

Procedure

1. Verify that you have logged into the correct server in the Administrative Console.
2. Go to *System* > *Edit File System*.

   A list of folders appears in the tree, and the folders and files in the selected folder appear in the detail panel.
3. If you want to view a different directory, select it from the *Path* dropdown list.

   If you have edit privileges for the directory, *(Write)* appears in the name.
4. Navigate in the tree to the folder containing the file you want to work with.

   A list of files appears in the detail panel.

6.4.1 Downloading Files

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *System* > *Edit File System*.
   A list of folders appears in the tree, and the folders and files in the selected folder appear in the detail panel.
2. Navigate in the tree to the folder containing the file you want to work with.
   A list of files appears in the detail panel.
3. Double-click the filename and choose *Save*.

   The *Save As* dialog displays.
4. Select the location where you want to store the downloaded file, and choose *Save*.

6.4.2 Uploading Files

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *System* > *Edit File System*.
   A list of folders appears in the tree, and the folders and files in the selected folder appear in the detail panel.
2. Navigate in the tree to the folder containing the file you want to work with.
   A list of files appears in the detail panel.
3. Right-click anywhere in the detail panel below the list of file names, and choose *Upload*.
   The *Upload Files* dialog appears.
4. For each file you want to upload, perform the following steps:
   a. Choose *Browse*.

The *Choose File* dialog appears.

b. Navigate to the folder containing the file you want to upload, select it, and choose *Open*.
   The selected file appears in the *Upload Files* dialog.
c. Repeat for any other files you want to upload. You can upload a maximum of eight files at one time.
d. Choose *Upload*.

Results

The uploaded files appear in the detail panel.

6.4.3 Creating Folders

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *System* > *Edit File System*.
   A list of folders appears in the tree, and the folders and files in the selected folder appear in the detail panel.
2. Navigate in the tree to the folder containing the file you want to work with.
   A list of files appears in the detail panel.
3. Right-click anywhere in the detail panel below the list of filenames, and choose *New Folder*.
   A dialog appears.
4. Enter a name for the new folder.

   The folder name can't contain spaces or other special characters such as slashes, asterisks, or colons.
5. Choose *OK*.

Results

The new folder appears in the list in the detail panel.

6.4.4 Deleting Files

You can delete system and application files that are stored in the file system.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *System* > *Edit File System*.
   A list of folders appears in the tree, and the folders and files in the selected folder appear in the detail panel.
2. Navigate in the tree to the folder containing the file you want to work with.
   A list of files appears in the detail panel.
3. Right-click the name of the file you want to delete, and choose *Delete*
4. Confirm your decision.

6.4.5 Downloading APIF Logs

If you are troubleshooting policy issuance issues, you can download the APIF log files to view the data sent from FS-QUO to FS-PM

Prerequisites

Ensure that the `FSPMBookingServiceEnabled` and `WriteFSPMPolicyDataToFile` configuration settings have been overridden to Yes.

Context

If your solution includes an integration with FS-PM, you can download the APIF log file.

> ⚠ Restriction
>
> This option is only applicable for the Runtime Administrative Console.

Procedure

1. Go to *System* > *Edit File System*.
2. From the *Path* dropdown, choose *AuthoritySuite-APIF XML Folder*.
3. Select the file from list on the right.
4. Right-click and select *Download*.

Related Information

Application > Product Authority > Env > Application Environment [page 138]

6.5 Working with Log Files

In FS-PRO, you can automatically record events in a log file, to provide an audit trail that can be used to diagnose problems.

6.5.1 Enabling Log Files

You can use log settings to specify the content of log files generated by the system. You can enable log files from the *System* menu. This topic describes how to set up your system so that it records log information.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

The interface consists of categories in the tree panel on the left side of the screen, and specific variables that you can enable in the detail panel on the right side of the screen.

Procedure

1. Go to *System* > *Enable Logs*.
2. Navigate the tree panel to locate the folder containing the log file you want to enable.

   For example, go to *Product Logs* > *Application Logs* > *Form Service Log* to enable the `Form Service` log file.
3. Choose Enabled for each variable in order to include its activities in the log.

   The list of variables depends on your application.
4. Save your changes.

6.5.2 Specifying a Path for Log File Storage

You can use this task to specify a path for log file storage for the msgRating, APIF and exception log files.

Procedure

1. Log in to the Runtime Administrative Console.
2. Choose *System* > *Edit Configuration Settings* from the menu bar.
3. Navigate to *Configuration* > *System* > *Log* > *System Logs* in the tree.

   The *Default* field of the `RuntimeLogPath` configuration variable displays the current location of the log file storage.
4. If you want to change this setting, enter a new path in the *Override* field.
5. Save your changes.
6. Choose the *Reload Config* icon to load the change into the system.

6.5.3 Viewing Log Files

A log file contains a list of activities in the system, depending on the variables selected.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *System* > *View Logs* .
2. In the tree panel, navigate to the log file that you want to view.

   For example, go to *View Logs* > *System* > *Log* > *System Logs* > *Form Service Log* to view the `Form Service` log file.

Results

The activities appear in the detail panel.

6.6 Searching for Objects

An Object Search tool is available in the Administrative Console to assist in locating the path or location of any rule or screen.

Prerequisites

Verify that you are logged in to the Design Time Administrative Console.

Procedure

1. Go to *Product* > *Search Objects* .
   The *Object Search Tool* displays.
2. Enter the rule/ID in the *Rule/Screen ID* field.

   For example, **`30560`** .
3. Select *Retrieve*.

   If you don't enter a Rule/Screen Id or enter an invalid Rule/Screen ID, a warning message will be displayed.

Results

The *Object Search Results* are displayed showing the *Root Name*, *Data Row Path* and *Usage Data Path*.

6.7 Working with the Application Cache

If you are receiving inconsistent results in FS-PRO, this may be caused by your cache containing out-of-date data. In this case, you should consider using the procedure below to first clear out the cache, and then to reload it.

Prerequisites

Verify that you are logged in to the Runtime Administrative Console.

Context

> ⓘ Note
>
> This feature is not supported. Do not use.

A cache is a memory area where the system stores frequently-used data for rapid access. There are two caches:

Common Cache	Contains the built-and-checked-in versions of all the objects in the system.
Cache	Contains all your checked-out objects.

The *Clear Application Cache* screen contains a list of object types, the number of each in memory, and the amount of memory each object type is using.

> ⚠ Caution
>
> If you are unsure about the consequences of modifying a specific cache, contact your system administrator or create an incident in the Product Support Portal.

Procedure

1. Select *Clear Application Cache* from the *Product* menu.
   The *Clear Application Cache* screen appears.
2. If you want to open and view a user cache, choose the *Cache* tab.
3. If you want to open the system's cache, choose the *Common Cache* tab.
4. If you want to select all the items in a tab, select the *Clear* checkbox.
5. If you want to see a report of the objects in a cache, in the *Available Objects in Memory* column, choose the number.
6. If you want to update your cache view to show the latest changes, choose *Refresh*.

7. If you want to remove all items from the cache, perform the following steps:
    a. Choose *Clear*.

       A confirmation message appears.
    b. Choose *Yes* to confirm your decision.
8. If you want to reload some or all of the items in the cache, choose *Reload*;

   The *Reload Cache* window opens.
    a. If you want to reload a particular process, choose its link in the *Process Name* list.
       A confirmation message appears.
    b. If you want to reload all processes, choose *Reload Cache (All)*.
9. When you are finished, choose *Close*.

6.8 Working with *View Product Cache*

The product cache uses the Apache Ehcache implemented in Runtime to cache products and reference products.

The product cache consists of four predefined caches. You can't rename or delete these caches; however, you can change their configuration parameters.

> ⚠ Caution
>
> If you are unsure about the consequences of modifying a specific cache, contact your system administrator or create an incident in the Product Support Portal.

The following predefined caches are available:

PROD_BAS	Stores the definitions for your products.
PROD_DEFAULT	Stores data for older products that don't have modifier codes.
REF_DATA_PROD	Stores the modifier codes for reference products.

There is an additional cache that you can add for coverages. The required name is COV and you can use it if you have coverages that you want to publish.

You can also add custom caches, but they will only work with custom classes that you provide.

6.8.1 Adding a Cache

You can add custom caches to work with custom classes that you provide.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to ▶ *Product* ▶ *View Product Cache* ◀.
2. Choose *Add Cache*.
   A blank row appears in the list of caches.
3. Enter the maximum number of objects that this cache can store in memory in the *Max Elements in Memory* field .
4. Assign a value to the *Eternal* option to determine whether elements in the cache expire.

   If this value is set to `true`, time-outs are ignored and the elements never expire.

   The default value is true.
5. Assign a value to the *Overflow to Disk* option.

   If this value is set to true, it allows elements beyond the number specified in the *Max Elements in Memory* field to be swapped out to disk.

   The default value is true.
6. Set the logic for removing elements from the cache by choosing a value from the *Memory Store Eviction Policy* dropdown list:

LRU	Least recently used. This is the default value.
LFU	Least frequently used.
FIFO	First in, first out. The oldest element determined by time of creation.

7. Save your changes.

6.8.2 Deleting a Cache

You can delete your custom product caches if they are no longer required.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

You should only delete the custom product caches that have been created by your organization. The predefined caches provided with the application should not be deleted. If you are unsure about the consequences of deleting a specific custom product cache, contact your system administrator.

Procedure

1. Go to *Product* › *View Product Cache*.
2. Select the custom product cache to be deleted.
3. Choose the *Delete Products* icon.

6.8.3 Viewing Statistics for the Product Cache

Procedure

1. Go to *Product* › *View Product Cache*.
2. Choose the *Cache Report* tab.

   Besides restating the *Max Elements in Memory* value, the tab includes the following information:

Elements in Memory	Indicates the number of elements currently in the cache's memory.
Number of Hits	Indicates the number of times a requested item was found in the cache.
Average getTime	Indicates how long it took on average to retrieve an item from the cache, in milliseconds. Because Ehcache supports JDK 1.4.2, each `getTime` uses `System.currentTimeMilis`, rather than nanoseconds, thus limiting the accuracy.
Eviction Count	Indicates the number of evictions from the cache, since the cache was created or statistics were last cleared.
Cache Misses	Indicates the number of times a requested item wasn't found in the cache. This value will always at least equal the number of elements in the cache unless you clear the cache.
Memory Used	Indicates the size of the memory store for the cache, in kilobytes.

   > **Note**
   >
   > The amount of memory used by a product can't be displayed because the Document object isn't serializable.

   > **Caution**
   >
   > The *Memory Used* field uses an "expensive" method to run and therefore could cause response problems, as the object lock is held for a long period; allow approximately one second per megabyte of entries.

3. If you want to update the statistics, choose *Refresh*.

4. If you want to see a list of the objects in a cache, choose the *View Cache* icon.
5. If you want to clear the statistics for a single cache, choose the *Clear Cache* icon; to clear the statistics for all the caches, choose *Clear All*.

6.8.4 Tuning the Product Cache

Procedure

1. Run production scenarios on a UAT or performance test bed to determine the cache growth and check for the point where it plateaus.
2. As products are added or as products are modified, their usage scenarios will need to be revisited. Run the above tests again to see if the cache size needs to grow.

6.9 Managing Function Libraries

Use the *Library Manager* to view, add, edit, or delete function libraries.

The *Library Manager* screen lists existing function libraries.

6.9.1 Working with Function Libraries

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Select *Register Function Library* from the *System* menu.
   The *Library Manager* screen appears. The top half of the screen shows a list of the existing function libraries.

2. You can use the validate option to determine whether a function library is properly registered. You would do this, for example, if you believe there is a problem with using the library's stem in *Rule Painter*. Validation ensures that information entered in a field is appropriate to that field or falls within a predefined range.
3. Identify the library that you want to validate and at the end of its row choose the *Validate* link.
   A validation message appears, indicating success or failure.
4. If you want to clear the function library cache, choose the icon beside the screen title. The cache is updated.
   Doing so refreshes the contents of the cache. You might want to do this if you believe that other users have made recent changes to the function library.
5. After you create a new function library, you use the *Library Manager* to add the function library to the server.
   a. To register a new function library on the server, complete the following fields:

Class name	The name of the Java class that contains the functions, including their logic and a list of arguments needed for each of them.
Library Name	Short description of the contents of the library.
Library Description	Long description of the contents of the library.
Stem Name	The rule stem that a user types in the Rule Painter expression field to access the functions of this library.
Stem Synonyms	Optionally, use this field to add any equivalents for the same. For example, if your stem was PRODUCT., you might add PROD. as a short form. Synonyms function exactly the same as the actual stem.

   b. To add the function library to the server environment, choose *Register*.

6.10 Resetting the Class Loader

FS-PRO contains a custom class loader used for loading all custom classes, including data integration interface classes and auto-process classes.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

The class loader can be used by both administrators and authors.

Procedure

Select *Reset ClassLoader* from the *System* menu.
A confirmation message appears in the main panel.

Related Information

The Resource Loader [page 162]

6.11 Working with Edit Product Deployments

When you publish from the Product Modeler server, you promote a product to the FS-PRO and Product Services application servers, which makes the product available to external systems. As well, the Product Modeler application server updates the local cache that resides on each application server. Each cache enhances performance by storing a reduced copy of the product repository.

The *Edit Product Deployments* screen consists of a list of products and information about each product.

6.11.1 Testing a Connection Before Publishing

Before you publish a product to a remote server, you should test the connection to that server.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *Product* > *Edit Product Deployments*.

   The *Edit Product Deployments* screen appears.
2. For any product, choose *Test Connection*.

   A confirmation message appears.
3. If the confirmation dialog indicates that there is a problem with the connection, make changes and test again.

4. Choose *Close*.

6.11.2 Changing an Association Between a Product and a Remote Server

The *Edit Product Deployments* screen displays the associations that currently exist between a product and the servers on which it runs. If necessary, you can change the details of an existing association.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *Product* > *Edit Product Deployments*.

   The *Edit Product Deployments* screen appears.
2. For any product, change the connection information in the *Server Name* or *Server URL* columns.
3. Test the connection.
4. Choose *Update*.

   A confirmation dialog appears.
5. Choose *OK*.

Related Information

Testing a Connection Before Publishing [page 78]

6.11.3 Adding an Association Between a Product and a Remote Server

When you publish a product in the Product Modeler, it is automatically published to the local server. You may want to also publish it to a remote server.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

To publish the product to a remote server, you need to set up an association between the product and one or more remote servers.

Procedure

1. Go to *Product* > *Edit Product Deployments*.

   The *Edit Product Deployments* screen appears.
2. Choose *Add New*.
   The *Add New Product to Runtime Server* dialog appears.
3. Choose for the product.
   The *Object Search* dialog appears.
4. Select the name of a product to view information about it.
5. Choose *Select* when you have found the correct product.
   Its name appears in the *Add New Product to Runtime Server* dialog.
6. Enter a name for the server in the *Server Name* field.
7. Enter the user ID in the *User Name* field.
8. Enter the password for the user in the *Password* field.
9. Save your changes.
   The new association appears in the list in the *Product Manager* screen.
10. Test the connection.
11. Choose *Update*.

    A confirmation dialog appears.
12. Choose *OK*.

Related Information

Testing a Connection Before Publishing [page 78]

6.12 Working with the Tree Loader Framework

6.12.1 Optimizing Build and Publish

This topic describes how to enable an alternative tree loader framework for the build and publish process.

Context

An alternative tree loader framework is available for use when processing products within the Product Modeler. It will allow for more efficient building of metadata and allow for a more efficient publishing process.

The following settings in the FS-PRO Administrative Console need to be applied to utilize the alternate frameworks.

> **Note**
>
> The original tree loader, building and publishing frameworks remain available, for backward compatibility.

Procedure

1. Go to *Configuration* > *PC* > *Env* > *PC Environment* in the FS-PRO Administrative Console.
2. Ensure that the `UseCompUsagePublisher` attribute is set to **`Yes`**.

   This attribute enables the use of alternate publishing logic that is compatible with the alternate tree loader, permitting the more efficient publishing of products.
3. Save your changes.
4. Choose the *Reload Config* icon to refresh the configuration settings.

6.13 Synchronizing Flowstores

A flowstore is a data definition that defines the relationship of the tables, fields and data attributes required by an application.

Flowstores can be sychronized in the following ways:

- Synchronizing the Database to all Flowstores
- Synchronizing the Database to Flowstores for a Product

6.13.1 Synchronizing the Database to all Flowstores

This process checks for differences between the application database and the data definitions in the application data model product, and optionally executes database operations to create database tables and columns to match those data definitions.

Prerequisites

You have defined your data definitions for your main tables in your application data model product.

Context

The flowstore definitions are derived from the application data model product during the publish of the application data model product. The flowstore XMLs are generated. You may find this procedure helpful when you need to re-synch any changes with the database, so that you can proceed to packaging and deployment tasks.

Procedure

1. Launch Internet Explorer and log in to the Runtime Administrative Console at the following location: `<pro_runtime_app_url>/csiroot/admin/`.
   The Runtime Administrative Console will open after a short delay.
2. Go to *Process* > *Synchronize Process Flowstore*.
   A log file appears. During this process, *Next* is dimmed.
3. When the log is complete, choose *Next*.
   A list of flowstores appears in the tree panel.
4. The flowstores that appear in red don't match the contents of the database. SQL statements are generated by the system to match the database to the flowstore. To view SQL statements related to those flowstores, select them. The SQL statements appear in the detail panel.
   - If the flowstore table is missing from the database, you will see statements that will create it.
   - If the table exists on the database, but the structure doesn't match that of the flowstore table, you will see SQL statements that will modify the database table accordingly.
5. To create the database tables, select each table name that is red and then choose *Apply*. Upon successful execution of the SQL statements, the table name will turn blue.

   A confirmation message appears.

> ⓘ Note
>
> When you choose *Apply*, you are applying all SQL statements that appear on the screen. To apply individual statements, copy them and execute them using a different SQL tool.

6. If you want to apply individual statements, copy them and execute them using a different SQL tool.

6.13.2 Synchronizing the Database to Flowstores for a Product

This process checks for differences between the application database and the data definitions in product extended data models, and optionally executes database operations to create database tables and columns to match those data definitions

Context

You may find this procedure useful if you have made changes to the product definition in the Product Modeler and you copied the product JAR to the application server.

Procedure

1. Launch Internet Explorer and log in to the Runtime Administrative Console at the following location: `<pro_runtime_app_url>/csiroot/admin/`.
   The Runtime Administrative Console will open after a short delay.
2. Go to *Product* > *Synchronize Product Flowstore* on the menubar.
3. Enter the name of the product or the extension data model in the *Product Name* field.
4. Enter the version number of the product or the extension data model in the *Product Version* field.
5. Choose *Next*.

   A log file appears. During this process, *Next* is dimmed.
6. When the log is complete, choose *Next*.
   A list of data definitions appears in the tree panel. The data definitions that appear in red don't match the contents of the database.
7. If you want to view SQL statements related to those data definitions, select them. The SQL statements appear in the detail panel
   - If the data definitions table is missing from the database, you will see statements that will create it.
   - If the table exists on the database, but the structure doesn't match that of the data definitions table, you will see SQL statements that will modify the database table accordingly.
8. If you want to run all SQL statements that appear on the screen, choose If you want to view SQL statements related to those data definitions, select them. The SQL statements appear in the detail panel.*Apply*.

A confirmation message appears.

9. If you want to apply individual statements, copy them and execute them using a different SQL tool.

6.14 Manually Deploying a Product

When you publish a product in the Product Modeler, the product is automatically published to the local server.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

You can also choose to deploy to a remote server. Behind the scenes, the JAR file on the local server is copied to a location on the remote server, and then loaded into the server's cache.

Use the following procedure if you want to deploy a product to a Runtime server so that you can test and use the product along with its process.

Procedure

1. Go to *Product* > *Deploy Products*.
2. Enter the path to the product JAR file, or choose *Browse* to find it.
3. Select a server from the dropdown list.

   This will be the server that the product will be deployed to.
4. Choose *Upload*.

   A confirmation message appears.
5. Confirm your decision.

6.15 *Product Web Services Manager*

To expose the Product Services rules as web services, you must specify the location of the *Product Web Services Manager* and enable deployment in the Design Time Administrative Console. The *Product Web Services Manager* is required to deploy Product Web Services.

> ⓘ Note
>
> You must also configure the `Custom Publishing` object in the Product Modeler. This object contains post-publishing services that upload the published JAR file to the Design Time *Product Web Services Manager* repository, deploy the JAR file to the *Product Web Services Manager* Runtime, and generate the WSDL.

Product Web Services Manager is launched from the Administrative Console, at *Product* > *Manage Product Web Services*.

> ⓘ Note
>
> Only the `System Administrator` user role can access the *Product Web Services Manager*.

Related Information

Configuring Servers [page 101]

6.15.1 Configuring the *Product Web Services Manager* in the Design Time Instance

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. Choose *System* > *Edit Configuration Settings* from the menu bar.
3. Go to *Configuration* > *PC* > *Env* > *PWS Manager Setting*.
4. In the *Override* field of the `ProductRepositoryUrl` configuration variable, enter the location of the *Product Web Services Manager* repository on the application server. Use the following format: `http://<server>:<port>/csiroot/psrepository`
5. In the *Override* field of the `ProductWSRuntimeUrl` configuration variable, enter the location of the *Product Web Services Manager* Runtime on the application server. Use the following format: `http://<server>:<port>/csiroot/psruntime`
6. Save your changes.
7. Choose *Configuration* > *PC* > *Env* > *PC Environment* in the configuration tree.
8. Change the *Override* field value of the `PrePostPublishingEnabled` configuration variable to Yes.

9. Save your changes.

6.15.2 Configuring the *Product Web Services Manager* in the Runtime Instance

To change the number of versions maintained for a product (the default value is 5), a retention setting must be configured.

Context

Configuration variables, `psruntime.default.user` and `psruntime.default.password` exist in Administrative Console at *System* > *Edit Configuration Settings* > *Configuration* > *PS* > *Env* > *PWS Manager Settings*. They are used for internal communication between the Product Services-repository and Product Services-Runtime. The values are setup out-of-the-box and you aren't required to modify them under normal circumstance. Any user and password pair from the SAP user repository can also be used instead of the default values.

Procedure

1. Go to *System* > *Edit Configuration Settings* > *Configuration* > *PS* > *Env* > *PWS Manager Settings* in the Runtime Administrative Console.
2. Change the value of *NumProductBuilds* to the desired value from the default of 5.
3. Save your changes.

   Next, adjust the parameters for web service request and response schema validation.
4. Go to *System* > *Edit Configuration Settings* > *Configuration* > *PS* > *Env* > *PWS Manager Settings* in the Runtime Administrative Console.
5. Change the value of *RequestValidation* to provide validation of request to schema.

   Select *No* to disable validation, or *Yes* to enable validation.
6. Change the value of *ResponseValidation* to provide validation of response to schema.

   Select *No* to disable validation, or *Yes* to enable validation.
7. Save your changes.
8. If you want to configure the *Log* tab to view the product web server performance statistics, perform the following steps:
   a. Go to *System* > *Edit Configuration Settings* > *Configuration* > *PS* > *Env* > *PWS Manager Settings* in the Runtime Administrative Console.
   b. Change the value of *ServiceAudit* to enable or disable logging.

      Select No to disable statistics, or Yes to enable statistics.

c. Save your changes.

Results

The *Product Web Services Manager* is configured.

6.15.3 Enabling Product Web Services Tracing

Product Web Services has a `ServiceTrace` option permitting Product Web Services rule tracing, which is useful for debugging.

Prerequisites

For performance reasons, Product Web Services tracing must be turned off in a production environment.

Context

When enabled, the web service response contains the tracing information at the bottom. Input data is marked with an "I" and output data is marked with an "O". If there is an error, the Java stack trace is shown.

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. Choose *System* > *Edit Configuration Settings* from the menu bar.
3. Go to *Configuration* > *PS* > *Env* > *PWS Manager Setting*.
4. Change the *ServiceTrace* entry to *Yes*.
5. Save your changes.
6. Reload the new setting.

6.15.4 Viewing Published Products and Product Details

You can create a listing of published products and product details.

Context

To view the details, perform the following steps:

Procedure

1. Go to *Product* > *Manage Product Web Services*.
2. Choose the *Repository* tab.

   The tab displays a listing of published products and product details, showing an archive of versions per product.

   > **Note**
   >
   > The system assigns a version number with each upload.

3. Search for a published product using the following search criteria:

Product Name	The name of the product.
Last Published By	The name of the user who published the product.
Last Published From/To	The date the product was published.

   The search results display the Product Name and Version of all published products matching the search criteria. The following also applies to search results:
   - By default, search results are prepopulated with products in the environment.
   - Only the latest version of each product displays.
   - The default number of products per page is 25.
4. Select a product from the search results in the left panel.
   The details for each published version of the product in the environment are displayed in the right panel:

Version	The version number of the published product
Published On	The publish date
Published By	The publisher
Comments	The publish comments

5. If you want to deploy a published product, right-click the product and choose *Deploy*.
   The product JAR file deploys to the Product Web Services Runtime environment, overriding the existing version (only one version is maintained), and generates and deploys the WSDL file to the Runtime environment.

6. If you want to download the product JAR file to the local computer, right-click on the product and choose *Download*.
7. You can manually upload a JAR file into the repository using *Upload JAR*.

   The dialog details for this include the following:

Product JAR	Browse to select a JAR file for upload. This is a required field.
Comments	Free form text area. This field is optional.

6.15.5 Testing Web Services

You can view and test the execution of web services in the Runtime environment using the *Product Web Services Manager*.

1. From the *Product Web Services Manager*, select the *Runtime* tab.
2. You can search for a deployed product using the following search criteria:

Product Name	The name of the deployed product.
Deployed By	The name of the user that deployed the product.
Deployed From/To	The date the product was deployed.

3. Search results display the *Product Name* and *Version* of all deployed products matching the search criteria. The following also applies to search results:
   - By default, search results are prepopulated with products in the environment.
   - The default number of products per page is 25.
   - Only one version of a product is stored in the Runtime environment.
4. Selecting a product displays the product details. The *Document* tab displays a WSDL report of all product services in detail including input, output, and the WSDL source code.
5. Right-click options for published products are as follows:

Undeploy	Removes the product JAR file from the Product Web Services Runtime environment.
Download	Downloads the product JAR file.

6. You can view the WSDL source code by selecting the *WSDL* link.
7. Product services can be validated and tested in the Runtime environment by selecting a Product Service from the *Operations* dropdown list and choosing *Test*. You can execute a test request and the system will display the appropriate response. A valid user name and password from the directory must be supplied in the request header for successful execution.
8. You can manually deploy a JAR file into the Runtime environment directly, skipping the Repository by choosing *Upload JAR* and browsing to select a JAR file to upload.
9. By default, the system performs schema validation against the web service request and response. This feature can be disabled.

Related Information

Configuring the Product Web Services Manager in the Runtime Instance [page 86]

6.15.6 Viewing Performance Statistics

Prerequisites

This feature must already have been be configured in order to perform this procedure.

Context

To view statistics in the *Product Web Services Manager* console, perform the following steps:

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. For the desired product web service, choose the *Log* tab.
3. Choose the *Refresh* icon at the bottom of the right panel to update the results.

Results

The *Log* tab displays the following web service details:

Product Service Name	The name of the product service.
Start Time	The time the product service was executed.
Success	Y for successful execution of the product service. N for unsuccessful execution of the product service.

6.15.7 Viewing Product Web Services Errors

Context

To view Product Web Services errors, perform the following steps:

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. Locate the desired product web service on the *Log* tab of the *Manage Product Web Services*.
3. Double-click the desired log entry to view additional details.

Results

The error log displays for unsuccessful transactions.

6.15.8 Product Web Services Access

In the Design Time Administrative Console, the Product Web Services function is also available so you can run ps-runtime from the repository of the same instance.

Procedure

1. Launch Internet Explorer and log in to the Design Time Administrative Console at the following location: `<pro_designtime_app_url>/csiroot/admin/`.

   The Design Time Administrative Console will open after a short delay.
2. Select *Product* > *Manage Product Web Services* from the menu bar .
   The display is refreshed.
3. Select the *Runtime* tab.
4. Select your product in the *Product Name* panel .
5. From the *Operations* dropdown list, select test.

6. Choose *Test*.

   The *Request Panel* is populated with XML.

   A security header is added to all outgoing requests and will have placeholders ('?') which you need to fill in.
7. Substitute the ? with user name and password information at both of the following lines:

```
<wsse:Username>?</wsse:Username>
<wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0#PasswordText">?</wsse:Password>
```

   > Note
   >
   > `Username` and `password` must already exist in directory.
8. Choose *Execute*.

Results

The *Response* pane is populated.

> Note
>
> If the user name and/or password doesn't match, the request will fail and an error message displays in the *Response* panel.

6.15.9 XML Output for Errors

In Product Web Services you are able to see an XML representation of an error caught by the system provided PWS has been configured.

The exceptions thrown when running a rule will be output into an XML format.

- The error code and error message information is only populated for the rule that caused the error exception.
- The rule that cause the exception is displayed first.
- Mapped from Error Object.
- The XML is the error output to the Product Web Services Runtime console.
- For backwards compatibility, the `<faultcode>` and `<faultstring>` tags will be populated with the `<userErrorCode>` and `<userErrorMessage>` respectively, if they exist. Otherwise, `<systemErrorCode>` and `<systemErrorMessage>` values are used, respectively.
- The rule Error Object created by the script rule will appear following the `<faultstring>` tag in the Product Web Services error response.
- The generated product WDSL in Product Web Services Runtime will include the definition of the error object.

6.15.10 Viewing Rule Error Messages in the Administrative Console

Context

To view rule error messages in the Administrative Console, perform the following steps:

Procedure

1. Log in to the Administrative Console.

   The Design Time Administrative Console will open after a short delay.
2. Go to *Product* > *Manage Product Web Services*.
   The display is refreshed.
3. Select the *Runtime* tab.
4. Select your product in the *Product Name* panel.
5. Select a rule name from the *Operations* dropdown list.

   For example, stateLookup.
6. Select *Test*.
7. The *Request* panel is populated with XML.
8. Enter your test parameter.

   In our example, at line 10, replace "string" with "FL" as an invalid state code to illustrate the error message.
9. Choose *Execute*.
10. The XML code for the "errorObject" is displayed with the `<userErrorCode>` and `<userErrorMessage>` as illustrated below:

```
<errorObject>
<ruleName>stateLookuo</ruleName>
<ruleId>85342</ruleId>
<lineNumber>3</lineNumber>
<productName>Script Rule Exception Handling</productName>
<productVersion>1</productVersion>
<xPath/>
<userErrorCode>USRERR_STATE_NOT_FOUND</userErrorCode>
<userErrorMessage>STATE=&quot;FL&quot; was not found. Please verify your
component contains this data.</userErrorMessage>
<systemErrorCode>SYSERR_LOOKUP</systemErrorCode>
<systemErrorMessage>No results returned from LOOKUP: stateLookup : 3
==&gt; LOOKUP &quot;STATE&quot; FROM &quot;ComponentOne&quot; FILTER BY
&quot;&quot; WHERE STATE=&quot;FL&quot; ORDER BY &quot;&quot; LOCATE IN
PRODUCT &quot;Script Rule Exception Handling&quot; IN VERSION &quot;1&quot;</
systemErrorMessage>
<stackTrace>
```

ⓘ Note

The <stacktrace> is also shown.

ⓘ Note

At *Deployed Product Search*, selecting the *Expand* icon will expand both the *Request* and *Response* panels.

11. Selecting the icon will collapse both the *Request* and *Response* panels to display the *Deployed Product Search* panel.

6.16 Viewing Reference Metadata

Metadata is data that is used to describe other data. The data definition is used to add tables and attributes to components, flowstores, datastore definitions, and data definitions.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Go to *Product* > *View Reference Metadata*.
2. Complete these fields

Product Name	The name of the product.
Product Version	The version number of the product.
Reference Name	A reference name specifies the group names assigned to tables. To view metadata only for a particular reference name, select Reference Name and enter the name.
Display Columns	To view columns associated with the metadata, select Display Columns.

3. Choose *Next*.
   A screen appears, with the tables defined by the product's data definitions appearing in the tree panel.
4. Choose *Close* when you are finished.

6.17 Querying the Database

If the system is providing unexpected results, you may want to examine the database to view the raw data. Depending on your role and level of expertise, you may also be able to make changes to that data.

Prerequisites

Verify that you are in the correct server in the Administrative Console (Design Time or Runtime).

The *Execute SQL Queries* menu item will only be visible if the *System* > *Env* > *Debug Setting* > *EnableSQLConsole* configuration variable has been set to Yes. After the configuration variable's setting has been changed to YES, you will need to refresh the configuration settings by going to *System* > *Reload Configuration Setting* > *Reload System Config*. Then logout and log back into the Administrative Console.
The *Execute SQL Queries* menu item will be visible, if you have been assigned a role collection that contains the `Administrator_RT` role template

Context

> ⚠ Caution
>
> You must be extremely cautious about executing anything other than SELECT statements, or else the database may become corrupted with no way to recover.

To query the database, perform the following steps:

Procedure

1. Go to *System* > *Execute SQL Queries*.

   The *Query* screen appears.
2. Complete these fields:

Create Insert Statement	Select this option to automatically generate an insert statement based on this query.
Create Update Statement	Select this option to automatically generate an update statement based on this query.
Data Only	Select this option if you want to exclude columns containing null data. This option requires that either the *Create Insert Statement* or *Create Update Statement* option be selected.

All Columns	Select this option if you want to see all columns, even if they contain null data. This option requires that either the *Create Insert Statement* or *Create Update Statement* option be selected.

3. Enter a SQL query in the large area of the screen.
4. Select the numbered *Query* buttons at the top of the screen to create more than one query at a time.
   You can create a maximum of four different queries.
5. Choose the *Display Tables* option to view a list of table names that you can use in your query.
   This may help to ensure that you create the query accurately.
6. Choose the table name to view the names of the columns in a particular table.
   This may help to ensure that you create the query accurately.
7. Choose *Run Select Query*.

Results

The query results appear below the main screen area.

If you selected the *Create Insert Statement* or *Create Update Statement* options, the code for the insert or update query appears below the query results. You can copy and paste this code if necessary.

Related Information

Administration Role Templates [page 16]
Product OData Services Development User and Authorization Management [page 27]

6.18 Creating the RFC Destinations

RFC destination details are configured in a properties file, to allow connections to be integrated into other SAP systems.

Prerequisites

Confirm the destination names for FS-PM and SAP GP-FS RFC connectivity that's defined in the RFC reference object in FS-PRO.

For more information, see Configuring RFC Destinations in Products.

Procedure

1. Create a properties file for the FS-PM connection.
    a. Open a text editor.
    b. Add the following text to the file and add the appropriate JCo client settings in the file with connection details for the FS-PM system:

    ```
    jco.client.client=
    jco.client.user=
    jco.client.sysnr=
    jco.client.ashost=
    jco.client.lang=
    ```

    where:

    client

    Specifies the SAP client number.

    Three-digit client number; preserve leading zeros if they appear in the number.

    user

    Specifies the login user ID.

    User name for logging in to the SAP system.

    Ensure that the User ID entered here has all of the required permissions in FS-PM.

    This is a technical user account. SSO/Principal Propagation is not supported.

    sysnr

    Indicates the SAP system number.

    ashost

    Identifies the SAP application server.

    Host name of a specific SAP application server.

    lang

    Specifies a login language.

    Use the SAP-specific single-character language code.
    c. Save the file as `FSPM_DEST.jcoDestination` to a local directory.
2. Launch Internet Explorer and go to <quo_runtime_app_url>/csiroot/admin/.
3. Log in as the <admin_user>.
The Runtime Administrative Console will open after a short delay.
4. Upload the files to the file system.
    a. Go to *System* > *Edit File System*.

    A list of folders appears in the tree, and the folders and files in the selected folder appear in the details panel.
    b. From the *Path* dropdown list, select *PA - Custom Web Root (Write)*.
    c. Expand the `resource` folder in the tree and go to the `rfcdestination` folder.
    d. Right-click the `rfcdestination` folder, and choose *Upload*.

    The *Upload Files* dialog appears.

   e. Upload both of the destination properties files.

5. Add a new configuration variable for your FS-PM connection password entry.
   a. In the editor pane, select RFC Standalone Credential Setting from the *Configuration* dropdown list.
   b. Choose the *Add Config Variable* button.
   c. Enter **`FSPM_DEST_STANDALONE_PWD`** in the *Name* field.
   d. Enter the password for `<FS-PM_dest_name>` in the *Description* field.
   e. Ensure that the *Mandatory* checkbox is deselected.
   f. Enter **`Encrypt Text`** in the *Control* field.
   g. Ensure that the *Enumeration* field is blank.
   h. Ensure that the *Default* field is blank.
   i. Save your changes.
6. Add a new configuration variable for your SAP GP-FS connection password entry.
   a. In the editor pane, select RFC Standalone Credential Setting from the *Configuration* dropdown list.
   b. Choose the *Add Config Variable* button.
   c. Enter **`BP_DEST_STANDALONE_PWD`** in the *Name* field.
   d. Enter the password for `<BP_dest_name>` in the *Description* field.
   e. Ensure that the *Mandatory* checkbox is deselected.
   f. Enter **`Encrypt Text`** in the *Control* field.
   g. Ensure that the *Enumeration* field is blank.
   h. Ensure that the *Default* field is blank.
   i. Save your changes.
7. Go to *Configration* > *System* > *Env* in the configuration tree and select *RFC Standalone Credential*.
8. Enter the password of the user for the RFC connection to the FS-PM system in the *Override* field of the `FSPM_DEST_STANDALONE_PWD` property and choose *Save*.
9. Enter the password of the user for the RFC connection to the SAP GP-FS system in the *Override* field of the `BP_DEST_STANDALONE_PWD` property and choose *Save*.
10. Select *System* > *Reload Configuration Setting* from the menu bar and choose *Reload All Config* for the new password settings to take effect.

Related Information

Uploading Files [page 66]

6.18.1 Testing RFC Destinations

You need to verify that the connectivity defined in the destinations is functional.

Procedure

1. Log in to the Administrative Console.

   The Administrative Console will open after a short delay.
2. Select *System* > *Test SAP Server Connectivity* .
3. Go to the *Test JCo3 Feature* tab, select the destination and choose *Connect* to test the connection.
4. Repeat these steps for all destinations.

Results

A pop up message indicates whether the connection is valid or invalid.

6.19 Viewing Exceptions

Context

In order to view exception details in the *Debug Report*, you need to override the `SuppressErrorinAS` configuration variable setting. If this setting is not overridden, the *Details* pane on the right-hand side of the *Debug Report* will be blank. Only the exception message will be shown.

> Note
>
> Detail information will not be included for exceptions generated prior to the default setting being overridden.

Procedure

1. Go to *System* > *View Exceptions* .

   The *Error List* window opens, displaying a list of exceptions.

2. Select the *ErrorId* link to view the exception details.

Related Information

Application > Product Authority > Env > Application Environment [page 138]

7 Configuring Servers

You use the *Edit Configuration Settings* tool in the Administrative Console to load and edit the FS-PRO configuration settings.

> ⓘ Note
>
> To perform any procedure related to the Administrative Console, you require the appropriate permissions.

Related Information

FS-PRO Configuration Files [page 181]

7.1 Using the Edit Configuration Settings

This topic provides an overview of the *Edit Configuration Settings* feature.

Edit Configuration Settings has two main functions:

- Creating configuration variables. It also organizes the variables into logical groups. Because the system is installed with a default set of parameters and values, you may never or seldom need to use this feature.
- Overriding the default configuration variables.

Both modes exist within the same user interface, and their differences may not be readily apparent.

To access the *Edit Configuration Settings*, go to *System* > *Edit Configuration Settings*. The interface consists of a tree panel and a detail or main panel. The tree organizes related configuration variables in units called configuration entries. Related configuration entries are grouped in folders. The folders are grouped in one of the main folders: `System` and `PS` (common to all server types) or `PC`, `PPMC`, and so on. The main folders contain settings unique to a given type of server. In the tree, when you select a configuration entry, its settings appear in the main panel.

Note that warning symbols sometimes appear in the tree. A yellow triangle indicates the presence of a variable that has no value assigned but for which value assignment isn't mandatory. In this case, you can ignore the warning and the system should still function properly. A red icon with an "x" means there is a variable that has neither a default value nor an override value but for which value assignment is mandatory. In this case, for the setting to function properly, you must assign a value. In the detail or main panel, each variable that lacks values has one of these icons in its row header.

7.1.1 Creating a New Configuration Entry for a Server

This procedure describes how to create a new configuration entry for a server (System, PPMS, or PS).

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Right-click the server's folder in the *Configuration* tree and choose *Add Config Entry*.
   The *Add Configuration Entry* dialog appears.
3. Enter a name for the configuration entry in the *Configuration Name* field.
   This name will appear in the tree panel, under the configuration type (Env or Log). It will also appear in the *Configuration* dropdown list in the detail panel.
4. Enter the name of the file that will contain the configuration entries in the *File Name* field.
   This name will appear below the *Configuration* dropdown list in the detail panel.
5. In the *Configuration Type* dropdown list, specify where in the folder hierarchy the new folder goes.
   Valid values are Env and Log, and in an application server, Process.

   > ⓘ Note
   >
   > If you are creating a Process configuration entry, the *Configuration Name* and *File Name* fields are disabled. You only need to select the *Configuration Type*.
6. Save your changes.

Results

The new configuration entry appears in the folder that you specified in the *Configuration Type* dropdown list.

7.1.2 Creating a New Configuration Entry for an Application

This procedure describes how to create a new configuration entry for an application.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Expand the *Application* folder in the *Configuration* tree to reveal the installed application.
3. Right-click its folder in the *Configuration* tree and choose *Add Config Entry*.
   The *Add Configuration Entry* dialog appears.
4. Enter a name for the configuration entry in the *Configuration Name* field.

   This name will appear in the tree panel, under the configuration type (Env or Log). It will also appear in the *Configuration* dropdown list in the detail panel.
5. Enter the name of the file that will contain the configuration entries in the *File Name* field.

   This name will appear below the *Configuration* dropdown list in the detail panel.
6. In the *Configuration Type* dropdown list, specify where in the folder hierarchy the new folder goes.

   Valid values are Env and Log, and in an application server, Process.

   > Note
   >
   > If you are creating a Process configuration entry, the *Configuration Name* and *File Name* fields are disabled. You only need to select the *Configuration Type*.
7. Save your changes.

Results

The new configuration entry appears in the folder that you specified in the *Configuration Type* dropdown list.

7.1.3 Renaming a Configuration Entry for a Server

This procedure describes how to change the name of a configuration entry for a server.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Select a folder in the *Configuration* tree.
   A toolbar appears in the detail panel, along with a dropdown list containing the editable configuration entries in the folder.
3. Select the configuration entry from the dropdown list.
4. Choose the *Edit Config Entry* icon.
   The *Edit Configuration Entry* dialog appears.
5. Enter a new name for the configuration entry.
6. Save your changes.

Results

The configuration entry is renamed.

7.1.4 Renaming a Configuration Entry for an Application

This procedure describes how to change the name of a configuration entry for an application.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Expand the *Application* folder in the *Configuration* tree to reveal the installed application, and select it.
   A toolbar appears in the detail panel, along with a dropdown list containing the editable configuration entries in the folder.
3. Select the configuration entry from the dropdown list.
4. Choose the *Edit Config Entry* icon.
   The *Edit Configuration Entry* dialog appears.
5. Enter a new name for the configuration entry.
6. Save your changes.

Results

The configuration entry is renamed.

7.1.5 Deleting a Configuration Entry

This procedure describes how to delete a configuration entry for either a server or an application.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Navigate to the entry in the *Configuration* tree, right-click and choose *Delete Config Entry*.
   The *Delete Configuration Entry* dialog appears.
3. Choose *Delete*.

Results

The configuration entry is deleted.

7.1.6 Creating a Configuration Variable for a Server

A configuration variable is stored with related variables inside a configuration entry. The default variables are created at system installation. However, you can add new variables, and edit existing ones.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Select the folder for the server in the *Configuration* tree.
   A toolbar appears in the detail panel, along with a dropdown list containing the editable configuration entries in the folder.
3. From the *Configuration* dropdown list, select the configuration entry that you want to add the variable to.
   Its existing variables appear in the detail panel.
4. Select the row below which you want the new configuration variable to appear, and choose the *Add Config Variable* icon.
   An empty row appears below the selected row.
5. Complete the attributes for the new variable.
6. Save your changes.

Results

The configuration variable is created.

7.1.7 Creating a Configuration Variable for an Application

A configuration variable is stored with related variables inside a configuration entry. The default variables are created at system installation. However, you can add new variables, and edit existing ones.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Choose ▶ *System* ▶ *Edit Configuration Settings* ◀ from the menu bar.
2. Expand the *Application* folder in the *Configuration* tree to reveal the installed application, and select it.
   A toolbar appears in the detail panel, along with a dropdown list containing the editable configuration entries in the folder.
3. From the *Configuration* dropdown list, select the configuration entry that you want to add the variable to.
   Its existing variables appear in the detail panel.
4. Select the row below which you want the new configuration variable to appear, and choose the *Add Config Variable* icon.
   An empty row appears below the selected row.
5. Complete the attributes for the new variable.
6. Save your changes.

Results

The configuration variable is created.

7.1.8 Editing a Configuration Variable

This procedure describes how to update the values of a configuration variable.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Context

The following values can be updated:

- *Mandatory*
- *Control*
- *Enumeration*
- *Default*

Procedure

1. Choose *System* > *Edit Configuration Settings* from the menu bar.
2. Select the folder for the server or application in the *Configuration* tree.
   A toolbar appears in the detail panel, along with a dropdown list containing the editable configuration entries in the folder.
3. From the *Configuration* dropdown list, select the configuration entry where the variable you want to edit resides.
4. Select the variable you want to edit.

   Only the *Mandatory*, *Control*, *Enumeration*, and *Default* columns can be changed.
5. Save your changes.

   After you make changes to a configuration variable, you should reload its configuration entry. When you reload, you force the system to refresh the configuration variables loaded in the system cache. This feature is useful when you want to make quick changes without having to stop and restart the server.
6. Go to *System* > *Reload Configuration Settings*
   A list of options appears in the tree panel.
7. Select the configuration setting type that you want to reload, or choose *Reload All Config* to reload all the settings.

Results

An acknowledgment message appears in the detail panel.

7.1.9 Overriding Configuration Variables

At system installation, most of the configuration variables are given default values. In some cases, you may need to change values to reflect your environment.

Prerequisites

Verify that you are in the correct server in the Administrative Console.

Procedure

1. Navigate in the tree panel to the configuration entry containing the variable you want to change.
   The variables appear in the detail panel.

2. Add or select a new value in the *Override* field.
3. If you want to specify that the variable shouldn't have a default value, select the *Empty Value* checkbox and make sure that the *Override* field is blank.
4. Save your changes.
5. Choose the *Reload* icon to reload the values.

Results

The configuration variable is overridden.

7.1.10 Protected Configuration Variables

To provide psruntime application security, two protected flags `psruntime.default.user` and `psruntime.default.password` are implemented in *PS* > *Env* > *PWS Manager Setting* in the Administrative Console.

Any SOAP requests to psruntime require an additional SOAP secure header which contains (plaintext) username and password credentials that must be registered within the SAP user directory. A security handshake is done using the default username and password as psruntime communicates with psrepository internally only. If psrepository is accessed outside of psruntime, and fails authentication with the default credentials, it will perform a look up using the normal SAP user directory.

7.2 Understanding the Edit Configuration Settings Structure

The contents of *Edit Configuration Settings* differ depending on which server you are looking at and what's been installed on the server. However, the *Configuration* tree has two branches that exist on all servers, are always identical, and can't be edited: System and PS (the Product Services branch). The other folders, such as PPMC, PPMS, and PC are server-type dependent.

The remainder of this section is organized in two parts: the first describes the variables in the branches that are common to all servers; the second describes those that belong to the various servers.

7.2.1 Common Configuration Settings

7.2.1.1 System Configuration Settings

The settings in System apply to the system environment itself, and are divided between two subfolders: Env and Log.

7.2.1.1.1 System > Env > System Manager

Although each of the variables below are optional, every server must have at least one of these enabled. PPMC and PPMS can be enabled together to make a Design Time and Runtime system. All other combinations are not possible.

The following configuration variables are available:

PPMCEnabled

Must be set to Yes on the Runtime server.

The default value is No.

This setting is optional.

PPMSEnabled

This configuration variable is not supported. Don't edit or alter its value.

The default value is No.

This setting is optional.

IIPCEnabled

Specifies whether Product Modeler is running on this server.

The default value is No.

This setting is optional.

PEEnabled

This configuration variable is not supported. Don't edit or alter its value.

The default value is No.

This setting is optional.

7.2.1.1.2 System > Env > System Environment

The following configuration variables are available:

JavaCompilerOptions

Specifies the java compiler options.

This is a required setting.

The default value is: `-XDuseUnsharedTable -nowarn -classpath ${CSI.home}/system/lib/camilionlib.jar:${APP.home}/pa/custom/class:${APP.home}/runtime/class:${PS.home}/custom/class:$[PC.Classpath]:${CSI.home}/system/lib/camilion-apps.jar:${CSI.home}/system/lib/com.sap.fs.pqm.debug.jar`

EmailServerName

Specifies the URL of the server used for email.

This is a required setting.

No default value is provided.

UserDirectoryStoreType

Specifies the authentication and authorization mode.

This is a required setting.

The default value is Database.

UniqueSignOn

Specifies whether more than one person can sign on simultaneously to a server using the same user name.

Yes Specifies that only one user can sign on under a particular user ID at one time.

No Specifies that multiple users can sign on using the same user name.

This is the default value.

This is a required setting.

The default value is useful for a development environment.

DefaultPassword

Specifies the default password for a new user.

This is a required setting.

By default, users are prompted to change it immediately after they log in for the first time.

PasswordMaxDay

Specifies the number of days a new password remains valid before the system prompts to change it.

This is a required setting.

The default value is 25.

ErrorPage

Specifies a custom error page.

This is an optional setting.

No default value is provided.

XmlEncoding

Specifies encoding type used in XML files.

This is a required setting.

The default value is UTF-8.

HighPrecisionCalcMode

Specifies use of BigDecimal data type in the back-end for all script rule step calculations.

If this value is set to No, a Long data type will be used.

This is a required setting.

The default value is Yes.

FullLOOKUPExceptionHandling

Specifies backwards compatibility for rules created prior to Product Authority 7.1.

This is a required setting.

The default value is Yes.

SupportedBrowsersTridentVersion

List of supported browser user agents.

The default value is Trident/8.0;Trident/7.0;Trident/6.0;Trident/5.0;Trident8.0;Trident7.0;Trident6.0;Trident5.0

This setting is optional.

Sapui5Url

Specifies the absolute URL of an SAPUI5 distribution to which the application redirects requests of SAPUI5 resources.

Note that this setting has no effect if the application is deployed on SAP HANA XS Advanced, as redirection is handled by the approuter.

The default value is `https://sapui5.hana.ondemand.com/1.44.31`

7.2.1.1.3 System > Env > SSO Setting

The following configuration variables are available:

SSOImplementor

Single sign on (SSO) implementation to access the credentials by a third-party authorization server.

The default value is com.camilion.sso.UME_SSOHandler.

This setting is optional.

AdminSSOEnabled

Enables SSO for the Runtime Administrative Console.

The default value is No.

This setting is optional.

RuntimeSSOEnabled

Specifies the SSO-specific URL used to access the Runtime application.

The default value is No.

This setting is optional.

SSORuntimeURL

Specifies the SSO-specific URL used to access the Runtime application.

This setting is optional.

DesignerSSOEnabled

Enables SSO for Designer.

The default value is No.

This configuration variable is not supported. Don't edit or alter its value.

PCSSOEnabled

Enables SSO for the Product Modeler.

The default value is No.

This setting is optional.

UMEUserPrefix

Specifies the string prefix to expect on user unique IDs from the UME.

The default value is USER.PRIVATE_DATASOURCE.un:

This setting is optional.

UMERolePrefix

Specifies the string prefix to expect on role unique IDs from the UME.

The default value is ROLE.UME_ROLE_PERSISTENCE.un:

This setting is optional.

UMERoleNamePrefix

Specifies the string prefix to expect on user names from the UME.

The default value is PA_ROLE_.

This setting is optional.

UMEAuthProfilePrefix

Specifies the string prefix to expect on special authorization profile roles names from the UME.

The default value is PA_AUTH_.

This setting is optional.

> ⓘ Note
>
> Authorization Profiles are no longer supported. Similar functionality is provided in scopes and role collections.

Related Information

Resource URL Authorization [page 14]

7.2.1.1.4 System > Env > (PC only) JMS Setting

These configuration settings are not supported. Do not edit or alter these settings.

The following configuration variables are available:

JmsConnFactoryConfig

Specifies the JMS connection factory configuration file.

The default value is Apache ActiveMQ.

The override value is Application Server JMS Provider.

This is a required setting.

JmsBrokerURL

Specifies the JMS broker URL.

This value is applicable to the Product Modeler and ActiveMQ only.

This setting is optional.

QcfJndiName

Specifies the JNDI name of the QueueConnectionFactory.

This value is applicable to the application server JMS Provider only.

The override value is openejb:Resource/ifwConnection .

This setting is optional.

JMSWMQHost

Specifies the WebSphere MQ server host name.

This value is applicable to the Product Modeler and WebSphere MQ only.

This setting is optional.

JMSWMQPort

Specifies the WebSphere MQ server port.

This value is applicable to the Product Modeler and WebSphere MQ only.

This setting is optional.

JMSWMQQueueManager

Specifies the WebSphere MQ queue manager name.

This value is applicable to the Product Modeler and WebSphere MQ only.

This setting is optional.

JMSWMQChannel

Specifies the WebSphere MQ channel name.

This value is applicable to the Product Modeler and WebSphere MQ only.

This setting is optional.

JmsUsername

Specifies the username for accessing JMS.

This value is applicable to the Product Modeler only.

This setting is optional.

JmsPassword

Specifies the password for accessing JMS.

This value is applicable to the Product Modeler only.

This setting is optional.

JMSRecoveryInterval

Specifies the interval between recovery attempts in milliseconds.

The default value is 30000.

This value is applicable to the Product Modeler only.

This setting is optional.

JMSTransportOptions

Specifies the transport options that are appended to the `JmsBrokerURL` that provide additional configuration.

This value is applicable to ActiveMQ only.

The default value is ?maxReconnectAttempts=1.

This setting is optional.

7.2.1.1.5 System > Env > Debugging Setting

The following configuration variables are available:

EnableRuleTrace

Enables rule tracing.

In the Product Modeler, this setting lets you trace rules in the *Rule Unit Testing* dialog of *Rule Painter*.

In the Runtime application, you can access the trace results by clicking `Ctrl` + `+` + `F11`. These trace results are helpful for debugging rating rules.

For more information about rule tracing, see Rule Tracing.

In development environments, don't change the default setting. This setting must be enabled so that you can trace rules during testing and debugging.

In production environments, we recommend that you change this setting to No to improve performance.

If you change this setting to No in the Product Modeler, you won't be able to trace rules in the Runtime application even if this setting is set to Yes in the Runtime instance. This configuration isn't appropriate for production because rule testing is typically completed during development.

The default value is Yes.

This is a required setting.

EnableChangeLog

Enables the Design Time audit log, which enables the Runtime instance to record many of the basic actions that users perform, as well as who performed them and when.

The default value is Yes.

This setting is optional.

ProcessDataTraceEnabled

Enables trace on the methods `setString()` and `getString()` in `UpdateableResultSet`, which are used by processes and activities to write and read to the database. A data trace log file is created in <CSI_HOME>/env/log/dataTrace/. Pressing `Ctrl` + `+` + `F11` opens the file.

The default value is No.

This setting is optional.

ProcessStackTraceEnabled

Enables process stack trace for process and activities.

The default value is No.

This setting is optional.

EnableSQLConsole

Enables the SQL console and adds the *Execute SQL Queries* item on the *System* menu.

The default value is No.

This setting is optional.

7.2.1.1.6 System > Env > Content Setting

The following configuration variables are available:

ApplicationDataModelProduct

Specifies the product that will contain the main data tables for all applications.

This value must be defined with the same value in both the Design Time and Runtime Administrative Console.

Default setting: `Insurance Application Data Model Product`

This setting is optional.

ExtTblConnName

Specifies the external table connection name.

No default value is provided.

This setting is optional.

ExtTblConnStr

Specifies the external table connection string.

No default value is provided.

This setting is optional.

ExtTblConnUser

Specifies the external table connection user.

No default value is provided.

This setting is optional.

ExtTblConnPass

Specifies the external table connection password.

No default value is provided.

This setting is optional.

7.2.1.1.7 System > Env > Globalization Setting

The following configuration variables are available:

AvailableLanguages

Defines the language list available for selection as primary language for newly-created products.

For non-product type objects, content is supported in all languages specified in `AvailableLanguages`.

The default values are Chinese (Simplified), Chinese (Traditional), English, German, Japanese, and Korean.

This is a required setting.

GlobalPrimaryLanguage

Specifies the default primary language for newly-created products.

The default value is English.

This is a required setting.

7.2.1.1.8 System > Env > Globalization Format Setting

The following configuration variables are available:

DateStyle

Specifies the format of dates.

The default value is MEDIUM.

This is a required setting.

TimeStyle

Specifies the format of time.

The default value is MEDIUM.

This is a required setting.

DateTimeStyle

Specifies the format of dates and time.

The default value is LONG.

This is a required setting.

7.2.1.1.9 System > Env > Logger Config Setting

The following configuration variables are available:

AdminAuditTrailMode

Specifies the audit trail for the Administrative Console.

The default value is ENABLED.

This setting is optional.

XPathLogger.Enabled

Enables the XPath logger for XPath queries in the rule engine.

The default value is No.

This setting is optional.

XPathLogger.FileSize

Specifies the XPath log file size in bytes for log rotation.

The default value is 1024000.

This setting is optional.

XPathLogger.FileCounter

Specifies an XPath log file counter to limit the maximum rotated file numbers.

The default value is 5.

This setting is optional.

LogRetentionPeriod

Specifies the period of time to keep LRP logs.

The unit of measure is days.

The default value is 2.

This setting is optional.

UserActionLogRetentionPeriod

Specifies the period of time (in days) to keep user action logs.

The default value is 365.

This setting is optional.

LRPLogPath

Specifies the location of long-running process logs.

The default value is ${CSI.home}/env/log/lrp.

This setting is optional.

MaxRolloverFileSize

Specifies the maximum file size for a log file stored by the Administrative Console.

The default value is 1024000.

This setting is optional.

MaxRolloverFileNumber

Specifies the maximum number of rollover log files stored by the Administrative Console.

The default value is 10.

This setting is optional.

7.2.1.1.10 System > Env > Content Management System Setting

The following configuration variables are available:

CMSSerivce

Specifies the Service name of the Content Management System.

Default setting:
`com.sap.fs.uw.cms.service.smd.SAPMobileDocumentsService`

This setting is optional.

CMSServerURL

Specifies the URL of the Content Management System.

No default value is provided.

This setting is optional.

CMSServerUser

Specifies the logon user of the Content Management System.

No default value is provided.

This setting is optional.

CMSServerPassword

Specifies the password of the logon user of the Content Management System.

No default value is provided.

This setting is optional.

7.2.1.1.11 System > Env > Virus Scan Interface Setting

The following configuration variable is available:

VirusScanService

Specifies the service interface of the virus scan.

This configuration variable is important if you have configured a Content Management System (CMS) as part of your solution.

Default setting:
`com.sap.fs.uw.cms.service.impl.VirusScanServiceForNetWeaverVSI`

This setting is optional.

7.2.1.2 PS Configuration Settings

Product Services is the module that hosts the product repository, which contains all published products. The settings in `PS` are for Product Services, and are divided between two subfolders: `Env` and `Log`.

7.2.1.2.1 PS > Env > PS Environment

The following configuration variables are available:

DBDataDefinitionSync

Specifies whether to synchronize product data definitions against the database when the Product Modeler publishes a product to this server.

The default value is No.

This setting is optional.

EnabledDistributedPS

Enables the Distributed product service.

The default value is No.

This setting is optional.

GetProductsFromCsiHome

Enables using products from the EAR file if they are in the EAR file.

If they aren't in the EAR file, they are retrieved from the CSI_HOME directory (Runtime environment only).

If they are in neither location, an exception is thrown.

The default value is No.

This setting is optional.

WriteXmlDobDataToFile

Enables logging of the policy model Xml Dob data to the file system.

The default value is No.

This setting is optional.

7.2.1.2.2 PS > Env > Cluster Settings

The following configuration variables are available:

ClusterEnabled

Enables clustering.

The default value is No.

This setting is optional.

ClusterJMSProvider

Specifies the JMS provider for the clustered environment.

The default value is Apache ActiveMQ.

This setting is optional.

ClusterJMSBrokerURL

Specifies the URL of the cluster JMS broker.

This setting is applicable only to ActiveMQ.

No default value is provided.

This setting is optional.

ClusterJMSWMQHost

Specifies the host name of the WebSphere MQ server.

No default value is provided.

This setting is optional.

ClusterJMSWMQPort

Specifies the port number of the Websphere MQ server.

No default value is provided.

This setting is optional.

ClusterJMSWMQQueueManager

Specifies the name of the WebSphere MQ queue manager.

No default value is provided.

This setting is optional.

ClusterJMSWMQChannel

Specifies the name of the WebSphere MQ channel.

No default value is provided.

This setting is optional.

ClusterJMSUsername

Specifies the cluster JMS user name.

No default value is provided.

This setting is optional.

ClusterJMSPassword

Specifies the cluster JMS password.

No default value is provided.

This setting is optional.

ClusterConfig

Specifies the file that stores the properties for an environment that uses clustering.

No default value is provided.

This setting is optional.

ClusterCacheTopic

Specifies the cluster cache topic name.

No default value is provided.

This setting is optional.

ClusterCacheQueue

Specifies the cluster cache queue name.

No default value is provided.

This setting is optional.

ClusterEventTopic

Specifies the cluster cache event topic name.

No default value is provided.

This setting is optional.

7.2.1.2.3 PS > Env > Product Reference (PC only)

The following configuration variables are available:

FunctionsLibrary

Specifies the Function Library.

> **Note**
>
> If the FunctionsLibrary product JAR file isn't ready, leave this variable blank.

No default value is provided.

This setting is optional.

FunctionsLibraryVersion

Specifies the version number of the Function Library.

No default value is provided.

This setting is optional.

DataListSourceLibrary

Specifies the reference product that hosts the library of data lists (a standard system object).

The default value is Data List Base.

This is a required setting.

DataListSourceLibraryVersion

Specifies the version number of the reference product that hosts the library of data lists.

The default value is 1.

This is a required setting.

Related Information

Managing Function Libraries [page 76]

7.2.1.2.4 PS > Env > PWS Manager Settings

The following configuration variables are available:

NumProductBuilds

Specifies the number of builds per product.

The default value is 5.

This setting is optional.

RequestValidation

Specifies a request validation.

The default value is Yes.

This setting is optional.

ResponseValidation

Specifies a response validation.

The default value is Yes.

This setting is optional.

ServiceTrace

Enables Product Web Services tracing.

The webservice response will have the tracing information. It must be turned off in the production environment.

The default value is No.

This setting is optional.

ServiceAudit

Specifies a service audit.

The default value is Yes.

This setting is optional.

psruntime.default.user

The default username for authenticating the internal access to the psrepository services from psruntime.

This is a required setting.

psruntime.default.password

The default password for authenticating the internal access to the psrepository services from psruntime.

This is a required setting.

7.2.1.2.5 PS > Log > Publishing Log

The following configuration variables are available:

ps_db_synch

The log for data synchronization errors.

The default value is: `ps/publish_db_synch.log` .

This is a required setting.

7.2.1.2.6 PS > Env > Custom UI5 Setting

For SAP UI5 applications, you can use a custom JavaScript to execute in Runtime. This configuration entry is for settings that affect your application environment.

This file is located in `<CSI_HOME>/ppms/app/pa/custom/web/eapp/screen/controller/`. The file can updated to include custom functions, but it will then need to be uploaded again to the Administrative Console.

You can register your own custom JavaScript within the Administrative Console by overriding the default value with the fully qualified name of your JavaScript file.

The following configuration variables are available:

UI5CustomJS

Specifies the custom JavaScript to be used in the UI dispatcher.

The default value is `eapp.screen.controller.GlobalCustomController`.

7.2.2 Configuration Settings Unique to Servers

7.2.2.1 PC Configuration Settings

The Product Modeler is the module used to create and manage a catalog of products.

The Product Modeler allows the product manager to create customized products for specific markets, to specify state, industry, and channel variations, and to test product rules. Product definitions are stored for easy reuse and reconfiguring in a product repository within the Product Modeler.

7.2.2.1.1 PC > Env > PC Environment

These configuration settings are for the Product Modeler environment.

The following configuration variables are available:

QuesMaxTextBoxSize

Specifies the maximum size of a text box.

If the current question control length is greater than this value, a text area is displayed, otherwise a text box is displayed.

The default value is 100.

This setting is optional.

IIPCValuePageSize

Specifies the IIPC Value Page Size.

The default value is 15.

This is a required setting.

ServerUUID

Specifies the name identifying a server instance.

The value should be unique.

No default value is provided.

This is a required setting.

PrePostPublishingEnabled

Controls whether pre- and post-publishing service is enabled.

The default value is No.

This is a required setting.

ImportFormIdentifier

Specifies the name of a column in the `Forms Library` component that uniquely identifies form records; the form field mapping importer uses the name to cross-reference which field mappings belong under which forms.

The default value is FORM_NAME.

This setting is optional.

PrintInterfaceTemplate

Specifies the template used to represent the base structure of a web service XML required for third-party documentation printing (for example, DocuCorp).

The default value is com.camilion.connector.template.PublishingPrintInterface.

This setting is optional.

ProductSpecDocumentationFolder

Specifies the directory for storing the reports generated by the Product Modeler's *Product Specification Report* feature.

The default value is ${APP.home}/pa/custom/web/documentation.

This setting is optional.

AllowGenerationFromOutOfTheBoxViews

For *Product Specification Reports* feature, this lets you generate a report using the report "views" provided by SAP.

The default value is Yes.

This setting is optional.

AuditTrailMode

Specifies how the *Audit Trail* feature operates.

The options are:

ENABLED	The *Audit Trail* feature is enabled. This is the default setting.
DISABLED	The *Audit Trail* feature is disabled.

> → Tip
>
> The current setting appears in the top right corner of the *Audit Trail* dialog.

The default setting is ENABLED.

This setting is optional.

ProductSpecDocGenAfterPublishSelection

Specifies a default state of the generate documentation checkbox in the publish product dialog.

The default value is No.

This setting is optional.

DefaultProductSpecDocView

Specifies a default product specification documentation View to be generated from the publish product dialog.

No default value is provided.

This setting is optional.

CustomReleaseManagerStatus

Specifies the custom values dropdown list in the *Release Manager*.

The default value is In Progress,Completed,On Hold, Cancelled.

This is a required setting.

This configuration variable is not supported. Don't edit or alter its value.

IndexJobMaxWaitTime

Specifies the maximum time (in milliseconds) for an indexing job to wait for an in-progress one to finish.

The default value is 600000.

This setting is optional.

StudioCompanyFolderName

Specifies the folder name of the `Configuration Group` folder in System Repository used for IFBC imports.

The default value is Template Library.

This setting is optional.

UseCompUsagePublisher

Specifies the `Component Usage Publisher` as the publisher.

The default value is Yes.

This is a required setting.

UseLocalVariableDeclarationInRule

Enables the use of local variable declarations in rules.

The default value is No.

This is a required setting.

IFBCDataLoadingMode

Specifies whether XPath or In Memory Tree will be used to load data during an IFBC push.

The default value is Tree.

This is a required setting.

7.2.2.1.2 PC > Env > Message Queue Setting

These configuration settings are for the *Message Queue* option.

The following configuration variables are available:

ConsumerCount

Defines the number of concurrent JMS Consumer instances to be run in parallel.

Setting the value to 1 makes JMS Consumer singleton and all messages are consumed in FIFO order.

Setting the value to <n> allows <n> number of JMS Consumers to run and consume messages in parallel.

The default value is 1.

This is a required setting.

Asynch.Request.Queue

The name of the queue to which all messages should be sent and received from.

No default value is provided.

This is a required setting.

7.2.2.1.3 PC > Env > Message Template

These configuration settings pertain to the Custom Messaging feature in FS-PRO. The feature uses the Apache Velocity scripting language.

The following configuration variables are available:

resource.loader

Specifies the public names of the resource loaders to be used.

This public name is used in the specification for the properties for the resource loader.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

runtime.log.logsystem.class

Specifies the class to use for the Velocity-instantiated log system.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

velocimacro.library.autoreload

Controls Velocimacro library autoloading.

When set to true, the source Velocimacro library for an invoked Velocimacro will be checked for changes, and reloaded if necessary. This allows you to change and test Velocimacro libraries without having to restart your application or servlet container, just as with regular templates. This mode only works when caching is off in the resource loaders.

This option is meant for development, not for production.

The default value is true.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

velocimacro.library

Specifies the file names of the Velocimacro libraries to load when the Velocity Runtime Engine starts. These Velocimacros are accessable to all templates.

The file is assumed to be relative to the root of the file loader resource path.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camiliondsresource.loader.cache

Controls caching of the templates in the loader for the SAP datasource lookup.

The default value is false.

Set this configuration variable to true for production environments (this also enables the `modificationCheckInterval` configuration variable).

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camiliondsresource.loader.class

Specifies the name of the class that reads template files from a DataSource.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camiliondsresource.loader.description

Specifies the description string for the loader.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camilionds.resource.loader.modificationCheckInterval

Specifies the how often the file modification time is checked (in seconds).

The default value is 2.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camilionds.resource.loader.public.name

Specifies the public name for the loader class.

The default value is DataSource.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camilionds.resource.loader.resource.keycolumn

Specifies the ID column in the lookup table.

The default value is TEMPLATE_UID.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camilionds.resource.loader.resource.table

Specifies the lookup table.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camilionds.resource.loader.resource.templatecolumn

Specifies the template definition.

The default value is TEMPLATE_TEXT.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

camilionds.resource.loader.resource.timestampcolumn

Specifies the template timestamp (for last modification information).

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

customfs.resource.loader.cache

Controls caching of the templates in the loader for file system lookup.

The default value is false.

For production environments set this configuration variable to true (this also enables the `modificationCheckInterval` configuration variable).

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

customfs.resource.loader.class

Specifies the name of the class Velocity uses to read the template files.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

customfs.resource.loader.description

Specifies the description string for the loader.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

customfs.resource.loader.modificationCheckInterval

Specifies the frequency, in seconds, that the file modification time is checked.

The default value is 10.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

customfs.resource.loader.path

Specifies the roots from which the loader loads templates.

Templates can exist in subdirectories of this root.

No default value is provided.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

class.resource.loader.class

Specifies the name of implementation class for the loader.

The default loader is the file loader.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

7.2.2.1.4 PC > Env > Message Template Tools

These configuration settings are for the templates for the Custom Messaging feature in FS-PRO. These settings pertain to the Java classes that support the templates.

In each case, you need to provide the full class path name. The default paths and files are provided by SAP.

The following configuration variables are available:

publishTool

Specifies a template tool for publishing custom messages.

The default value is com.camilion.iipc.vt.tools.PublishTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

dateTool

Specifies a template tool for formatting dates and getting current date information.

The default value is com.camilion.iipc.vt.tools.DateTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

parseTool

Specifies template tool for parsing a number object to a long value.

The default value is com.camilion.iipc.vt.tools.ParseTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

extensionTool

Specifies a template tool for parsing the Extension and getting information from it.

The default value is com.camilion.iipc.vt.tools.ExtensionTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

collectionTool

Specifies a template tool that handles collections.

The default value is com.camilion.iipc.vt.tools.CollectionTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

lookupTool

Specifies a template tool for accessing the lookup table component.

The default value is com.camilion.iipc.vt.tools.LookupTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

escapeTool

Specifies a template tool for escaping special characters.

The default value is com.camilion.iipc.vt.tools.EscapeTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

rowTool

Specifies a template tool for accessing a data value row of a component.

The default value is com.camilion.iipc.vt.tools.RowTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

dimTool

Specifies a template tool for accessing dimension information.

The default value is com.camilion.iipc.vt.tools.DimensionTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

filterTool

Specifies a template tool for filtering data value rows.

The default value is com.camilion.iipc.vt.tools.FilterTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

numberTool

Specifies a template tool for number formatting.

The default value is com.camilion.iipc.vt.tools.NumberTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

numTool

Specifies a template tool for number formatting.

The default value is com.camilion.iipc.vt.tools.NumberTool.

This setting is optional.

This setting is no longer used by standard functionality, and may be deleted in a future release.

7.2.2.1.5 PC > Env > PWS Manager Setting

These configuration settings are for Product Web Services and the Product Modeler.

The following configuration variables are available:

ProductRepositoryUrl

Specifies the URL of the product repository.

Include the domain name when `IgnoreServerCertificate=No`.

No default value is provided.

This setting is optional.

ProductWSRuntimeUrl

Specifies the URL of the Product Web Services Runtime application server.

Include the domain name when `IgnoreServerCertificate=No`.

No default value is provided.

This setting is optional.

PWSUserName

Specifies the user name for Product Web Services.

The default value is admin.

This setting is optional.

PWSPassword

Specifies the password of the user.

This setting is optional.

IgnoreServerCertificate

Specifies that the server's certificate won't be checked while establishing an SSL connection.

The default value is Yes.

This setting is optional.

KeyStoreViewName

Specifies the name of the Keystore View where the certificate for HTTPS connections is stored.

This value is only needed when `IgnoreServerCertificate=No`.

The default value is FS-PQM.

This setting is optional.

7.2.2.1.6 *PC > Env > Logger Setting*

These configuration settings are for Product Modeler logging.

The following configuration variables are available:

ExportDataRow.FileSize

Specifies the file size in bytes.

The default value is 1024000.

This setting is optional.

ExportDataRow.FileCount

Specifies the file counter to limit maximum rotated file numbers.

The default value is 5.

This setting is optional.

7.2.2.1.7 *PC > Env > Product Log Item Setting*

These configuration settings are for Product Modeler logging.

The following configuration variables are available:

DisableProductImportExportLogging

Specifies whether the log will be written into a table.

The default value is Y.

This setting is optional.

7.2.2.2 PPMS Configuration Settings

PPMS is the Runtime server.

7.2.2.2.1 *PPMS > Env > PPMS Environment*

The following configuration variables are available:

CreateScreenXml

The default value is No.

This is a required setting.

ReuseData

The default value is Yes.

This is a required setting.

ScreenSourceType

The default value is HTML.

This setting is optional.

LazyFlowstoreInsert

By default, when a row is inserted to a flowstore table, a new row is added to all extension, product, and writable reference tables that are attached to the target flowstore table. In some cases, these supplementary rows remain empty, creating unused rows in the database as well as empty nodes in the XML representation of the flowstore.

When this configuration variable is enabled, it causes the system to add rows to supplementary tables only when there is data to be persisted to them.

The default value is No.

This setting is optional.

> ⓘ Note
>
> You should enable or disable this configuration variable only once per instance. After you have set the configuration variable, changing it may corrupt the database.

7.2.2.2.2 *PPMS › Env › Async Scheduler Setting*

The following configuration variables are available:

AsyncTask.Enabled

Activates the `Asynchronous Task Scheduler`.

The default value is false.

This is a required setting.

AsyncTask.Delay

For an asynchronous process, the number of milliseconds to wait until the task starts.

The default value is 1000.

This setting is optional.

AsyncTask.Period

Using this variable, the timer schedules the task to execute, at most, the specified period.

The default value is 60000.

This setting is optional.

AsyncTask.Cancellable

For an asynchronous process, specifies whether the task can be canceled.

The default value is true.

This setting is optional.

AsyncTask.FixedRateTask

Using this variable, the timer schedules the task at the fixed rate value.

The default value is false.

This setting is optional.

AsyncTask.Daemon

When an asynchronous process executes, and the system encounters an asynchronous task, a timer is created, whose associated thread may or may not be specified to run as a daemon.

Choose from one of the following values:

true	Specifies that the timer will be used to schedule repeating maintenance activities. The activities must be performed as long as the application is running, but shouldn't prolong the lifetime of the application.
false	Specifies that no repeating maintenance activities are scheduled.

The default value is true.

This setting is optional.

AsyncTask.ThreadSafe

For an asynchronous process, specifies whether two instances can run simultaneously.

Choose from one of the following values:

true	Specifies that the task will start regardless of whether or not another asynchronous task is already running.
false	Specifies that the task won't start if there are any instances of this task already running. If there are no instances running, the task will start.

The default value is true.

This setting is optional.

AsyncTask.ScheduledDate

Using this variable, the timer schedules the task to be executed on the specified date.

No default value is provided.

This setting is optional.

AsyncFailureEmailTo

When an exception occurs in an asynchronous task (that is, the task doesn't complete successfully), the system sends an email to the specified email address.

Use `AsyncFailureEmailFrom` to specify name of the sender.

No default value is provided.

This setting is optional.

AsyncFailureEmailFrom

Specifies name of the sender of the email that is sent when an exception occurs in an asynchronous task.

No default value is provided.

This setting is optional.

AsyncFailureSubjectPrefix

Specifies the prefix text for the subject line of notification emails sent when asynchronous task exceptions occur.

No default value is provided.

This setting is optional.

> ⓘ Note
>
> In these variables, `AsyncTask` represents the name of an asynchronous subprocess. Depending on your system, you may have several sets of these variables, each with different subprocess task names.

7.2.2.2.3 *PPMS* › *Env* › *Document Attachment Setting*

The following configuration variables are available:

UploadableFileTypes

Lists the suffixes of the file types that the FS-QUO Runtime application users can attach to transactions.

The default value is .doc,.xls,.pdf,.wav,.txt,.jpg,.jpeg.

This setting is optional.

> ⓘ Note
>
> If document attachment isn't configured, users will be able to attach any file type they choose.

UploadableFileContainingSpace

Controls whether the system accepts document attachments with spaces in the file name.

The default value is No.

This setting is optional.

7.2.3 Application Configuration Settings

The *Application* folder contains a subfolder for each application installed on the server (though only one application can be enabled at a time).

The following list indicates the possible application types:

- FS-QUO eApp Runtime with only eApp Runtime settings.
- Quotation and Underwriting application settings

7.2.3.1 ProductAuthority

7.2.3.1.1 Application > Product Authority > Env > Application Environment

These configuration settings are for the application environment.

The following configuration variables are available:

DefaultCSS

Specifies the default CSS file.

The default value is `Screen.css`.

This is a required setting.

WebRoot

Specifies the custom web root, if any.

The default value is ${APP.home}/pa/custom/web.

This setting is optional.

MailContentPath

Specifies the custom email content path, if any.

The default value is ${APP.home}/pa/custom/email.

This setting is optional.

ServletFilter

Specifies the custom servlet filter, if any.

No default value is provided.

This setting is optional.

WebFilter

Specifies the custom web filter, if any.

No default value is provided.

This setting is optional.

SignOnUrl

Specifies the URL of the custom sign-on page, if any.

No default value is provided.

This setting is optional.

ClassPath

Specifies the custom class path, if any.

No default value is provided.

This setting is optional.

DefaultClientPage

Specifies the default client page for a login user.

No default value is provided.

This is a required setting.

DefaultSchemeName

Specifies the default CSS ID for a login user.

No default value is provided.

This setting is optional.

CustomIncludeScript

Specifies the custom JavaScript file for a Runtime screen.

No default value is provided.

This setting is optional.

MandatoryIndicator

Specifies character to appear beside mandatory fields on screens and questionnaires.

The possible values are:

- 1=Before Label
- 2=After Label
- 3=Before Field
- 4=After Field

All four values specify the asterisk (*) as the indicator.

The default value is After Field.

This is a required setting.

ErrorDisplayType

Specifies the type of error display to be used for a screen that has errors.

Valid values are List and Field.

The default value is Field.

This is a required setting.

eAppSummaryScreenEnabled

Specifies that the last tab in eApp screen will not be displayed as summary screen if the flag is set to No.

The default value is Yes.

This setting is optional.

CustomErrorTypes

Specifies the custom error types to be ignored when re-running validation rules are being re-run and displayed in a screen.

No default value is provided.

This setting is optional.

eAppCustomGridEnabled

Specifies the use of the custom grid, which doesn't use the right click menu for its options.

The default value is No.

This setting is optional.

ErrorNumberIndicator

Specifies the position of the error number indicator.

The possible values are:

- 1=Before Label
- 2=After Label
- 3=Before Field
- 4=After Field

The default value is After Field.

This is a required setting.

Log4jXML

Specifies the location of the Log4j config file.

No default value is provided.

This setting is optional.

SuppressErrorInAS

Specifies the suppression of the stack trace in Design Time or Runtime. You will need to set this variable in the applicable Administrative Console to see a result.

The default value is Yes.

This is a required setting.

CURR_DECIMAL_PLACE_FILE_LOC

Specifies the custom definition for currency decimal place, if different than ISO-4217 standard.

A server restart will be requested for the location file change.

The default value is =${APP.home}/pa/custom/web/resource/CurrencyDecimalPlaceDef_default.csv.

This setting is optional.

FSPMBookingServiceEnabled

Enables FS-QUO to send policy XML to FS-PM

The default value is No.

WriteFSPMPolicyDataToFile

Enables FS-QUO to write policy data to `<CSI_Home>`.

The default value is No.

7.2.3.1.2 Application › Product Authority › Env › Custom Servlet

If you are using a custom servlet, you must register it here

The following configuration variables are available:

FlowletGroupImportServlet

The default value is com.camilion.servlet.build.FlowletGroupImportServlet.

This setting is optional.

SchemeImportServlet

The default value is com.camilion.servlet.build.SchemeImportServlet.

This setting is optional.

ProductFlowstoreSyncServlet

The default value is com.camilion.servlet.build.ProductFlowstoreSyncServlet.

This setting is optional.

7.2.3.1.3 *Application > Product Authority > Process*

The Process configuration entry contains the processes used by an application. If the *Configuration Tree* doesn't contain a Process entry, you can create it like any other entry, but note that only one Process entry is allowed per application.

Once Process exists, you can attach a list of processes to it. These processes then execute at Runtime.

The same list of processes appears when you select their application during cache reload.

Related Information

Creating a New Configuration Entry for a Server [page 102]
Creating a New Configuration Entry for an Application [page 103]
Renaming a Configuration Entry for a Server [page 104]
Renaming a Configuration Entry for an Application [page 104]
Deleting a Configuration Entry [page 105]
Working with the Application Cache [page 72]

7.2.3.2 AuthoritySuite

7.2.3.2.1 *Application > AuthoritySuite > Env > Application Environment*

These configuration settings are for the Runtime application environment.

The following configuration variables are available:

SignOnURL

Specifies the URL of the custom sign-on page, if any.

The default value is /csiroot/cservlet/LoginInToRuntime.

This is a required setting.

ProductServiceLog

The default value is Yes.

This setting is optional.

SUPPRESS_ATTACHMENT_ERROR

The default value is Yes.

This setting is optional.

SSO

The default value is bypass.

This setting is optional.

FormServletLog

The default value is No.

This setting is optional.

REPORT_PRODUCT_NAME

The default value is AuthorityUnderwriter Report.

This setting is optional.

ContextFactory

The default value is com.ibm.websphere.naming.WsnInitialContextFactory.

This setting is optional.

REPORT_URL

The default value is com.ibm.websphere.naming.WsnInitialContextFactory.

This setting is optional.

OOSE_LOG_ENABLED

The default value is Yes.

This setting is optional.

AscendantOne_URL

The default value is http://10.16.16.101/AscendantOne/WebServices/AscendantOneWS/AscendantOne.asmx.

This setting is optional.

LoggingEnabled

The default value is Yes.

This setting is optional.

NBAccountSearchQLClass

No default value is provided.

This setting is optional.

Google_Map_Api_Key

Specifies that the last tab in eApp screen will not be displayed as summary screen if the flag is set to No.

The default value is TEST.

This setting is optional.

AppConfig_Product_Name

The default value is Extended_Underwriting_Application_Configuration.

This setting is optional.

AppConfig_Product_Version

The default value is 1.

This setting is optional.

DebugEnabled

The default value is Yes.

This setting is optional.

CustomIncludeScript

Specifies the custom JavaScript file that includes in runtime screen.

The default value is /csiroot/as/j_scripts/ASGlobalFunctions.js.

This is a required setting.

CustomWebServiceEndPoint

The default value is customServiceEndPoint.

This setting is optional.

LoginSchemeName

Specifies the scheme name for login preferences.

The default value is AS Scheme.

This is a required setting.

SpringMessageDelegate

The default value is com.camilion.as.ifw.web.spring.servlet.SpringMessageDelegateImpl.

This setting is optional.

ClearAllSessionWhileResetClassLoader

Specifies that all sessions will be cleared, while resetting the class loader. The application context will be refreshed as well.

The default value is Yes.

This setting is optional.

ErrorDisplayType

Specifies the error display type for a screen that has errors.

The default setting is List.

This is a required setting.

eAppCustomGridEnabled

Specifies the use of the custom grid, which doesn't use the right click menu for its options.

The default value is No.

This setting is optional.

eAppCustomGridEnabled

Specifies the use of the custom grid, which doesn't use the right click menu for its options.

The default value is No.

This setting is optional.

ErrorNumberIndicator

Specifies the position of the error number indicator.

The possible values are:

- 1=Before Label
- 2=After Label
- 3=Before Field
- 4=After Field

The default value is After Field.

This is a required setting.

RuntimeApplicationURL

The default value is https://localhost:8080

This is a required setting.

DefaultClientPage

Specifies the default client page for a login user.

The default values is AS Client.

This is a required setting.

DefaultSchemeName

Specifies the default CSS ID for a login user.

The default value is AS Scheme..

This is a required setting.

SUPPRESS_ERROR_DETAILS

The default value is Yes.

This is a required setting.

FSPMBookingServiceEnabled

Enables FS-QUO to send policy XML to FS-PM

The default value is No.

This setting is optional.

WriteFSPMPolicyDataToFile

Enables FS-QUO to write policy data to `<CSI_Home>`.

The default value is No.

This setting is optional.

SuppressExceptionOnDuplicateFields

Specifies to not throw an exception when detecting a field value is populated into an extension table instead of (or at the same time) into the base table while both tables have the same field.

The configuration affects some RFC calls.

The default value is No.

This setting is optional.

CustomEAppLauncher

Specifies the custom class implementing EAppLauncherServer, which is used to launch the eApp.

No default value is provided.

This setting is optional.

ProcessLock

Locks a process to a user in AuthoritySuite, but Process ID.

The default value is No.

This setting is optional.

RefreshProduct

Interrupts the session to refresh risk-first products after deployment.

The default value is No.

This setting is optional.

RunDeltaDisplaySequencing

Disables the default algorithm for sequence numbering.

The default value is True

This is a required setting.

AppTimeOutPage

Specifies the application-specified timeout page.

The default value is /as/html/ASTimeOut.html?setErrorResponseCode=0.

This setting is optional.

DefaultTimeOutPage

Specifies the platform-specified timeout page.

No default value is provided.

This setting is optional.

7.2.3.2.2 Application › AuthoritySuite › Env › Interface Touchpoints

These configuration settings are for the Runtime application environment.

The following configuration variables are available:

Print_subproc

The default value is ASPrint.

This setting is optional.

AddressValidation_enabled

The default value is No.

This setting is optional.

ProducerChange_enabled

The default value is Yes.

This setting is optional.

TerminationService_enabled

The default value is No.

This setting is optional.

Renew_leadtime

The default value is 90.

This setting is optional.

OoseCommitHandlerClass

The default value is com.camilion.is.oos.interfaces.CommitHandlerInstance

This setting is optional.

Booking_subproc

The default value is ASBooking.

This setting is optional.

Allow_duplicate_product

The default value is Yes.

This setting is optional.

Document_link_enabled

The default value is No.

This setting is optional.

Quoted_policy_changeable

The default value is Yes.

This setting is optional.

7.2.3.2.3 Application › AuthoritySuite › Env › Risk Flowlets

These configuration settings are for the Runtime application environment.

The following configuration variables are available:

POLICY

The default value is 6710.

This setting is optional.

LOCATION

The default value is 6305.

This setting is optional.

BUILDING

The default value is 6714.

This setting is optional.

CLASSIFICATION

The default value is 7105.

This setting is optional.

ADDL_INSUREDS

The default value is 6814.

This setting is optional.

ADDL_INTERESTS

The default value is 6813.

This setting is optional.

PRODUCER

The default value is 2002.

This setting is optional.

ACCOUNT

The default value is 909.

This setting is optional.

ADDRESS

The default value is 4501.

This setting is optional.

DECLINE

The default value is 5801.

This setting is optional.

DRIVER

The default value is 12405.

This setting is optional.

VEHICLE

The default value is 12406.

This setting is optional.

7.2.3.2.4 Application > AuthoritySuite > Env > Custom Servlet

These configuration settings are for the Runtime application environment.

The following configuration variables are available:

Quote

The default value is com.camilion.custom.servlet.Quote

This setting is optional.

FormServlet

The default value is com.camilion.custom.servlet.policyservice.FormServlet

This setting is optional.

ReportServlet

The default value is com.camilion.indsol.weburl.report.ReportServlet

This setting is optional.

VariableData

The default value is com.camilion.custom.servlet.VariableData

This setting is optional.

BulkBlackout

The default value is com.camilion.custom.servlet.policyservice.BulkBackout

This setting is optional.

JWhichServlet

The default value is com.camilion.custom.servlet.JWhichServlet

This setting is optional.

RatingWorkSheet

The default value is com.camilion.custom.servlet.RatingWorkSheet

This setting is optional.

FormsList

The default value is com.camilion.custom.servlet.FormsList

This setting is optional.

CustomService

The default value is com.camilion.custom.servlet.CustomServiceServlet

This setting is optional.

FormChooser

The default value is com.camilion.custom.servlet.FormChooser

This setting is optional.

FormVariable

The default value is com.camilion.custom.servlet.FormVariable

This setting is optional.

RatingWorkSheetManager

The default value is com.camilion.custom.servlet.RatingWorkSheetManager

This setting is optional.

NavigationBar

The default value is com.camilion.custom.servlet.NavigationBar

This setting is optional.

CustomServletLookup

The default value is com.camilion.custom.servlet.CustomServletLookup

This setting is optional.

DocumentUploadServlet

The default value is com.camilion.custom.docarchive.DocumentUploadServlet

This setting is optional.

Manuscript

The default value is com.camilion.custom.servlet.ManuscriptServlet

This setting is optional.

Log

The default value is com.camilion.custom.servlet.LogServlet

This setting is optional.

Audit

The default value is com.camilion.custom.servlet.AuditServlet

This setting is optional.

OoseProcessor

The default value is com.camilion.is.oos.servlet.OoseProcessorServlet

This setting is optional.

ASDebug

The default value is com.camilion.custom.servlet.ASDebug

This setting is optional.

LoginInToRuntime

The default value is com.camilion.as.servlet.ASLogin

This setting is optional.

DeclineTxnTest

The default value is com.camilion.custom.servlet.debug.DeclineTxnTest

This setting is optional.

GenerateProductModelXSD

The default value is com.camilion.custom.servlet.GenerateProductModelXSD

This setting is optional.

VoidTxnTest

The default value is com.camilion.custom.servlet.debug.VoidTxnTest

This setting is optional.

ServiceComponentTest

The default value is com.camilion.as.service.ServiceComponentTest

This setting is optional.

DialogServerServlet

The default value is com.camilion.as.servlet.DialogServerServlet

This setting is optional.

SpringRefreshServlet

The default value is com.camilion.as.servlet.SpringRefreshServlet

This setting is optional.

FramePageServlet

The default value is com.camilion.as.servlet.FramePageServlet

This setting is optional.

getXMLforURS

The default value is com.camilion.custom.servlet.getXMLforURS

This setting is optional.

SerializeModel

The default value is com.camilion.custom.servlet.SerializeModel

This setting is optional.

TestEMFXpath

The default value is com.camilion.custom.servlet.TestEMFXpath

This setting is optional.

EMFXpathModelDump

The default value is com.camilion.custom.servlet.EMFXpathModelDump

This setting is optional.

ViewPremiums

The default value is com.sap.fs.quo.web.servlet.ViewPremiumsServlet

This setting is optional.

ViewRollups

The default value is com.sap.fs.quo.web.servlet.ViewRollupsServlet

This setting is optional.

BatchFlowBuildServlet

The default value is com.camilion.apps.common.servlet.BatchFlowBuildServlet

This setting is optional.

EappRendererHandlerServlet

The default value is com.camilion.custom.servlet.EappRendererHandlerServlet

This setting is optional.

ViewVersionInfo

The default value is com.camilion.custom.servlet.ViewVersionInfo

This setting is optional.

InsurableObjectServlet

The default value is com.camilion.as.integration.fspm.servlet.InsurableObjectServlet

This setting is optional.

BusinessPartnerServlet

The default value is com.camilion.as.integration.fspm.servlet.BusinessPartnerServlet

This setting is optional.

ASIntegrationServlet

The default value is com.camilion.as.integration.fspm.servlet.ASIntegrationServlet

This setting is optional.

ContextServlet

The default value is com.sap.custom.servlet.ContextServlet

This setting is optional.

7.2.3.2.5 Application > AuthoritySuite > Env > FS-PM Integration Setting

These configuration settings are for the Runtime application environment.

The following configuration variables are available:

FSPMEnabled

The default value is No.

This setting is optional.

jco.client.client

No default value is provided.

This setting is optional.

jco.client.sysnr

No default value is provided.

This setting is optional.

jco.client.ashost

No default value is provided.

This setting is optional.

jco.client.user

No default value is provided.

This setting is optional.

jco.client.passwd

No default value is provided.

This setting is optional.

jco.client.lang

The default value is en.

This setting is optional.

icm.application.id

This value is required when Native APIF booking service is used so that the participant role of a business partner can be retrieved from the ICM system per this ICM application ID.

No default value is provided.

This setting is optional.

BAPIExchangeRateGetFactorsMethod

The default value is BAPI_EXCHANGERATE_GETFACTORS.

This setting is optional.

BAPIExchangeRateGetFactorsType

The default value is M.

This setting is optional.

policy_rfc_destination

This value specifies the RFC Destination for FS-PM.

No default value is provided.

This setting is optional.

insuobj_ref_destination

This value is the RFC Destination for insurable objects.

No default value is provided.

This setting is optional.

bpartner_rfc_destination

This value specifies the RFC Destination for FS-BP.

No default value is provided.

This setting is optional.

model.calc.IV_REFUSALREASON_CD

This value specifies the refusal reason for model calculation.

The default value is 1.

This setting is optional.

model.calc.number.range

This value specifies the number range for application number and policy number for model calculation.

The default value is 0.

This setting is optional.

CURRENCY_CODE_OVERRIDE_ENABLED

This value specifies that the currency code be overridden in the chile tables CURRENCY_ID.

The default value is No.

This setting is optional.

7.2.3.2.6 Application › AuthoritySuite › Env › Form Attachment

These configuration settings are for the Runtime application environment.

The following configuration variables are available:

StateColumn_default
: The default value is POLICY_QUOTE.PRIMARY_RISK_STATE_CD.

  This setting is optional.

StateColumn_LOCATION
: The default value is LOC_ADDRESS.STATE_PROV_CD.

  This setting is optional.

StateColumn_TableName
: The default value is LOCATION.

  This setting is optional.

7.2.3.2.7 Application › AuthoritySuite › Log › Rating Log

These configuration settings are for the Runtime application rating log.

The following configuration variables are available:

RATING_DEBUG
: The default value is ppms/app/as/logs/rating/rating_debug.log.

  This setting is optional.

7.2.3.2.8 Application › AuthoritySuite › Log › Timer Log

These configuration settings are for the Runtime application timer log.

These configuration settings are for the Runtime application rating log.

TIMERLOG
: The default value is ppms/app/as/logs/timerservice/timer.log.

  This setting is optional.

7.2.3.2.9 Application › AuthoritySuite › Log › Form Service Log

These configuration settings are for the Runtime application form service log.

These configuration settings are for the Runtime application rating log.

FORM_XPATH

The default value is ppms/app/as/logs/formservice/form_xpath.log.

This setting is optional.

FORM_XPATH_DUMP

The default value is ppms/app/as/logs/formservice/form_xpath_result.log.

This setting is optional.

FORM_METHOD_CALL

The default value is ppms/app/as/logs/formservice/form_method_result.log.

This setting is optional.

FORM_DEBUG

The default value is ppms/app/as/logs/formservice/form_debug.log.

This setting is optional.

7.3 Using the SAP Server Connection Test

After you configure your remote SAP servers in the Administrative Console, you can verify that your server configuration works correctly. This test uses JCo2 and JCo3 protocols to verify that the servers configured are correct and that connectivity based on the provided configuration works without issues.

Context

To test the connectivity of your servers, perform the following steps:

Procedure

1. Log into the FS-PRO Administrative Console.
2. Go to *System* > *Test SAP Server Connectivity*.
   The *Test Connectivity* window appears with a list of your configured servers.
3. Choose *connect* in the *Action* column to connect to that server.

Results

A window appears with connection details. If your connection was successful, your server connection is working properly.

Next Steps

If you would like to temporarily change any of the connection parameters you can do so.

> ⚠ Caution
>
> Changes can be tested but not saved.

If your connection failed, you must reconfigure the SAP Server.

If you want to refresh the list of servers and run the test again, choose *Reset* and the list of servers is reset.

7.4 Overriding the JDBC Connection Pool Settings

You can configure connection pool settings for the JDBC data sources set up in Tomcat and TomEE, that are part of the SAP Java Buildpack, on SAP HANA XS Advanced (XSA) environments. FS-PRO uses the SAP Java Buildpack to define and customize the data source.

The SAP Java Buildpack includes a custom (wrapper) implementation of the JDBC data source provider. It provides some additional functionality specific to SAP HANA XS Advanced (XSA), including the resolution of JDBC connection details from services, multi-tenancy, and connection personalization.

The data source is defined in `/WEB-INF/resources.xml` in the webapp for TomEE. The default for the variables are provided in `/META-INF/sap-java-buildpack/config/resource_configuration.yml`.

The default values can be overridden during deploy time using the manifest file using the `JBP_CONFIG_RESOURCE_CONFIGURATION` property, if you push the webapp directly.

The following list details the configurable attributes for the FS-PRO application:

maxActive	The maximum number of active connections that can be allocated from this pool at the same time. Data type: int The default value is 100.
maxIdle	The maximum number of connections that should be kept in the pool at all times. Idle connections are checked periodically (if enabled) and connections that have been idle for longer than `minEvictableIdleTimeMillis` will be released. Data type: int The default value is 100.
minIdle	The minimum number of established connections that should be kept in the pool at all times. The connection pool can shrink below this number if validation queries fail. Data type: int The default value is 0.

initialSize	The initial number of connections that are created when the pool is started. Data type: int The default value is 0.
maxWaitTime	The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception. Data type: string (time) The default value is 180000 millisecond
testOnBorrow	The indication of whether objects will be validated before being borrowed from the pool. If the object fails to validate, it will be dropped from the pool, and an attempt will be made to borrow another. Data type: Boolean The default value is false.
testOnConnect	The indication of whether objects will be validated when a connection is first created. If an object fails to validate, it will throw an SQLException. Data type: Boolean The default value is false.
testOnReturn	The indication of whether objects will be validated before being returned to the pool. Data type: Boolean The default value is false.
testWhileIdle	The indication of whether objects will be validated by the idle object evictor (if any). If an object fails to validate, it will be dropped from the pool. This property has to be set in order for the pool cleaner/test thread to run. Data type: Boolean The default value is false.
validationQuery	The SQL query that will be used to validate connections from this pool before returning them to the caller. If specified, this query doesn't have to return any data, it just can't throw a SQLException. If not specified, connections will be validated by the `isValid()` method. The default value is null. Data type: string For SAP HANA, set this value to SELECT 1.
validationQueryTimeout	The timeout in seconds before a connection validation queries fail. This works by calling `java.sql.Statement.setQueryTimeout(seconds)` on the statement that executes the validationQuery.

	The pool itself doesn't timeout the query, it is still up to the JDBC driver to enforce query timeouts. A value less than or equal to zero will disable this feature. Data type: int The default value is -1.
validatorClassName	The name of a class which implements the `org.apache.tomcat.jdbc.pool.Validator` interface and provides a no-arg constructor (may be implicit). If specified, the class will be used to create a Validator instance which is then used instead of any validation query to validate connections. Data type: string The default value is null.
timeBetweenEvictionRuns	The number of milliseconds to sleep between runs of the idle connection validation/cleaner thread. This value shouldn't be set to under 1 second. It dictates how often a check is performed for idle, abandoned connections, and how often idle connections are validated. Data type: string (time) The default value is 180000 millisecond.
numTestsPerEvictionRun	The number of connections to examine during each run of the idle connection evictor thread (if any). Data type: int The default value is 3.
minEvictableIdleTime	The minimum amount of time an object may sit idle in the pool before it's eligible for eviction. Data type: string (time) The default value is 30 minutes.
poolPreparedStatements	If true, a statement pool is created for each Connection and PreparedStatement. The pools are created by one of the following methods: • `public PreparedStatement prepareStatement(String sql)` • `public PreparedStatement prepareStatement(String sql, int resultSetType, int resultSetConcurrency)` Data type: Boolean The default value is false.
maxOpenPreparedStatements	The maximum number of open statements that can be allocated from the statement pool at the same time, or zero for no limit. Data type: int The default value is 0.
removeAbandoned	Flag to remove abandoned connections if they exceed the `removeAbandonedTimeout`.

	If set to `true` a connection is considered abandoned and eligible for removal if it's been in use longer than the `removeAbandonedTimeout` value. Setting this to `true` can recover database connections from applications that fail to close a connection. Data type: Boolean The default value is true.
removeAbandonedTimeout	Timeout in seconds before an abandoned (in use) connection can be removed. The value should be set to the longest running query your application might have. Data type: int The default value is 10800.
logAbandoned	Flag to log stack traces for application code which abandoned a connection. Logging of abandoned connections adds overhead for every connection borrow because a stack trace has to be generated. Data type: Boolean The default value is true.
jdbcInterceptors	The property contains a list of semicolon separated class names of JDBC interceptors. The default value is `ConnectionState`, which is used to enable auto commit on connection return. Data type: string This value should **not** be overridden.
validationInterval	Avoid excess validation, only run validation at most at this frequency - time in milliseconds. If a connection is due for validation, but has been validated previously within this interval, it won't be validated again. Data type: int The default value is 3000.
commitOnReturn	If `autoCommit` is `false` then the pool can complete the transaction by calling commit on the connection as it is returned to the pool. If `rollbackOnReturn` is `true` then this attribute is ignored. Data type: Boolean The default value is true. This value should **not** be overridden.

ⓘ Note

The values for the attributes must be of the native data type stated above. Integer and Boolean values should **not** be enclosed by single quotes, but the time with unit values should be treated as strings.

Example

The following `manifest.yml` snippet depicts how to override the initial connection pool size and the max connection wait time:

```
---
applications:
- name: FS-PRO-webapp


...


env: {
JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee/webapps/ROOT/WEB-INF/
resources.xml': { 'service_name_for_camiliondb': 'my-pro-db', 'initialSize' :
1 , 'MaxWaitTime' : '360000 millisecond' } ]",


...
```

8 Developing Applications

Applications are composed of flowstores, datastores, and data definitions.

If an application development team wants to achieve more extensive customization, it can create and manage J2EE component and web content within its own application-level EAR. In this case, the team creates and manages its own application under a different context root (other than "/"). However, this approach should be carefully considered, since it can't leverage the FS-PRO platform to its fullest extent and greatly increases the complexity of the project.

All customized FS-QUO content is placed in specified locations within the `CSI_HOME` directory, where it's automatically accessed by SAP Runtime code.

8.1 Customizing FS-PRO Applications

8.1.1 Directories and the JVM

All product and application content resides under the `<CSI.home>` JVM argument directory called the `Product_Home` directory. All the application-specific content, including all artifacts created by FS-PRO, resides under the `<CSI.home>/ppms/app` directory, known in the JVM as the `Application Home` directory.

8.1.2 Auto-Process Classes

The auto-process classes are directly executed by the Runtime instance. A custom ClassLoader loads these classes into the JVM. Any other supporting classes for auto processing are also loaded by the same custom ClassLoader if they haven't already been loaded by the system.

Related Information

The Resource Loader [page 162]
Custom-Class Coding Standards [page 163]
Using Third-Party Class Libraries [page 164]

8.1.3 Servlets

In order to avoid merging customized application content with the base product distribution EAR for each package, don't include application-specific servlets. To make these features available for applications, use the standard servlet skeleton classes available in the product. You should externalize the actual business logic into custom classes.

A standard servlet (`com.camilion.servlet.CsiServlet`) is available in FS-PRO to serve all application-specific servlet needs. You can call this servlet with the custom class name you want to run. The custom class should extend from the `com.camilion.custom.Servlet` abstract class and override the following static method: `public void service( HttpServletRequest request, HttpServletResponse response )`

All URLs containing `/csiroot/cservlet/` and followed by `http://<server>:<port>` are mapped to the CsiServlet servlet. To map the custom servlet to an actual custom servlet implementation class (for example, for a custom servlet named `com.camilon.custom.SampleServlet`) the following entry should appear in the `cservlet.config` file: `SampleServlet=com.camilon.custom.SampleServlet`

The URL `http://<server>:<port>/csiroot/cservlet/SampleServlet` invokes the above servlet. Any arguments passed to this servlet can be accessed in the custom servlet implementation class using standard Java http APIs.

The custom servlet name should be registered in the Administrative Console. Navigate to *Application* > *Edit Configuration Settings* > *Configuration* > *Application* > *ProductAuthority* > *Env* and in the *Custom Servlet* node enter the actual implementation class name. The custom servlet name should be implemented as a thread-safe class and must follow the singleton pattern. The getInstance method should return the only instance of the actual implementation custom class.

The following sample shows the custom servlet output of `com.camilon.custom.SampleServlet`, which submits browser data with the post command to the same servlet:

```
<html>
<head>
<title> title goes here </title>
</head>
<body>
<form action='/cservlet/SampleServlet' method='post' enctype='application/x-www-form-urlencoded'>
…
</body>
</html>
```

SampleServlet.java

```
package com.camilion.custom.servlet;
import java.util.Properties;
import com.camilion.util.HttpUtil;
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.ServletException;
import com.camilion.custom.Servlet;
```

```
public class SampleServlet extends Servlet {
    private SampleServlet() {
    }
    public void service( HttpServletRequest request,
         HttpServletResponse response ) throws IOException, ServletException{
            StringBuffer loHtml = new StringBuffer();
            Properties loProperties = HttpUtil.parseInput( request );
        int liCount = Integer.parseInt( loProperties.getProperty( "COUNT",
"0" ) );
            loHtml.append( "<html>" );
            loHtml.append( "<head>" );
            loHtml.append( "<title> title goes here </title>" );
            loHtml.append( "</head>" );
            loHtml.append( "<body>" );
            loHtml.append( "<form action='/cservlet/SampleServlet' method='post'
enctype='application/x-www-form-urlencoded'>" );
    loHtml.append( "<input type=text name='COUNT' value='" + (++liCount) + "'>");
            loHtml.append( "<INPUT TYPE='button' name='button' value='Submit'
onclick='document.forms[0].submit();'>" );
            loHtml.append( "</body>" );
            loHtml.append( "</html>" );
            response.getWriter().println( loHtml.toString() );
    }
                public static Servlet getInstance() {
                return new SampleServlet();
    }
}
```

8.2 Managing Custom Classes

8.2.1 The Resource Loader

FS-PRO contains a custom class loader used for loading all custom classes including auto-process classes. These classes, and all their supporting classes other than the ones that are part of FS-PRO, should be deployed in the development server under: `<CSI.home>/ppms/app/pa/custom/class`.

In the Administrative Console, navigate to *System* > *Edit File System*. From the *Path* dropdown list, select PA - Custom Class Folder, and navigate the tree to the stems node.

If these classes are packaged into a JAR file, they should be deployed under `<CSI.home>/ppms/app/pa/custom/lib`.

In the Administrative Console, navigate to *System* > *Edit File System*. From the *Path* dropdown list, select PA - Custom & 3rd Party Lib and select the ext node.

> **Note**
>
> If any resource files are accessed using the `ResourceLoader.getInstance` to call the methods `getResourceAsStream` and `getResource` from a particular JAR file, the application server instance will lock that JAR file. The same JAR file can't be hotdeployed without restarting the application server instance.

8.2.2 Custom-Class Coding Standards

Each custom class should extend the appropriate standard FS-PRO ancestor class. These classes shouldn't use `System.out` and `System.err`. Instead, they should use the standardized logging and error handling mechanism discussed below. You should only use the built-in configuration mechanism and never write code that depends on the internal tables of the Product Modeler. These classes shouldn't update any tables other than their own flowstore rows and should never update another transaction's data.

> ⓘ Note
>
> Avoid updating the Product Modeler internal tables. Doing so can break the threading model of FS-PRO and can affect user concurrency.

8.2.2.1 Ancestor Classes

The implementation classes may use third-party classes and drivers that aren't packaged with FS-PRO. These third-party libraries and drivers and the custom classes should be reachable by the product—that is, they should be in the classpath or they should by deployed under: `<CSI.home>/ppms/app/pa/custom/lib/ext`.

The ancestor classes available in FS-PRO are as follows:

com.camilion.security.Authenticator	Custom authentication classes should extend this abstract class.
com.camilion.custom.Servlet	Custom servlet classes should extend this abstract class.
com.camilion.processengine.processserver.AutoProcess	All application-specific auto-process classes should extend from this abstract class and override the following method: `public ServiceCommand execute(ServiceCommand aoServiceCommand)`
com.camilion.util.functions.FunctionLibraryBase	Custom rule stems should extend from this class.

8.2.2.2 Logging and Error Handling

FS-PRO provides standard facilities to handle logging. All custom classes should use this facility to write application log files.

> ⓘ Note
>
> An administrative tool is provided for enabling and disabling the log entries.

Use *Edit Configuration Settings* in the Administrative Console to set up different log types, along with different log file names, for different sets of custom classes.

To write a log to the log file belonging to that log type, specify the logger type when getting a CsiLogger instance. The `CsiLogger.getAppLogger` method returns a CsiLogger instance. Use one of the appropriate methods to write log files:

```
CsiLogger.getAppLogger( "<logger type>" );

CsiLogger.getLogger( SystemConfig.APP, "<logger type>" );
```

The `<logger type>` is the log entry in *Edit Configuration Settings* under *Configuration* > *Application* > *ProductAuthority* > *Log*. The node is named `<name>` *Log*. The entry name should be `<logger_type>` and the value should be the name of the file located at `<CSI.home>/env/log`.

8.2.2.3 External Resources

To access and execute stored procedures, packages, flowstores, and reference tables in FS-PRO schemas, get the connection by using the method: `AppEnvironment.getJDBCConnection()`

> **Note**
>
> Don't use custom classes to manage the transaction; doing so will break the product JTA model.

Be sure to call the return connection in the final code block:
`AppEnvironment.returnConnection(connection)`

8.2.3 Using Third-Party Class Libraries

If your custom classes use any third-party classes (whether from open-source or licensed versions), they should by deployed under `<CSI.home>/ppms/app/pa/custom/lib/ext`

Place any licensing information or license files in the same directory, unless the third party vendor specifies a requirement for a separate directory structure.

8.3 Creating Custom Stems

A custom stem is a rule stem that you create yourself. Broadly, the steps required are:

1. Create your Java class.
2. Upload the class using the Administrative Console.
3. Register the class using the Administrative Console.
4. Reset the class loader.

8.3.1 Custom Stem Java Classes

Your Java class must meet the following conditions:

- It extends from `com.camilion.util.functions.StaticFunctionLibrary`.
- Its exposed methods should be public static.
- It includes a static String array variable that defines the information that appears when the end user types the stem name in *Rule Painter*.
- It includes the mandatory `getUsageArray` and `getInstance` functions.

Related Information

Example: Custom Stem Class Standard Rules [page 167]

8.3.2 Naming Custom Stems

An important restriction applies when naming a custom stem: the name must not be the same as, or end with, the name of any system stem (for example, FORM, TREE, DATA, USER, RULE, INPUT, etc.).

For example, these names would be illegal: VARIABLEFORM, OURFORM, VARDATA; whereas these names would be acceptable: FORMVARIABLE, OURFORMS, VARDATAX.

> ⓘ Note
>
> Ignoring this restriction can cause unexpected behaviors. For example, if you create a custom stem named VARDATA, and type vardata. in the *Rule Painter* expression field, instead of a list of custom functions appearing, the system launches the column picker (just as if you typed data. in the field).

8.3.3 Uploading Custom Stem Java Classes

The following procedure describes how to upload your custom stem class to the FS-PRO environment.

Prerequisites

You may need to create the target folder if it doesn't exist in the file system.

Context

Procedure

1. Log in to the Design Time Administrative Console.
2. Go to *System* > *Edit File System*.
3. Choose PS - Custom Classes Folder from the *Path* dropdown list.
4. Navigate the file system and upload the file to the directory that is similar to your package structure.

Next Steps

You should now register the class.

8.3.4 Registering Custom Stem Java Classes

The following procedure describes how to register a custom stem class.

Context

To register a custom stem Java class, perform the following steps:

Procedure

1. If necessary, log on to the Administrative Console for the environment where you uploaded the class file.
2. Go to *System* > *Register Function Library*.
   The *Library Manager* appears.
3. Register the class.
4. Reset the class loader by going to *System* > *Reset ClassLoader*.

8.3.5 Adding Stem Functions to the *Rule Painter* Popup

In the *Rule Painter* expression field when you type a stem name followed by a period (.) a popup dialog appears listing the functions available for the stem. You use a static 2-D String variable in your custom stem to define this list of functions. This String variable is written as follows:

```
static String[][] USAGE = new String[][] {
    { FUNC, "getString", "CsiString", "" },
        { ARG, "aoServiceCommand", "ServiceCommand", ""},
        { ARG, "asColumnName", "String", "" },
;
```

The keyword FUNC denotes that it is a function. The arguments that follow this are the function name and return type.

→ Tip

If you want to return a String from the function, then you need to typecast it to CsiString in a standard rule

The keyword ARG means that this is the argument. For each argument there should be one array. ARG is followed by the parameter name and type. The fourth argument isn't used.

It's recommended as a best practice that the variable "serviceCommand" now be used in Script Rules when calling custom stems.

⚠ Restriction

The variables: CsiString, CsiBoolean, CsiDouble, Csitimestamp and CsiDataTable can't be used in Custom Classes for script rules.

Related Information

Example: Custom Stem Class Standard Rules [page 167]
Example: Custom Stem Class Script Rules [page 169]

8.3.6 Example: Custom Stem Class Standard Rules

The following is an example of how to implement a custom stem class in standard rules.

```
package com.camilion.custom.stem;
import com.camilion.base.*;
import com.camilion.data.DataTable;
import com.camilion.util.functions.StaticFunctionLibrary;

import java.text.SimpleDateFormat;
import java.sql.Timestamp;
public class CustomDataSource extends StaticFunctionLibrary {
// We use a singleton so we can have the static methods call instance methods.
```

```
// This allows for easier unit testing.
private static CustomDataSource instance = new CustomDataSource();

static String[][] USAGE = new String[][] {
{ FUNC, "getString", "CsiString", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
{ FUNC, "getNumber", "double", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
{ FUNC, "getDate", "CsiTimestamp", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
{ FUNC, "getDouble", "double", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
};
public String[][] getUsageArray(){
return USAGE;
}
public static CustomDataSource getInstance()
{
return instance;
}
public CustomDataSource()
{
super();
// TODO Auto-generated constructor stub
}
public static CsiString getString(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable)aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return new CsiString(loTable.getString(asColumnName));
}
public static double getNumber(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return Double.parseDouble(loTable.getString(asColumnName));
}
public static double getDouble(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return Double.parseDouble(loTable.getString(asColumnName));
}
public static CsiTimestamp getDate(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
String lsValue = loTable.getString(asColumnName);
return new CsiTimestamp(convertTimestamp(lsValue));
}
private static void validateDataTable(DataTable aoTable, String asColumnName)
throws Exception{
if ( !aoTable.hasColumn(asColumnName) )
    throw new CsiException("Exception while running getString method",
        asColumnName + " does not exist in the Datatable");
}
private static Timestamp convertTimestamp(String asDate) throws Exception {
SimpleDateFormat loSDF = new SimpleDateFormat("MM-dd-yyyy");
java.util.Date loDate = loSDF.parse(asDate);
java.sql.Timestamp loTimestamp = new java.sql.Timestamp(loDate.getTime());
return loTimestamp;
}
public static String getVersionInfo(){
return "$Id: CustomDataSource.java,v 1.2 2007/09/07 13:43:29 Exp $";
```

```
}
}
```

8.3.7 Example: Custom Stem Class Script Rules

The following is an example of how to implement a custom stem class in script rules.

```
package com.camilion.custom.stem;
import com.camilion.base.*;
import com.camilion.data.DataTable;
import com.camilion.util.functions.StaticFunctionLibrary;
import java.text.SimpleDateFormat;
import java.sql.Timestamp;
public class CustomDataSource extends StaticFunctionLibrary {
// We use a singleton so we can have the static methods call instance methods.
// This allows for easier unit testing.

private static CustomDataSource instance = new CustomDataSource();
static String[][] USAGE = new String[][] {
{ FUNC, "getString", "String", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
{ FUNC, "getNumber", "double", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
{ FUNC, "getDate", "Timestamp", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
{ FUNC, "getDouble", "double", "" },
{ ARG, "aoServiceCommand", "ServiceCommand", HIDE + ""},
{ ARG, "asColumnName", "String", "" },
};
public String[][] getUsageArray(){
return USAGE;
}
public static CustomDataSource getInstance()
{
return instance;
}
public CustomDataSource()
{
super();
// TODO Auto-generated constructor stub
}
public static String getString(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable)aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return loTable.getString(asColumnName);
}
public static double getNumber(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return Double.parseDouble(loTable.getString(asColumnName));
}
public static double getDouble(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return Double.parseDouble(loTable.getString(asColumnName));
}
```

```
public static Timestamp getDate(ServiceCommand aoCommand, String asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
String lsValue = loTable.getString(asColumnName);
return convertTimestamp(lsValue);
}
private static void validateDataTable(DataTable aoTable, String asColumnName)
throws Exception{
if ( !aoTable.hasColumn(asColumnName) )
    throw new CsiException("Exception while running getString method",
        asColumnName + " does not exist in the Datatable");
}
private static Timestamp convertTimestamp(String asDate) throws Exception {
SimpleDateFormat loSDF = new SimpleDateFormat("MM-dd-yyyy");
java.util.Date loDate = loSDF.parse(asDate);
java.sql.Timestamp loTimestamp = new java.sql.Timestamp(loDate.getTime());
return loTimestamp;
}
public static String getVersionInfo(){
return "$Id: CustomDataSource.java,v 1.2 2007/09/07 13:43:29 Exp $";
}
}
```

8.3.8 Creating Custom Stem Classes to Handle BigDecimal Type in Script Rules

If you have created custom stem classes to be used in your script rules and are using the `HighPrecisionCalcMode` (parsing double number type to BigDecimal), modification may be required for your methods.

`HighPrecisionCalcMode` off

If you have elected to leave the `HighPrecisionCalcMode` setting as No, then no change is required.

`HighPrecisionCalcMode` on

If you have elected to leave the `HighPrecisionCalcMode` setting as Yes (parsing double number type to BigDecimal) then new methods should be created as follows:

```
BigDecimal method1(ServiceCommand)
BigDecimal method2(BigDecimal, CsiString, ...)
```

These methods return BigDecimal number type and therefore can only be used when `HighPrecisionCalcMode` is turned on. Also, for any String type arguments, the class CsiString has to be used.

It's recommended that you create new methods in this manner after `HighPrecisionCalcMode` is turned on and there is no plan to switch the setting back to off.

To have existing methods work with the `HighPrecisionCalcMode` and use BigDecimal number type, do the following:

Case 1:

Existing method:

```
double method1(ServiceCommand)
```

This method returns double number type, with no double or String type arguments.

System will automatically parse any double number values inside the method to BigDecimal for calculations, and parse back the result to double as the return value. No modification is needed.

Case 2:

Existing method:

```
double method2(double, String, ...)
String method3(String, ...)
```

These methods use double or String type arguments. The `HighPrecisionCalcMode` won't work with double or String as arguments. In these cases, the methods should be overloaded as follows:

```
BigDecimal method2(BigDecimal, CsiString, ...) { //call method2(double,
String, ...) }
String method3(CsiString, ...) { //call
method3(String, ...) }
```

For example:

```
public static double getNumber(ServiceCommand aoCommand, String
asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
return Double.parseDouble(loTable.getString(asColumnName));
}
```

Overload as follows

```
public static BigDecimal getNumber(ServiceCommand aoCommand, CsiString
asColumnName)
throws Exception{
double d = getNumber(aoCommand, asColumnName.toString());
return new BigDecimal(d);
}
```

Existing

```
public static CsiTimestamp getDate(ServiceCommand aoCommand, String
```

```
asColumnName)
throws Exception{
DataTable loTable = (DataTable) aoCommand.get("_DATA_TABLE");
validateDataTable(loTable,asColumnName);
String lsValue = loTable.getString(asColumnName);
return new CsiTimestamp(convertTimestamp(lsValue));
}
```

Modify as follows

```
public static CsiTimestamp getDate(ServiceCommand aoCommand, CsiString
asColumnName)
throws Exception{
return getDate(aoCommand, asColumnName.toString());
}
```

If you would like to create methods that support both the `HighPrecisionCalcMode` off and on there are two options: either have methods that don't use double and String arguments, as shown in Case 1 above; or overload the methods with double and String arguments with BigDecimal and CsiString, as the methods shown in Case 2 above.

> Note
>
> It isn't recommended to switch it off once it is turned on.

8.4 Managing Web Content

All static web content for applications should be deployed to the appropriate subdirectories under `<CSI.home>/ppms/app/pa/custom/web/`. The standard product servlet should be used to download and show this content, as necessary.

Examples

- To download the CSS file Accumulator.css deployed under `<CSI.home>/ppms/app/pa/custom/web/css/common`, use the following URL: `/pa/css/common/Accumulator.css`
- To download the HTML file Client.html deployed under `<CSI.home>/ppms/app/pa/custom/web/html/login`, use the following URL: `/pa/html/login/Client.html`
- To download the png file `logo.png` deployed under `<CSI.home>/ppms/app/pa/custom/web/image/camilion`, use the following URL: `/pa/image/camilion/logo.png`

To upload web content to the development server, you can use the Administrative Console, choosing the *Edit File System* option from the *System* menu. From the *Path* dropdown list, select PA - Custom Web Root.

8.5 Managing Database Content

A development environment typically contains multiple projects that share flowlets in a common database. As the projects/applications progress through the various levels in the development life cycle, they diverge into multiple streams. It can therefore become difficult to identify which artifacts belong to only a singe project, and as a result, any reference tables, packages, and stored procedures used by one application can be mistakenly bundled when packaging other applications.

This section addresses the issue of managing database content by splitting the schemas, as described in the following topics:

- Schema naming conventions
- Stored procedures best practices

8.5.1 Schema Naming Conventions

A schema is a collection of database objects. A schema is owned by a database user and has the same name as that user. Schema objects are the logical structures that directly refer to the database's data. Schema objects include structures such as tables, views, and indexes.

An application can consist of up to seven schemas.

A single development environment often contains multiple applications, as well as multiple copies of the same application that are in different development phases. To avoid naming conflicts, the following elements are included in schema names:

- the company name
- the name of the development life cycle phase
- the project name
- the server type
- the schema type

Thus a schema name consists of: <company name><development life cycle name><project name><server type>_<schema type>.

Each piece of the name gets a three character abbreviation, except for the schema type, which can have an abbreviation of up to four characters.

For example, in a Runtime client (PPMC) server for a project named World Insurance Underwriting (WIU) for the company Insurance Solutions Inc. (INS) the development environment (DEV) schema for system flowstore tables (_SYS) should be named: INSDEVWIUPPMC_SYS

The following table describes all the schema names created for the previous example:

Server Type (User Logon)	Schema Name	Content
PPMC (INSDEVWIUPPMC_SYS)	INSDEV_UDIR	User directory tables shared across all projects in the same company.
	INSDEVWIUPPMC_PPMC	Design Time content such as process definitions, rule definitions, screen lay-outs, etc.,

Server Type (User Logon)	Schema Name	Content
	INSDEVWIUPPMC_PPMS	Product tables that have process in-stance control data.
	INSDEVWIUPPMC_SYS	Tables that are created by the Product Modeler, such as flowstores.
	INSDEVWIUPPMC_USR	Application-specific reference tables, stored procedures, packages, and other application-specific database artifacts.
PPMS (INSDEVWIUPPMS_SYS)	INSDEVWIUPPMS_PPMS	Same as _PPMS above.
	INSDEVWIUPPMS_SYS	Same as _SYS above.
	INSDEVWIUPPMS_USR	Same as _USR above.

All application-specific database content resides under the ..._USR schema, and a private synonym for these tables along with database objects belong to other Schemas are created under ..._SYS schema to access these tables and other application-specific database artifacts.

> ⚠ Caution
>
> Don't give tables the same name as any of the flowstores.

8.5.2 Best Practices for HANA SQLScripts

This topic contains a set of recommendations for using HANA SQLScripts in your applications for the best performance.

For more information refer to the *Best Practices for using SQLScript* and *Developing Applications using SQLScript* sections of the SAP HANA SQLScript Reference Guide.

Avoid Hard-Coding Schema Names

Don't use the literal schema names in your SQL statements, as Schema Name can be different for each deployment.

Minimize Database Calls

The fewer database accesses you require, the more overhead you avoid. Reducing the number of physical trips to the database is particularly beneficial if you are accessing the database across a network.

→ Tip

This best practice brings one of the largest performance benefits.

Use Stored procedures/functions

Use stored procedures/functions to package related SQL statements together and to call the stored objects from your application to reduce the number of trips to the database.

Combine SQL Statements

Combine related SQL statements instead of retrieving values by running one query and passing the values to the second query to retrieve the final values from the database. Performance can also be improved by combining simple and unrelated queries. For example, instead of running the first query and passing the returned values one by one to the second query, combine the queries into one query and run it only once:

Query 1

```
Select order_no into :orders
From ord
Where buyer_id = 'abc';
```

Query 2

```
Select *
From item
Where order_no = orders(i);
```

Combined query:

```
Select item.*
From ord, item
Where ord.order_no = item.order_no
And ord.buyer_id = 'abc';
```

Share SQL Statements

To save memory resources and to avoid expensive parsing, dispatch identical SQL statements to the database. It's always a good coding practice to use the same pattern or format when compiling SQL statements.

Minimize parsing as far as possible. Application developers should design their applications to parse SQL statements once and execute them many times. This is done through cursors. Experienced SQL programmers should be familiar with the concept of opening and re-executing cursors.

Another excellent practice is to share SQL statements by using bind variables in the SQL statements. The values of the bind variables are assigned at Runtime. This is particularly true when the value of the specific column is evenly distributed. For example, instead of using: `Select * From EMP Where EMP_NO = 100;`, write the SQL as: `Select * From EMP Where EMP_NO = :v1;`.

> ⓘ Note
>
> If the value distribution of the specific column is uneven, for example, 70% of the values of a column is 1 and the rest ranges from 2 to 10, it may be more efficient to take advantage of histogram statistics on the column and directly pass the value to the query. In this case, directly pass the value in the query and don't use a bind variable.

Avoid Performing Calculations on Indexed Columns

If you apply a function to an indexed column in the `Where` clause of a query, the database ignores the index and instead performs a full table scan. For example, the following query:

```
Select *
From ord
Where UPPER(buyer_name) = 'ABC';
```

Even though there is an index on the buyer_name column, the database does a full table scan. To take advantage of the index in this situation, you would store values as all uppercase in the database, use a functional index, or if possible, do a case-insensitive search.

Try to avoid using `NOT` or `!=` in the `Where` clause for indexed columns. For example:

```
Select *
From ord
Where line_number != 0;
```

Instead, use the following symbol:

```
Select *
From ord
Where line_number > 0;
```

In the following example, the intention of the `Where` clause is to eliminate the *Time* portion of the *Date* field, but the `TRUNC` function disables the index.

```
Select acct_name, trans_date, amount
From transaction
Where TRUNC(trans_date) = TRUNC(sysdate);
```

To take advantage of the index on the *trans_date* column, you would use the following statement instead:

```
Select acct_name, trans_date, amount
From transaction
Where trans_date between TRUNC(sysdate) and TRUNC(sysdate) +
.99999;
```

At times, as the result of internal data type conversion, the database doesn't use indexes. For example, the data type of the column order_type is varchar2 and it is indexed. Consider the following query:

```
Select order_no
From ord
Where order_type = 100;
```

The index is still discarded, because when the query is run the database internally processes `TO_NUMBER(order_type)`.

Don't rely on automatic data type conversion, and always use a value with a data type matching that of the column.

Minimize Table Lookups in Queries

To improve performance, minimize table lookups in a query, particularly if the statement includes sub-query `SELECT` clauses or multi-column `UPDATE` clauses. For example, instead of specifying the following statement:

```
Update table_A
set col_A = (Select col_A From table_B Where ID=100),
col_B = (Select col_B From table_B Where ID=100);
```

Use the following statement instead:

```
Update table_A
set (col_A, col_B) = (Select col_A, col_B From table_B Where
ID=100);
```

Use the `NOT EXISTS` clause instead of the `NOT IN` clause

To avoid an internal sort/merge and thus improve performance, instead of using a `NOT IN` clause, as in the following example:

```
Select *
From lines
Where order_id NOT IN (select order_id
    From ord
    Where supplier_id = 'abc123');
```

Use the `NOT EXISTS` clause, as follows:

```
Select *
From lines
Where NOT EXISTS
( select 1
    From ord
    Where ord.order_id = lines.order_id )
```

```
Where supplier_id = 'abc123'
```

Use Joins instead of Sub-queries

Wherever possible, use joins instead of sub-queries. For example, instead of writing the following statement:

```
Select
From due
Where EXISTS (select '1'
From ord
Where order_id = due.order_id And order_type = 'abc');
```

You would write the following statement:

```
From due, ord
Where due.order_id = ord.order_id
And ord.order_type = 'abc';
```

Pay Attention to ORDER BY

An `ORDER BY` clause can be very expensive if the data to be sorted is large and not pre-sorted. Instead, it's ideal if you can use an index for sorting. To have an `ORDER BY` query use an index, all columns in the `ORDER BY` clause need to be NOT NULL and there must exist a single index with the same sequence as that in the `ORDER BY` clause. Try to have a query return the data in the correct order without using an `ORDER BY` clause. If a non-unique index with a NULL column exists, a dummy `WHERE` clause can make the query use the index and return the data in the order you want.

Pay Special Attention to `IN` Clauses

Often a database developer uses an `IN` clause in an SQL statement and at Runtime dynamically feeds the elements inside the `IN` clause or uses a sub-query inside the `IN` clause. Keep in mind the potential number of elements in the `IN` clause when the query runs against the production database. If the number is greater than 100, the database optimizer may have problems using an optimal execution plan for the SQL statements.

Controlling the Access Path and Join Order with Hints

You can influence the optimizer's choices by setting the optimizer approach and goal, and by gathering representative statistics for the query optimizer. Sometimes, the application designer, who has more information about a particular application's data than is available to the optimizer, can choose a more effective way to execute a SQL statement. You can use hints in SQL statements to instruct the optimizer about how the statement should be executed.

Hints, such as `/*+FULL */` control access paths. For example:

```
SELECT /*+ FULL(e) */ e.last_name
    FROM employees e
    WHERE e.job_id = 'CLERK';
```

Join order can have a significant effect on performance. The main objective of SQL tuning is to avoid performing unnecessary work to access rows that don't affect the result. This leads to three general rules:

- Avoid a full-table scan if it is more efficient to get the required rows through an index.
- Avoid using an index that fetches 10,000 rows from the driving table if you can instead use another index that fetches 100 rows.
- Choose the join order so as to join fewer rows to tables later in the join order.

The following example shows how to tune join order effectively:

```
SELECT info
FROM taba a, tabb b, tabc c
WHERE a.acol BETWEEN 100 AND 200
AND b.bcol BETWEEN 10000 AND 20000
AND c.ccol BETWEEN 10000 AND 20000
AND a.key1 = b.key1
AND a.key2 = c.key2;
```

The steps followed for the above example are:

1. Choose the driving table and the driving index (if any).
   The first three conditions in the previous example are filter conditions applying to only a single table each. The last two conditions are join conditions.
   Filter conditions dominate the choice of driving table and index. In general, the driving table is the one containing the filter condition that eliminates the highest percentage of the table. Thus, because the range of 100 to 200 is narrow compared with the range of acol, but the ranges of 10000 and 20000 are relatively large, taba is the driving table, all else being equal.
   With nested loop joins, the joins all happen through the join indexes, the indexes on the primary or foreign keys used to connect that table to an earlier table in the join tree. Rarely do you use the indexes on the non-join conditions, except for the driving table. Thus, after taba is chosen as the driving table, use the indexes on b.key1 and c.key2 to drive into tabb and tabc, respectively.
2. Choose the best join order, driving to the best unused filters earliest.
   The work of the following join can be reduced by first joining to the table with the best still-unused filter. Thus, if bcol `BETWEEN ...` is more restrictive (rejects a higher percentage of the rows seen) than ccol `BETWEEN ...`, the last join can be made easier (with fewer rows) if tabb is joined before tabc.
3. You can use the `ORDERED` or `STAR` hint to force the join order.

Combine Multiples Scans with CASE Statements

Often, it is necessary to calculate different aggregates on various sets of tables. Usually, this is done with multiple scans on the table, but it's easy to calculate all the aggregates with one single scan. Eliminating n-1 scans can greatly improve performance.

Combining multiple scans into one scan can be done by moving the WHERE condition of each scan into a CASE statement, which filters the data for the aggregation. For each aggregation, there could be another column that retrieves the data.

The following example asks for the count of all employees who earn less then 2000, between 2000 and 4000, and more than 4000 each month. This can be done with three separate queries:

```
SELECT COUNT (*)
    FROM employees
    WHERE salary < 2000;

SELECT COUNT (*)
    FROM employees
    WHERE salary BETWEEN 2000 AND 4000;

SELECT COUNT (*)
    FROM employees
    WHERE salary>4000;
```

However, it's more efficient to run the entire query in a single statement. Each number is calculated as one column. The count uses a filter with the CASE statement to count only the rows where the condition is valid. For example:

```
SELECT COUNT (CASE WHEN salary < 2000
        THEN 1 ELSE null END) count1,
    COUNT (CASE WHEN salary BETWEEN 2001 AND 4000
        THEN 1 ELSE null END) count2,
    COUNT (CASE WHEN salary > 4000
        THEN 1 ELSE null END) count3
    FROM employees;
```

This is a very simple example. The ranges could be overlapping, the functions for the aggregates could be different, and so on.

Use DML with RETURNING Clause

When appropriate, use INSERT, UPDATE, or DELETE... RETURNING to select and modify data with a single call. This technique improves performance by reducing the number of calls to the database.

Modify All the Data Needed in One Statement

When possible, use array processing. This means that an array of bind variable values is passed to Oracle for repeated execution. This is appropriate for iterative processes in which multiple rows of a set are subject to the same operation.

For example:

```
BEGIN
    FOR pos_rec IN (SELECT *
        FROM order_positions
        WHERE order_id = :id) LOOP
```

```
            DELETE FROM order_positions
            WHERE order_id = pos_rec.order_id AND
            order_position = pos_rec.order_position;
    END LOOP;
    DELETE FROM orders
    WHERE order_id = :id;
END;
```

Alternatively, you could define a cascading constraint on orders. In the previous example, one `SELECT` and `<n>` `DELETE`s are executed. When a user issues the `DELETE` on orders `DELETE FROM` orders `WHERE` order_id = :id, the database automatically deletes the positions with a single `DELETE` statement.

8.6 FS-PRO Configuration Files

To maintain the integrity of the application in different stages of the application life cycle, configuration files should be self-contained. You must separate environment-specific configuration settings from non-environment-specific configuration files.

Maintain environment-specific configuration settings under `<CSI.home>/ppms/app/pa/config`.

Maintain other static (non-environment specific) configuration settings under `<CSI.home>/ppms/app/config`.

> Note
>
> Both these sets of configuration files must have the CONFIG extension.

All configuration entries are loaded into one memory object in the same name space.
The settings for the current enabled application are available to custom classes by calling `com.camilion.env.AppConfig.getInstance().getProperty(<property name>, [<default value>])`.

The preceding method returns a string value.

The following new rule stem is available in the rule language to access any application-level configuration entry `Config.getAppProperty( <Configuration Name>, <Default Value> )`

Related Information

Configuring Servers [page 101]

8.7 Naming Configuration Entries

Entries in configuration files should be prefixed with the appropriate three-letter project code followed by an underscore (_).

⚠ Caution

To avoid accidentally overriding FS-PRO configuration settings, the system ignores any configuration entry that doesn't follow the standard schema naming conventions.

⚠ Caution

Avoid using Yes or No for Boolean values; instead use 1 or 0 as a flag.

Example: `WIU_send_email=0 or 1`, where WIU is the project code.

Related Information

Schema Naming Conventions [page 173]

9 The CSI_HOME Directory

All product and application content resides below a directory called `CSI_HOME`. The directory structure separation for product and application content below the `CSI_HOME` directory makes it easy to bundle content into separate packages.

> **Note**
>
> All servers use the same structure.

The following list describes each subdirectory and its purpose:

env	System-level, environment-specific configuration and artifacts. Also contains log files across all the modules and applications in FS-PRO.
pc	Product Modeler server configuration and other Product Modeler server content.
ppms	Runtime server configuration and other Runtime server content.
ps	Product Services configuration and other Product Services content.
system	System-level product configuration and other system-level artifacts.

10 Data Object APIs

The `Data Object Layer` allows you to model data objects using object modeling techniques including composition and inheritance. In addition, it allows you to provide ORM mapping.

The `Data Object Layer` lets you reuse the rules and Java code operating on abstract objects, saving time and increasing efficiency.

Persistent Objects	FS-PRO allows you to define hierarchical data objects and attributes, and map the data objects and attributes to physical database tables and columns. The `Data Definition` object allows you to configure data object metadata and data object mapping to physical database tables.
Transient Objects	The separation of the business layer from the presentation layer in FS-PRO allows for the development of business layers that fully define a business domain model and a presentation layer that may present the data with varying degrees of denormalization.
ORM Mapping	An object-relational mapping layer is introduced allowing custom code or script rules to retrieve and store objects in a database..

10.1 Using Custom Code to Define a Business Model

Transient objects can be instantiated using custom code, such as a custom class created by you. The custom objects must be configured as transient objects in order to access through rules. For example, entitlements must be modeled as data objects in order to be accessed through rules.

Transient object definitions support composition and inheritance. By using custom code, you can manipulate data objects in the current composition model.

10.2 APIs

The following classes in package `com.camilion.dataobject.publicapi` and `com.camilion.dataobject.service.publicapi` can be used in custom code to manipulate data objects, including create, retrieve, delete, and save:

- public interface DataObject
- public interface DataObjectMetadata
- public interface DataObjectMetadataAssociation
- public interface DataObjectMetadataAttribute
- public interface DataObjectService
- public interface PersistableDataObjectService

- public interface CSVDataObjectService
- public interface URSDataObjectService
- public interface XMLDataObjectService

The `RuleService` class in the package `com.camilion.rule.service.publicapi` provides methods for executing rules using information such as product name, product version, context date, rule ID, and rule arguments.

> ⓘ Note
>
> Transient objects can be initiated by using script rules and custom code.

10.3 External Reference Data Objects

Reference data (read only) may be persisted external to FS-PRO. Rule definitions have access to externally persisted data in a seamless fashion. A new system object, `Reference Data` component, is created out-of-the-box in the Product Modeler's Product Studio.

In script rules, you can query data in a component in the Product Modeler. However, sometimes the desired data may be located externally. The `Reference Data` component allows you to specify custom classes to retrieve external data.

In the product, if a selected data value row in the `Reference Data` component is configured with the component name to look for, and a custom class that contains a query, when the system executes the script rule, it will look for the desired data from the custom class' query instead of from the component. If the data value row isn't selected or doesn't exist, the system will look for the desired data from the specified component.

The following classes in the package com.camilion.publicapi.referencedata let you use custom code to perform queries in data outside of FS-PRO. These classes are used to perform queries in the custom classes referred to by the `Reference Data` component.

For additional information on the data object layer, see the Product Modeler User Guide.

For a full description of the classes and Java syntax, refer to the Javadocs distributed with the FS-PRO package.

11 Change Management

FS-PRO and FS-QUO consist of the following major system groups:

- A Product Modeler Design Time server where business users develop products for use in a Runtime environment.
- An underwriting Runtime environment based on FS-QUO where underwriters write new businesses for products developed in the Product Modeler.

Architecturally, these servers follow the typical Java Enterprise Edition (EE) application architecture, consisting of the following components:

- A Java EE application deployed on SAP HANA XS Advanced.
- An application database created on SAP HANA.
- A home directory (generally known as `CSI_HOME`) located on a file system accessible by the Java EE application and the application server.

To support product development and extension of the underwriting platform, a typical insurance suite solution generally has the following in the system landscape:

- One Product Modeler Design Time server, acting as the central product repository and development platform.
- Multiple lightweight development environments for FS-QUO based on Eclipse and TomEE+, known as the FS-QUO Toolkit, installed on developer workstations for FS-QUO extensions.
- Multiple FS-QUO underwriting Runtime environments to support the typical software development lifecycle through the phases of development and integration, quality assurance (QA), staging, and production.

The topics in this section describe the types of changes that can be made in this solution, and how the changes should be applied and propagated to the systems supporting each phase of the development lifecycle.

11.1 Managing Product Modeler Changes

In a typical FS-PRO and FS-QUO system landscape, there is a single Product Modeler instance acting as the central source and repository for products. Business users develop products in the web-based development tool. When a product is ready to be tested in a Runtime environment, it is published through the Product Modeler. The concept of product publish is explained in the Publishing section of the Product Modeler User Guide.

The Product Modeler provides an Administrative Console for users to configure settings that are both essential and optional for its operation. The settings are stored either in `CSI_HOME` or the application database. Basic settings are discussed in the Installation Guide, while advanced settings are discussed in the Administration Guide.

The Product Modeler itself is a change management system for product configurations. Business users can use the Product Modeler to make and keep track of changes. Details are provided in the Product Modeler User Guide.

As part of the product configuration function, the `Rule Scripting Language` calls custom function libraries (also known as stems), developed as part of customer implementation, which provides additional functionality that isn't available out of the box in Runtime. While FS-QUO requires the Java implementation class of a functional library to be available in Runtime, only its Java interface class is required in the Product Modeler to be made available to the script rule editor and build tools. The following steps are required in the Administrative Console of the Product Modeler to make a function library available:

1. Upload the function library interface class file using the *System* > *Edit File System* and *System* > *Reset ClassLoader* menu options.
2. Register the function library by providing the required information such as class name and library name, as well as clearing the function cache, using the *System* > *Library Manager* menu option.

11.2 Managing FS-QUO Changes

As a general deployment strategy, consolidate all changes as a set and apply to your FS-QUO environment as a single unit.

11.2.1 Source of Deployment Artifacts

There are two major categories of deployment artifacts to FS-QUO:

1. Insurance product binaries from the Product Modeler.
2. Extensions and customizations to FS-QUO for customer implementation.

Insurance Product Binaries from the Product Modeler

When product configuration is satisfactory and ready to be tested in Runtime, the insurance products are built and published in the Product Modeler. The publish process collects all product configuration information and artifacts into a binary format called a product JAR file. The main consumers of product JAR files are:

Product Web Services	Allows exposure of product functions (such as rating) as standard Web Services. Product Web Services are typically run alongside the standard FS-QUO Runtime on the same instance. For more information, see Working with Product Web Services.
FS-QUO runtime	The underwriting platform where underwriters write new businesses for developed insurance products.

FS-QUO Toolkit	The development bootstrap toolkit for creating FS-QUO extensions and customizations.

Extensions and Customizations to FS-QUO for Customer Implementation

The other major source of change for FS-QUO are extensions and customizations to FS-QUO providing functionality required by customers that aren't available out of the box. These can be further categorized as follows:

1. Source code, for example, Java classes, JavaScripts, JSPs and CSS.
2. Process flowstore, which is the definition of the database model storing transactional information captured in the underwriting workflow (for example, producer and contact information).
3. Settings in the Administrative Console, for example, general settings such as login and debug settings, and additional settings to support extensions such as interface touchpoints.

Such changes typically originate from a team of software developers responsible for FS-QUO extension and customizations. FS-QUO prescribes tools and processes in the form of the FS-QUO Toolkit. It enables developers to set up a local development environment consisting of Eclipse as the Integrated Development Environment (IDE), TomEE+ as the lightweight application server, and the FS-QUO binaries and limited set of source code as the Eclipse workspace for customization. With a shared application database created, developers can develop and test all local changes before making them available for deployment to other environments.

While code changes are made within Eclipse, process flowstore changes must be made in the Product Modeler tooling as part of the FS-QUO environment. For more information, see Domain Layers.

In addition, configuration settings such as login and FS-PM integration settings are made in the Administrative Console. Subsequent application and deployment of these changes can only be done manually, therefore they must be documented and made available as part of a change set.

11.2.2 Managing Deployment Artifacts

As explained in the previous section, some deployment artifacts are already in a consumable binary format, while some are maintained as source code and documentation. Regardless, all such deployment artifacts should be source-controlled centrally using a system such as Subversion and source code should be built into binary artifacts as appropriate.

The published product JAR files from the Product Modeler should be checked into the source control system. The same should be done with source code, and developers should leverage the Eclipse features provided by the source control system vendor. Lastly, the documentation on process flowstore changes and Administrative Console setting changes should be consolidated and checked in.

A build process should be defined to consolidate all deployment artifacts into a single package for ease of deployment. It is recommended that you use Apache Ant build scripts, some of which are provided out of the box in the FS-QUO Toolkit projects, in conjunction with a build management system such as Jenkins to automate the build process.

For details, refer to the *SAP Quotation and Underwriting for Insurance Coverage-based Developer Guide* or the *SAP Quotation and Underwriting for Insurance Risk-based Developer Guide*, as applicable.

11.2.3 Deploying FS-QUO Changes

This topic describes the deployment procedure of each type of deployment artifact to an FS-QUO environment.

Keep in mind that although the deployment package consolidates all deployment artifacts, some might not have changed from build to build and therefore can be skipped to save time.

Deploying Product JAR Files

Product JAR files are deployed to a FS-QUO environment either through the remote deployment functionality in the Product Modeler, or by uploading in the Runtime Administrative Console. The former method is recommended for a FS-QUO environment assigned for development, while the latter method is recommended for propagation to subsequent FS-QUO environments in the deployment pipeline such as QA, staging, and production after a sequential verification.

Deploying Product Flowstore Changes

The persistence model of FS-PRO and FS-QUO separates product data from process data. While process data (for example, the underwriting process/workflow) is generally static regardless of the line of insurance product, product data varies by lines (for example, coverages are different between household insurance and commercial automobile insurance). An insurance product created in the Product Modeler contains product data definitions, and must be brought into FS-QUO for Runtime use. As such, in addition to uploading the product JAR files, the product flowstores also need to be synchronized through the Administrative Console.

Deploying Product JAR files to *Product Web Services Manager*

As explained earlier, Product Web Services is another consumer of product JAR files for exposing product functions as Web Services. Product JAR files can be deployed as a post-publishing remote deployment step to the *Product Web Services Manager*, or can be uploaded directly within the *Product Web Services Manager* in the Runtime Administrative Console. Again, the former method is recommended for the development environment at the head of the deployment pipeline, while the latter method is recommended for subsequent environments.

Deploying FS-QUO Source Code Changes

FS-QUO source code changes are generally packages into a custom Multi-Target Application (MTAR) that is deployed to SAP HANA XSA. Since the FS-QUO Toolkit is organized into Maven modules, Maven can consequently used to create the custom SCA file for deployment. For details on how to use Maven to create the SCA file, refer to the Coverage-based Development Guide or the Risk-based Development Guide, as applicable.

FS-QUO source code changes are also packaged into a small set of loose web resource files. These artifacts are uploaded to FS-QUO using the Runtime Administrative Console *System* > *Edit File System* and *System* > *Reset ClassLoader* menu options. The Coverage-based Development Guide and the Risk-based Development Guide describes the upload procedure for different types of extensions and customizations in various topics. Alternatively, Maven can also be used to package these artifacts into a zip file in the FS-QUO-install module. It can then be unzipped onto the `CSI HOME` directory instead of uploading in the Runtime Administrative Console.

If new function libraries/stems are added, they must be registered in the Runtime Administrative Console. For more information, see Registering Stems in the FS-PRO Function Library (for coverage-based products) or Registering Stems in the FS-PRO Function Library (for risk-based products).

Also, as mentioned earlier, FS-QUO source code changes also contain any interface classes for function libraries/stems that must be uploaded and registered in the Design Time Administrative Console as described earlier in this document.

Deploying Process Flowstore Changes

Based on the document provided by developers on what changes are to be made, use the Product Modeler tool to apply the changes and synchronize the process flowstore. You will also need to configure table components and data definitions as necessary using the Product Modeler. See Synchronizing Flowstores [page 81] for more information.

Deploying Administrative Console Setting Changes

Based on the document provided by developers on what changes are to be made, apply the changes in the Runtime Administrative Console and reload the configuration settings.

Related Information

Resource URL Authorization [page 14]

11.3 Updating FS-PRO and FS-QUO with Support Packages

The preceding sections describe the change management process for custom and configuration changes. For on-going maintenance, FS-PRO and FS-QUO are periodically updated with Support Packages (SPs) available on the Software Downloads page of the SAP Support Portal. Follow the applicable *Support Package Upgrade*

Guide to apply the required fixes in each of the application environments in your system landscape. FS-PRO and FS-QUO SPs should be applied on each individual environment in the system landscape, as the above deployment process only covers external changes concerning products, extensions, and configurations. Fixes are generally applied to the MTAR file, the application database and `CSI_HOME`. At times, there are product bootstrap changes to be imported to the Product Modeler and subsequently published and deployed to FS-QUO.

An updated FS-QUO Toolkit is also provided for each FS-QUO SP release. A designated developer or solution architect should first update the environment, merging out-of-the-box code as necessary. The updated workspace should then be checked into the source control system for others to update. For FS-QUO extensions and customizations, it is required to first upgrade the FS-QUO Toolkit so that you can produce the custom FS-QUO MTAR file at the appropriate SP level for upgrading the FS-QUO Runtime environment.

11.4 Changing Application Server Settings

Aside from application changes, it is often necessary to change application server settings for tasks such as performance tuning. Such changes generally differ between environments and are applied separately. Typical changes include JVM heap sizes and other tuning arguments which can be applied to an SAP HANA XS Advanced container either through the command line client tools or by overriding the arguments in the MTAEXT file and redeploying the MTAR file.

11.4.1 Fine-tuning JVM Heap Size Based on Published Products

One of the major factors that will affect performance is the number of products that are published. These published products will appear as product JAR files on the file system. It is the size of the product JAR files that will determine how much JVM memory they will consume. Therefore, as part of the sizing exercise, you should look at each published product JAR to determine approximately how much memory needs to be allocated to the JVM heap. But remember: this is static product memory. The JVM heap size should also include room for transient data.

The formula used to estimate the initial JVM heap size is:

```
((A * 4) + (B  * 10) + (C * 3) + (D  * 0.005) + E + F + G + H)
```

In order to create an estimate, you will need to examine the following objects and determine a value based on their usage in your system:

A = PROD_BAS	This object in memory is configurable in the product cache configuration. This configuration only controls the total number of products that can be loaded into Ehcache at a time. The actual JVM memory usage will depend on the size of each product. SAP has determined, through measurements, the

	product `*_data.xml` file should be multiplied by a SAP Measured Multiplier (SMM) of four to determine the amount of heap memory required for that product. For example, if the product `*_data.xml` file is 15MB, then it will use up 60MB (A = 15 * 4) in JVM heap memory. Due to the varying sizes of products, a single large product could end up using the same amount of JVM heap as ten small ones. This value needs to be calculated for each product.
B = REF_INDEX_DOM_STORE	This object in memory is configurable in the product cache configuration. This cache is only used if the product is configured with reference components. Like PROD_BAS, the JVM usage is dependent on the size of a product. We recommend that this value be set to the same as PROD_BAS's. We have determined that the `reference_index.xml` file should be multiplied by a SMM of ten to estimate the amount of heap memory required for that product. This value needs to be calculated for each product.
C = REF_SMALL_DOM_STORE	This object in memory is configurable in the product cache configuration. This cache is only used if the product is configured with reference components. Unlike REF_INDEX_DOM_STORE, the JVM usage is determined by the total number of maximum elements defined being multiplied by 3MB. 3MB is an average size that was determined through measurements. Each product typically has several thousand REF_SMALL_DOM_STORE elements, most of which are not used frequently or used at the same time. We recommend that this cache be sized based on the total number of possible elements that can be stored at any given moment in the cache. The memory usage calculation for this object is based on Ehcache elements.
D = XPATH	This object in memory is configurable in the product cache configuration. The JVM usage is determined by the number of maximum elements multiplied by 0.005MB. 0.005MB is an average size that was determined through measurements. XPATHs are used by all products, so they are not dependent on the number of products loaded into memory.

	The memory usage calculation for this object is based on Ehcache elements.
E = Other combined Ehcache stores	This represents all of the other configurable Ehcache stores in the product cache configuration. While some of the stores can be configured with large maximum element numbers, their actual memory usage is quite small. Therefore, through measurements, we recommend using an average of 500MB for the initial sizing estimate.
F = FS-QUO Base Memory	This object in memory is defined as the minimum amount of memory required to perform FS-QUO operations. This memory will remain relatively static, over time. This value should be measured and adjusted over time to account for customizations.
G = User Session/Transient Data	This is memory that should be set aside to handle user sessions and transient data. This memory will vary over time as users login, perform actions, and then logout. As a result, this value should be measured and adjusted over time.
H = MSGRATING_GLOBAL_RUNTIME_ENV	This value is only applicable if msgPM rating is used in your solution. The number of elements is configured in the product cache configuration but the actual memory usage per element can be calculated using the msgPM *Sizing Guide*. Questions about msgPM sizing requirements can be sent to the msg.support-pm group at `support-msgpm@msg.group`.

⚠ Caution

You should never allocate a maximum JVM heap size that is equal to the available amount of physical memory on the server. There should be physical memory available for the operating system (the size depends on the operating system) and 1GB should be available for the JVM Metaspace (formerly known as the Perm Size).

→ Tip

Total JVM Heap Size is tuned by setting the number of elements for each of the Ehcache objects listed above. Keep in mind that there is a trade-off between performance and memory: better performance requires more memory. You should also keep in mind that, when sizing the total JVM heap, you must consider transient data. Transient data are things like memory required to load the products (often this requires 3-4 times the projected product heap size), policy data, and other operational overheads. If the transient data space is small, you should expect a lot more garbage collections.

Example: Calculating the Estimate

We will use two products, the Life Capital and Sample Household products that are provided with the software, to illustrate how initial sizing can be calculated.

For our example, the contents of the Life Capital product JAR file will be as follows:

Life_Capital_2009_92H0000S0002_1_0_D.jar file

Name	Size (MB)
<DIR> com	15.20
<DIR> Life_Capital_2009_92H0000S0002	1.20
Life_Capital_2009_92H0000S0002_1_0_D_data.xml	11.50
product_info.xml	0.0005
reference_index.xml	0.50

And the contents of the Sample Household will be as follows:

Sample_Household_Product_1_0_D.jar file

Name	Size (MB)
<DIR> com	12.20
<DIR> Sample_Household_Product	0.25
Sample_Household_Product_1_0_D_data.xml	2.70
product_info.xml	0.0005
reference_index.xml	0.03

Calculating the PROD_BAS value	For the PROD_BAS value, we need the size of the `*_data.xml` file for both products. For our sample products, that would be the `Life_Capital_2009_92H0000S0002_1_0_D_data.xml` file (size: 11.50MB) and the `Sample_Household_Product_1_0_D_data.xml` file (size: 2.70MB). Applying the SAP Measured Multiplier (SMM) value of four, we calculate this value as: **((11.5 + 2.7) * 4) = 56.80**.
Calculating the REF_INDEX_DOM_STORE value	For the REF_INDEX_DOM_STORE value, we need the size of the `reference_index.xml` file for both products. For our sample products, that would be a value of .50 MB for the `Life_Capital_2009_92H0000S0002_1_0_D.jar` file and a value of .03MB for the `Sample_Household_Product_1_0_D.jar` file. Applying the SAP Measured Multiplier (SMM) value of ten, we calculate this value as: **((0.50 + 0.03) * 10) = 5.3 MB**.

Calculating the REF_SMALL_DOM_STORE value	The JVM usage is determined by the number of Max. Elements defined multiplied by 3MB. For our example, we will use a Max. Ehcache Elements (MEE) value of 600. Using the SAP Measured Multiplier (SMM) of three, we calculate this value as: **600 * 3 = 1800**.
Calculating the XPATH value	The JVM usage is determined by the number of Max. Elements multiplied by 0.005MB. For our example, we will use a Max. Ehcache Elements (MEE) value of 10000. Using the SAP Measured Multiplier (SMM) of 0.005, we calculate this value as: **10000 * 0.005 = 50**.
Other combined Ehcache stores	For our example, we will use a value **500MB**.
FS-QUO Base Memory	For our example, we will use a value **500MB**.
User Session/Transient Data	For our example, we will use a value **3000MB**.

To complete our estimate, we need to add our values together to find the sum: **56.8 + 5.3 + 1800 + 50 + 500 + 500 + 3000 = 5912**.

Based on our calculations, we should initially allocate **5912MB** for our JVM heap size.

> → Remember
>
> The above represents the amount of static memory that the products will use if they are entirely loaded into memory. In practice, the number of components loaded into cache at any point in time should be lower as not all components of a product will be used by a given policy. Whenever a product is loaded into Ehcache, `*_data.xml` and `reference_index.xml` are always loaded. The number of components will vary.

11.5 Understanding the Deployment Pipeline

The overall FS-PRO and FS-QUO system landscape can be considered as a deployment pipeline, where changes are applied and verified before they are propagated to the next, more important system in the deployment pipeline. While there would only be one Product Modeler instance for centralized product configuration, there will be multiple FS-QUO Runtime applications to facilitate the development lifecycle, including:

1. Development and integration
2. Quality assurance (QA)
3. Staging
4. Production

These instances are all initially installed by following the installation process defined in the *Installation Guides*. The FS-QUO deployment package is then applied to the development and integration environment, verified, and signed off first. When certain quality criteria are achieved, the same is done on the QA FS-QUO environment, and so forth. Between each deployment, back up the environment (generally `CSI_HOME` and the application database) as a precautionary step.

Each organization generally applies the same methodology, but details may vary depending on needs and resources.

11.6 Enhanced Change and Transport System (CTS+)

The Change and Transport System (CTS) of ABAP has been enhanced so that it can be used for transporting non-ABAP objects as well – known as CTS+ or enhanced CTS.

CTS+ enables you to transport non-ABAP objects, such as Java objects and objects of SAP-related non-ABAP applications, in your system landscape, alongside ABAP objects. The aim is to have one transport tool that supports all development workbenches and applications. You only need access to one tool to execute imports for different back-end systems.

CTS+ can be used together with FS-QUO where you can transport objects from one FS-QUO system to another. For instance, you can export code-based or content-based objects from an FS-QUO build or development system to an FS-QUO test or production system via CTS+ Transport Requests.

CTS+ allows for several administrative tasks within FS-QUO to be executed through exposed Web-based RESTful APIs; permitting you to perform these tasks without going through the Administrative Console. Having such tasks centralized in one change management system allows for improved efficiency and maintainability.

For in-depth details, see the How to Configure SAP Underwriting for Insurance (FS-QUO) for CTS Guide.

12 Unsupported Features

The following features are not supported in SAP Quotation and Underwriting for Insurance, even if they are visible in the user interface or the directory structures.

APIs:

- `importFlowletGroup`
- `importFlowletGroup`
- `buildAndSyncProcessFlowlets`
- `exportFlowletGroup`
- `buildProcessFlowlets`

Administrative Console

- Uploading custom stems to the Runtime Administrative Console

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap