Cisco Prime Service Catalog 12.0 Designer Guide

First Published: 2016-11-30

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Key Terms

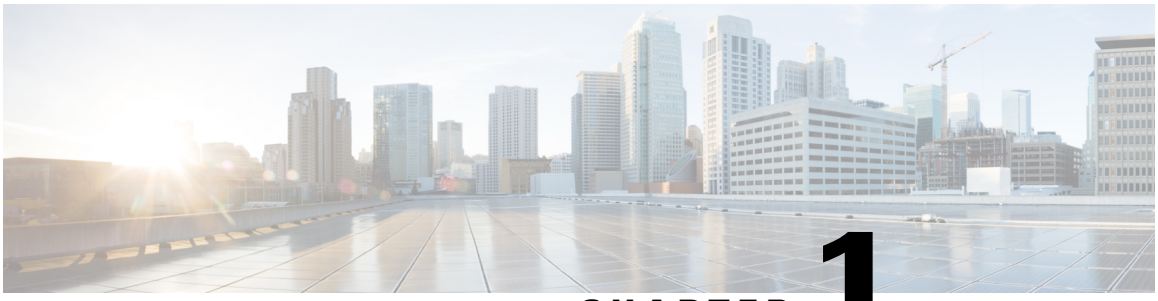
The following are some key terms to be familiar with when using Service Designer.

Table 1: Key Terms Table

Term	Definition
Active Form Components	Reusable forms that are built from one or more dictionaries and configured for use in one or more service forms. Active form components are the building blocks of a service form, and dictionaries, along with active form rules, are the building blocks of a form component.
Authorization	A task during which the performer reviews, approves, or rejects the service requested.
Permission	A Permission grants rights to act upon an object. For example: Order for Others (of a person or OU).
Search Facets	Search Facets allows the service designer to specify one or more facets, which each service can assign a value to. For example, for a “Available Location” facet, a service may have the values, Europe and Americas, while another service has the value Japan. An end user can narrow down the list of available services in their view but select the facet value they want.
Customer	The individual to whom a service is being delivered. The customer and the initiator are typically the same person, except in cases when an initiator orders a service for other people, such as an executive assistant ordering a service for the executive.

Term	Definition
Dictionary	Reusable groups of fields created for use on a form component that may, in turn, be used in multiple service forms. A dictionary defines the individual data items that are used in a service request.
Email Template	A standard email that can be sent upon the initiation, completion or other milestone associated with of a particular task. Email templates can be associated with particular services.
Escalation	Notifications triggered at specified intervals after a task is not completed by its due date.
Initiator	The requestor of, or person who orders, a service from the service catalog. The customer and initiator can be the same person.
Interactive Service Forms (ISF)	A JavaScript API that allows designers to customize the behavior of a service form using JavaScript. ISF coding supplements the use of active form rules to add further interactivity and a richer user interface to the service form.
Moments	Service Catalog manages the events from ordering through service completion as a sequence of discrete system moments or phases. The completion of one moment is the prerequisite for the beginning of the next moment in the sequence.
System Moment	Tracks what point of the requisition life cycle the requisition is in. The system moment evolves from ordering state to finally service completed state, as shown below: Ordering > Pricing > Authorizations > Service Delivery > Service Completed
Service	A process packaged and presented as a product the end user can order/request.
Service Link	The module that defines integrations with external systems; such integrations can be used within a delivery plan as external tasks, reviews, or authorizations.
Service Group	A folder that contains a group of similar services. Services are organized into service groups as a way to facilitate the service design process.

Term	Definition
Service Item	A product or intangible asset that can be provisioned via a service request and whose history can be tracked in the My Services and Service Item Manager modules.
Service Team	The individuals (or groups of individuals) who perform the steps to deliver the service. Service teams are organizational units created and managed in the Organization Designer module. It is critical to specify the appropriate service team for a service group. “Service Team” is listed as a default participant when configuring dictionary Access Control. This allows members of the service team to view or edit dictionaries in the service delivery moment of service fulfillment. All members of the service team are automatically able to perform work on all tasks defined in services in their service group. Other service teams need to be listed as “Additional Participants” in the Access Control subtab for the form components used in the service in order to perform tasks in the service.
Functional Position	Functional positions are associated with service groups and their member of the specified service team who is currently assigned to that position. A functional position is a job description associated with one of the following: • Organizational Unit • Service • Service Group In Service Designer module, you may assign functional positions to be performers of activities in the authorization, review, and delivery processes to avoid referring directly to people or queues. You may also use functional positions to identify the recipients of escalation notifications. For example, an email may be directed to the “Escalation Manager” for a particular organization or service group, rather than being routed to a specific person or queue. Or you may simply use some of the functional positions to document the person or other entity responsible for a particular service group or service.



CHAPTER 1

Introduction

This chapter contains the following topics:

Introduction

Cisco Prime Service Catalog is a self-service portal that allows:

- Users to order new IT services or modify existing services while ensuring compliance with defined IT policies and governance.
- Organizations to encourage adoption of standardized services and implement lifecycle management with governance across internal services such as private cloud services, collaboration, mobility (BYOD), desktop computing, and external services.
- Pay-per-use metering using the tracking capabilities and also allows implementing simple show-back or a more complex charge-back approach.

Overview of Service Designer

Service Designer module is a Prime Service Catalog module that enables service designers to design and package services as products, and offer these services for end users to browse through and order.

Using Service Designer, you can:

- Create categories and keywords that customers may use to search for a particular service
- Design the look and behavior of service forms, the interactive web page, using which service requisitions are ordered and tracked in Service Catalog.
- Construct request or service fulfillment plans
- Configure authorization flows from a service delivery perspective
- Configure service ordering permissions
- Link email templates with processes that require email notifications

The below table describes the components of the Service Design module.

Table 2: Service Designer Module Components

Component	Use this component to:
Services	• Create and modify service groups and service definitions, including the delivery (fulfillment) plan and presentation of the service, as well as the active form components used by the service form. • Configure service order permissions. • Configure authorization flows from a service delivery perspective. • Link email templates with processes that require email notifications.
Dictionaries	Include a group of individual data elements (fields) that allow users and service performers to enter and view data required to fulfill the service request. Dictionaries are a basic building block of a service form. Create and modify the dictionaries that specify the data fields required in a service.
Active Form Components	Create and modify reusable form components, which specify both the service's look (via the configuration of previously defined dictionaries) and feel (via the definition of rules which can dynamically adjust both the form's appearance and behavior). The appearance and behavior of a service form is determined by how the dictionaries and their component fields are configured as part of the *active form components* that are used in the service definition. Active form components provide the potential for *reusability* across service forms. With careful and thoughtful design, a designer may create an active form component from a commonly used dictionary, or set of dictionaries, and configure them only once. Then this form component can be included in as many services as necessary, with no additional configuration.
Scripts	Write JavaScript functions to supplement the rules defined in active form components and maintain JavaScript libraries.

Component	Use this component to:
Categories	Specify how services and service categories are displayed in Service Catalog module. Customers may use categories to search for a particular service.
Keywords	Define and manage the keywords used in the service catalog search engine. Customers may use keywords to search for a particular service.
Objectives	Define and manage the measurable service delivery objectives defined in Service Definition Offer tab.
Extensions	Define custom attributes for services and categories and manage the presentation of categories on the landing page of Service Catalog module.
History	Track the service design change history and view the details based on the entity type or filter by the user name.

All of the above may not be visible to all users of Service Designer. The components you see when you choose the Service Designer module correspond with the role you were granted in Organization Designer. For more information, see the Organization Design chapter of the Cisco Prime Service Catalog Administration and Operations Guide

What is a Perfect Service?

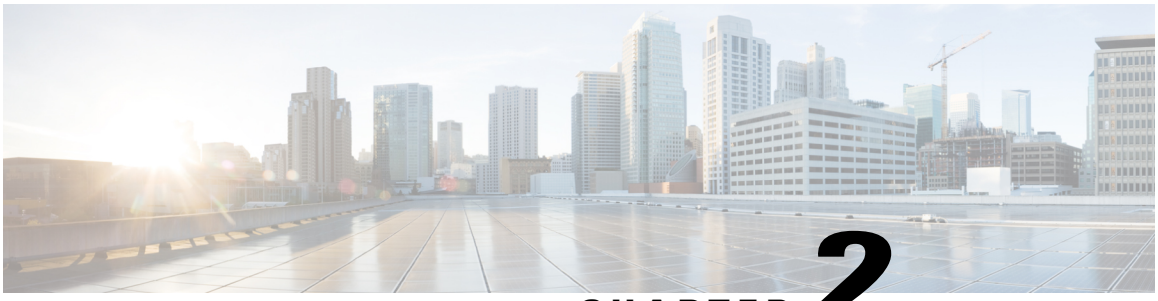
A perfect service should clearly communicate what the service is and what the expectations surrounding service delivery are at each stage of the process. Typically, it should answer these:

- What am I ordering in this service?
- What is included?
- Do I need this service? (Or, Is this the right service for me?)
- How do I order this service?
- How long will it take to be delivered?
- How much does it cost?

The service may also include more detailed information such as:

- Additional task instructions to the service delivery provider
- A checklist of sequential tasks that must be completed in order to fulfill the service request
- Safeguards for when services are delivered late, using conditional statements

End users should be able to rely on the information they see in Service Catalog. Likewise, the service definition and expectations should be equally clear to the service team who will receive the service request and deliver the service.



CHAPTER 2

Preparing to Design Services

This chapter contains the following topics:

Preparing to Design Services

This chapter covers the overview steps for designing services.

Defining the Service Taxonomy

Define the taxonomy of a service before you start designing a service. Consider the following before creating services in the service designer module:

1. Decide what services to offer to an end user in the service catalog
2. Standardize these services
3. Create descriptions
4. Define request fulfillment workflows and the required user inputs
5. Arrive at a portfolio of services that can be published in the service catalog.

For example, a services such as desktop, laptop, keyboard could be a part of the New Hire On-Boarding category or Desktop Service category. As the requester clicks on a category, the application drills down to display subcategories or the services associated with the current category.

Get more information about the services you want to offer.

- What services you will provide?
- Who provides these services?
- Who consumes these services?
- What approvals or reviews are needed before users can purchase services?
- Who is authorized or required to approve or review service requisitions?
- What policies (global or local) govern these approvals and reviews?
- What service levels govern the delivery of these services?

- Is this service delivered by orchestrating one or more API calls to a domain manager system? If so, what are the APIs, and required parameters?
- Who performs these delivery activities?
- Who manages the service delivery?
- What happens if a service delivery is late or interrupted?
- What information must the consumer supply in order to purchase a service?
- What information the service delivery team needs from the consumer?
- Are there any existing data sources that you need to include?
- Are there any existing forms that you can use as models for order forms?
- What assets or items will the services deliver? Will you want to track these items after the services have delivered them?
- How do you want to present the services to the consumer?
- What is the best way(s) for consumer to locate the services in the catalog?

Setting Up Services

The table below summarizes the tasks involved in setting up services. After you create the basic services, you may perform the additional tasks to enhance a service form or customize the appearance of the service form. These are summarized in Table 4: Additional Service Design Tasks table.

Table 3: Designing Services Steps

	Task	Sub Tasks	References
1	Analyze business requirement to define the taxonomy of services	-	Planning a Service Design
2	Design your organization, create people, functional positions, configure user roles, grant permissions	If you plan to use the roles defined in your LDAP, you should integrate Cisco Prime Service Catalog with LDAP.	• *Cisco Prime Service Catalog Integration Guide* • *Cisco Prime Service Catalog Administration and Operations Guide*

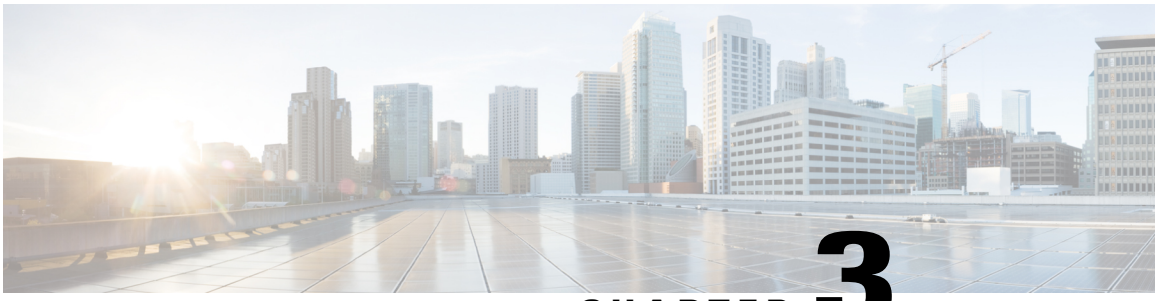
	Task	Sub Tasks	References
3	Configuring Service Items and Service Categories	Configuring Categories for grouping services on the Service Catalog Storefront module. Configuring Service Items. Configuring Service Item Policy. Define actions governed by policies that will be executed against a service item instance when certain conditions are met Associating Services for a Service Item.	Configuring Categories and Service Items for Services
4	Designing Form for Service Request	Configuring Dictionaries Design an Active Form Component that includes one or more dictionaries. Define Active Form Rules and standards for dynamic form behaviors or to extract data from standard tables.	Configuring Forms for a Service
5	Designing Services for Requesting Service Items	Configuring Services for Delivering Service Items Configuring Bundles of Services for automatically ordering services when a customer orders the bundle (parent). Defining permissions allowed to order a service	Configuring Services and Service Bundles
6	Configuring Additional Attributes for Services	Formatting the service presentation Defining service objectives Configuring Keywords for Search	Additional Configurations For Designing Services
7	Designing Plan for Delivering Services	Configuring Delivery Plan (workflows) that contains one or more tasks that must be completed to deliver a service to a customer. Configuring authorization, review, and escalation tasks for a service.	Designing Plan for Delivering Services

	Task	Sub Tasks	References
8	Configuring Rates and Accounts for Billing	Configuring Pricing Configuring Billing rates, User Accounts, and Agreements for Billing transactions	Configuring Rates for a Service Item
9	(Optional) Configuring Services for Communicating with External Systems	If you are designing a service that communicates with external third party systems such as VMware vCenter, or Amazon EC2 directly or through CPO, you will need to: • Configure an external service item task • Configure an agent	• External Dictionaries • Managing Service Items on an External System • Configuring an External Task • *Cisco Prime Service Catalog Integration Guide*
10	Managing the Services and Its Attributes During Service Requisition	Managing the Service Items During the Service Requisition Retrieving History, Subscription Information from the History Managing Order-On-Behalf Service Request	Managing the Services and Attributes

Table 4: Additional Service Design Tasks

Workflow	Reference
Define custom attribute for services and categories to manage the presentation of services and service categories on the landing page of Service Catalog module.	See Extensions in Configuring a Category
Add JavaScript programming to a service form	Interactive Service Forms (ISF) API Overview
Associate the Service with a service item. This allows users and managers to quickly see what services are available for reconfiguring or otherwise adjusting service items previously provisioned for them.	Associating Services for a Service Item

Workflow	Reference
Use Service Item Manager's Import feature to import externally defined service items and standards into Service Catalog, or use the Service Link file adapter to import data for service items or standards.	• Importing Service Items from an External System • Importing Standards
(Optional) Design Portlets and Portal Pages for Service Portal. Designing and managing pages and Service portal content. Creating portlets from external or third-party sources, Creating portlets to highlight common services Creating portlets to show users what they already own, with links to services related to those items	Configuring a Billable Rates
Export/ Import a Service from development to testing to production environments.	Exporting and Importing a Service, on page 251
Backup or migrate your service design to another environment for testing or user acceptance, see Chapter, Managing Content Deployment in Cisco Prime Service Catalog 11.0 Administration and Operations Guide.	Cisco Prime Service Catalog Administration and Operations Guide



CHAPTER 3

Configuring Categories and Service Items for Services

This chapter contains the following topics:

Configuring the Service Catalog Home Page

Configuring the Service Catalog Home page requires various sections to be configured such as Top-left, Top-center, Middle-Left, Middle-Right, Bottom-Left, Bottom-Center, and Bottom-Right.

- **Top (Left and Center):** In this section you can provide the general information. This section doesn't have the services or the categories associated with it to display them.
- **Middle (Left and Right):** This section is for displaying the Categories and Sub-Categories . You can associate the services only with the sub-categories and these can be found by navigating the tree.
- **Bottom (Left, Middle, and Right):** This section is for displaying Categories and its list of services. You cannot define or associate any sub-categories in this section.

You can also configure the Browse option on the Service Catalog Home Page.

To configure the Service Catalog home page, you need to create and configure various categories and subcategories . When the Service Catalog module is enabled, the showcase sections on the home page remains empty until you have defined and configured the categories and services to be displayed on it. For more information on configuring categories, see Creating Categories. Additionally, while configuring the categories, you need to define the location where the information is displayed in the carousal on the Service Catalog Home Page. For more information on showcase location, see Extension Tab Attributes table in Creating Categories.

Configuring Categories and Service Items for Services

Services are designed in Prime Service Catalog by service designers and made available in the Service Catalog portal for an end user to order. This chapter describes the service designing tasks like creating categories, associating categories to the services, configuring a service item to manage the life cycle of a service, and so on.

This chapter includes configuring categories, service items, and standards which are basic building blocks for designing the services.

Categorizing Services

Categories on the service catalog homepage helps customers find a service that meets their needs. The category hierarchy is completely user-configurable. Services can be associated with multiple categories. Categories sub module in Service Designer module is used to create and manage categories and category structure in your catalog.

When the Service Catalog module is enabled, the showcase sections on the home page remains empty until you have defined and configured the categories and services to be displayed on it. In addition, the search facets as filters for narrowing down search results can also be defined, along with the custom attributes, known as Extensions, in the Service Designer user interface. The attributes configured in the Extensions component are automatically displayed in the Extensions tab of the Services component for all services. The values assigned to the extension attributes are accessible through nsAPI GET service definition operations.

Categories can be navigated through the top-left category browser or the middle carousel sections where key categories are showcased. In the figure below, there are services grouped using categories.

Creating Categories

When you create a new category it is automatically added to the “Consumer Service Catalog” root category.

Note Ensure that you remove the category from the root category for the catalog if needed, and associate it with appropriate categories and subcategories for your catalog.

Step 1 Choose **Service Designer** > **Categories**.

Step 2 Select **New** > **New Category** and enter the information required for the new category.

- In the **Name** field, there is a limit of 60 characters.
- In the **Description** field, there is a limit of 255 characters.

Note Do not use **or from URL** field as you cannot add a category description from a URL. Instead use category’s Presentation tab for descriptive information for a category.

Step 3 Click **Add This Category** to create a category. After the category is created, you can define its visual presentation, and configure its other attributes. For configuring categories, see Configuring a Category.

You can delete a category if you no longer need it by choosing the category and clicking **Delete**.

Configuring a Category

Using this procedure, you can set display options, determine the visual presentation, and customize the attributes of a category.

Before You Begin

Create a category or select an existing category. For information, see Creating Categories.

Step 1 Choose **Service Designer** > **Categories**.

Step 2 Select a category from the Categories list pane and customize using the following tabs:

- Use the General tab to determine how categories and subcategories display in the Service Catalog and click **Save**. Selections made on these category tabs are visible within My Services. For details on each field, see Table 5: Field Configuration for Service Designer Categories General.
- Use the HTML tools on the Presentation tab to format how your category displays in My Services. After configuring the presentation of the category click **Save**. For details on each field, see Table 6: Field Configuration for Service Designer Categories Presentation.
- Define custom attributes for categories, in the **Extensions** tab. For details on the configurations done in this tab, see Configuring a Category. The attributes configured in the Extensions component are automatically displayed in the **Extensions** tab of the Services component for all services. The values assigned to the extension attributes are accessible through nsAPI GET service definition operations.

Note You can view the **Extensions** tab only if an administrator has enabled the Service Catalog feature in the **Administration** module.

Step 3 To preview the changes, do the following:

a) Save changes to the categories and associated services and update the page.

b) In the other browser session, refresh the My Services or Service Catalog page display to see the new category hierarchy.

Table 5: Field Configuration for Service Designer Categories General

Field	Definition
Name	The name of the category.
Description	Descriptive text to describe what kind of products or services can be found within this category. This description displays with the category link in the catalog.

Field	Definition
Display Style	Use the drop-down menu to choose whether to: • Display categories only. • Display categories and subcategories under them. If you display subcategories, the category appears in **My Services** module. The portal also displays all of its subcategories as bullet points with active navigation links directly beneath each bullet in the display.
Appears in categories	A list of the “parent” categories in which this category appears. To add this category to more categories, click **Include in More**. To remove this category from other categories, check the check boxes and click **Remove**.
Subcategories in this category	A list of the subcategories or “child” categories which appear in this category. To display more subcategories in this category, click **Display More Categories**. To remove subcategories from this category, check the check boxes and click **Hide**.
Services in this category	A list of the services which appear in this category. To display more services in this category, click **Display More Services**. To remove services from this category, check the check boxes and click **Hide**. You can also add a service to a category from the General tab for that service in Service Designer.

Table 6: Field Configuration for Service Designer Categories Presentation

Field	Definition
Select an Image to Insert	• To add an image from the list of images currently available in the system, choose the image in the list by clicking its radio button, and then click **Add Selected Files**. • To add a new image, click **Browse** to locate and choose the image, click **Attach** to upload the image to the system, choose the image in the list by clicking its radio button, and then click **Add Selected Files**. • To add a new image, provide a URL reference to the image rather than attaching the image directly. This method will help you to manage the images separately, or host it on a content delivery network (CDN) to provide faster image delivery for WAN deployment. Images which represent categories can be in JPG,TIF, PNG, or GIF format. SVG image type is also supported when the category is displayed in the Service Catalog module. By default, images are resized to 50 x 50 pixels in the **Service Catalog** or 64 (width) x 57 (height) pixels in **My Services** module. You can change these pixel sizes by using a custom style sheet (CSS). **Note** The type of images that can be attached to a category or a service is defined in **Administration > Security** option. You cannot attach an image until this option in the Administration tab is set. Category images should be created to the default (or custom) size and aspect ratio. You may use any image editing software to ensure that the size and aspect ratio of the image matches those dimensions. If the image is not sized correctly, it is resized by the browser and may appear stretched, pixelated, or distorted. Depending upon how you have customized Service Catalog using the custom CSS capabilities, you might also want to consider developing custom icons with a transparent background, or a border.
Category Details	Choose whether you want to show (and format) content in the Top, Middle, or Bottom of the category display area in the catalog by clicking the **Show** radio button for the section. If you choose to show multiple sections, click the section name (Top, Middle, or Bottom) you want to format.

Field	Definition
Top URL	Enter a URL to display within the section. The URL that you enter here must be fully qualified, beginning with “http://”.
HTML Editor	Use the included HTML editor to enter and format text or graphics (this is chosen by default). Or, to insert HTML code, click **Source** to disable the HTML editing tools and enter your coding directly. Click **Source** again to return to the HTML editor to display the rendered HTML code.

Table 7: Extension Tab Attributes

Attribute	Description	
Category Attributes		Select **New > New Category Attribute** Enter the following details: • ◦ Display Name ◦ Name ◦ Attribute Type Click **Add This Category Attribute**

Attribute	Description	
Showcases	You can use the fields in the **Showcases** attribute to define the categories that must be displayed in the Service Catalog home page.	**Browse Categories**: Select categories that must be displayed in the **Browse Categories** tab. Select a category that has subcategories associated with it. The application does not display any categories when you hover over the Browse Categories tab if you have not selected a category that has subcategories and services associated with it. **Top-Left, Top-Center**: Select categories that must be displayed in the Heros band. **Middle-Left, Middle-Right**: Select categories that must be displayed in the secondary carousel band. The sub-categories and services associated with this category can be navigated using the carousels. **Bottom-Left, Bottom-Center, Bottom-Right**: Select categories that must be displayed in the bottom panel. The subcategories and services associated with this category are listed in the band.

Attribute	Description	
Search Facets	You can configure Search facets to define multiple filters for search criteria. You can further configure multiple values for each search facet such that the application accepts one or any of those values only. Prime Service Catalog provides the Rating facet out-of-box that cannot be edited. This means that additional values cannot be added/modified to this facet . Using this facet, users can filter offerings based on ratings. The Ratings facet cannot be associated to a service, but whenever a review changes a facet is automatically associated to the service with the corresponding values. This Facet is hence hidden in the **Extensions** Tab in Service Designer module. These out-of-box rating facet cannot be imported or exported to an external system through Catalog Deployer. **Note** The rating facet cannot be localized. For example an operating system search facet can have values such as Windows 7, Windows 9, Linux, and Mac OS. The user must provide any or all of the above values only. Navigate to a category that has services associated with it in the Browse Categories tab in the Service Catalog home page. The Narrow Results column on the left-hand side of the Service Catalog page displays the Search Facets that you configured.	Select **New > New Search Facet** Enter the following details: • ◦ Display Name ◦ Name ◦ Data Type ◦ **Multivalue**: Select this check box to specify if the search facet has to accept any or all of the facet values ◦ **Facet Values**: Click **Add Facet Value** to define multiple values. Click on the check box to mandate a facet value to the Search Facet. Click **Save** to save the configuration.

Attribute	Description	
Service Attributes	You can customize the service definition by defining your own service attributes. For example, you can create attributes to customize the display of a service using an image, or a video, or even display some data like the review date or replacement offer.	Select **New > New Service Attribute** Enter the following details: • ◦ Display Name ◦ Name ◦ Data Type ◦ **Type**: Is a predefined type that displays the attribute value. Click **Add This Service Attribute**

Removing Categories, Subcategories, and Services

You may need to remove the categories in which a category displays, the subcategories that display in a category, or services from a category.

To remove, categories, subcategories, and services:

Step 1 Choose **Service Designer** > **General tab**and choose the **Categories** component.

Step 2 Click the name of the category for which you want to remove categories, subcategories, or services. The details for that category display on the right in the Category window.

Step 3 To remove a category in which this category displays:

- Check the check boxes next to the categories you want removed.
- Click **Remove**.

Step 4 To remove a subcategory from this category:

- Check the check boxes next to the subcategories you want removed.
- Click **Hide**.

Step 5 To remove a service from this category:

- Check the check boxes next to the services you want removed.
- Click **Hide**.

Step 6 Click **Save** to save your changes.

Defining Service Items for a Service

Configuring a Service Item during service design enables you to manage the lifecycle of a service. The life cycle starts at the time the service item is available as a corporate asset, to when it is provisioned through a service request to a Service Catalog user, updates or changes to the item to when it is no longer allocated to the requesting user and becomes available to be reassigned.

For example, service item may be physical, such as a physical hardware device (for example, a cell phone or data center server), or it could be creating a service such as ordering a VM using Service Designer, which ends as soon as the service request is complete and delivered to the end user. You could create service item and design it such that the service also enables you to get notification when the VM is delivered.

A service request may deliver multiple service items, particularly in the case of an employee on-boarding service, which may provide a cell phone, a laptop, a network login, email access, and multiple software applications.

Service Item Manager is a module of Service Catalog that allows users to design service items and also manage the service items that have been ordered from service catalog.

Service Items and Standards Database Tables

All data and configuration details relating to service items are maintained in the Metadata Repository (MDR). The MDR is implemented as a set of database tables within the Service Catalog transactional database.

Each service item type has a corresponding table in the transactional database that stores service item instances. The name of the table is the service item name as specified when the service item is defined plus the prefix “Si” for “Service Item”. Once a service item has been saved, its name cannot be changed. The display name can be modified. However, any dynamic data retrieval rules based on that service item would be invalidated.

Each attribute of the service item corresponds to a column in the database table. In addition to columns corresponding to the attributes specified for the service item, Service Catalog adds three columns used for processing and retrieving service items

Advantages of using Service Item Manager within Cisco Prime Service Catalog over Configuration Management Databases (CMDBs):

- When delivering a service to an end user, delivery performers often need the context of what other service items that user already has. Looking that up “from the CMDB” requires access to several disparate systems and the knowledge to traverse them to find this information, which makes it complex to maintain and use.
- The Service Item Manager allows you to configure the service form such that end users easily find the service items they have been provisioned for. Service Item Manager enables end users and their managers to find the service items and the associated service items and potentially resolve discrepancies, if any, with the items (according to asset systems or CMDBs) in the service catalog module without any pre requisite knowledge CMDB, and its constituent parts.
- Further, even companies with mature CMDBs have some items that are either not tracked at all (neither in a CMDB *nor* an asset system), or have incomplete information, making the job of managing them

difficult. For example, software licenses are not the types of items typically stored in a CMDB. Likewise, cell phones may very well be tracked, but perhaps in an existing system that does not record the business context data or status of the service item that is truly needed for delivering follow-on services to end users.

- In cloud management, the generation of new VMs, infrastructure elements, update of status of these objects, happens at a high rate. CMDB and traditional ITSM tools imposes change control that slow changes down, and often doesn't have the performance characteristic to handle high change rates. This is in contradiction of the requirement of cloud. In Prime Service Catalog, Service Items can be specified as an object type managed within the service catalog to handle these high change rates.

Role to Access Service Item Manager

Site administrators can use the Organization Designer module to grant user access to Service Item Manager and the Administration module to configure how end users can access service items in My Services.

To define a service item, you must have Service Lifecycle Management roles assigned using Organization Designer.

Roles and capabilities, managed with the Organization Designer module, allow site administrators to control access to functionality provided by the Service Item Manager module. The Service Item Manager role will allow administrators to view the service items assigned to all Service Catalog users.

The Service Item Manager roles should be assigned sparingly. Granting a user the capability to manage standards data or define standards means that user can edit all standards definitions and all standards data. A similar caveat applies to the capabilities controlling service item definitions and data.

For more information, see Capabilities for Service Item Manager section in Cisco Prime Service Catalog Administration and Operations Guide

Configuring Access to My Service Items

Access to the Service Items tab and Service Items portlet in My Services can be controlled in three ways:

- Global access for all users must be turned on via an Administration setting.
- Once the ability to view Service Items has been activated, users with appropriate roles are able to view the My Items portlet and the Service Items tab.
- Individual users can choose to hide the portlet via a Profile setting.

Administration Setting for My Items Portlet

The My Items Portlet is no longer necessary in the **Service Catalog** module. My Stuff replaces it with richer feature set .

However, if you want to show My Services module's My Items portlet to a user, the Administration module setting "View Service Items Portlet" must be turned on. To review or modify this setting, choose **Administration > Settings > Customizations**. Look in the My Services section for the "View Service Items Portlet" setting.

Turn this setting on to allow users to see the My Items portlet on the My Services home page, or off to omit the portlet from the page.

Roles with Service Item Access Capability

RBAC capabilities control the access to the service item information for an individual. Roles are also available out of the box for users who need to have such access:

Role	Description
My Services 360-Degree Consumer	Enables end-users to access those portions of the Service Catalog to which they are entitled, to view service component details or initiate requests for services, and to view and manage the service items delivered to themselves as part of catalog services.
My Services 360-Degree Professional	Enables end-users to access those portions of the Service Catalog to which they are entitled, to view service component details or initiate requests for services, and to view and manage the service items delivered to anyone in their business units as part of catalog services.

The My Items portlet and Service Items tab only appear if the end user has been granted the “My Services 360-Degree Consumer” or “My Services 360-Degree Professional” role or any custom role that contains the following permissions:

- Read- All SI instance data subscribed by self.
- Read- All SI instance data for SIs in my business units
- Read- All Instance data fro All SIs in my Business Units and their Sub units.

Creating a Service Item Group

A Service Item group is a logical grouping of various service items. You must create a service item group to add related service items. For example, create a service item group called hardware to be able to add various service items like laptop, mouse, and so on.

Before You Begin

You must have Service Lifecycle Management roles assigned and appropriate permissions to access Service Item manager.

Step 1 Choose **Service Item Manager** > **Design Service Items**.

Step 2 Click the plus sign (+), and select **Service Item GroupService** Item Group.

Step 3 Enter a Name and Description for the service item group and click **Add**.
The Service Item Group that you created is displayed as one of the groups under Service Items section on the Design Service Items page.

Creating Service Items

If you are a service designer or an administrator, you have to first create service items, create services that delivers them, and then make these services orderable. The service items are displayed in the Prime Service Catalog module home page. Users view the list of service items and order them as required. The service item is provisioned via a service request to a Service Catalog user.

Service items can be added to the Service Catalog repository in several ways:

- A user requests a service which provisions a new service item via a Service Item Task. For more information, see Defining Service Item Task in a Delivery Plan and Configuring an External Task .
- The Service Item Import utility is used to import service item definitions and instances. For more information, see Importing Service Items from an External System and Importing Standards.
- A Service Link file adapter is used to import service item definitions and instances. For more information, see Importing Service Items from an External System and Importing Standards.
- A service item administrator manually adds the item through the Manage Service Items tab. For more information, see Managing Service Item Attributes.

Before You Begin

Create Service Item group. See Creating a Service Item Group.

Step 1 Choose **Service Item Manager > Design Service Items.**

Step 2 Click the plus sign (+), and then select **Service Item**.

Step 3 In the Add New Service Item popup window, enter the required information for the new Service Item. Then fields are described in the table below.

Table 8: Service Item Attributes

Field	Description
Name	Unique Display Name, and a Name for the service item. • ◦ Once a service item has been created, its name cannot be changed. You can change the display name of the service item. ◦ The Display Name is the user-friendly version of the name; it may contain spaces. The display name is used in the My Items Portlet and as the basis for the caption for a Service Item-based dictionary. ◦ The Display Name must be unique within a site. ◦ The Name field is the name by which the system references the service item and its data. The item name can contain only alphanumeric characters and the underscore (_), with no embedded spaces. It must begin with an alphabetic character. It corresponds to a table that is dynamically created and maintained in the Service Catalog transactional database. Service Item Manager creates the database table with the same name as the item name with a prefix of “Si”.

Field	Description
Service Item Group	Associate the service item with one of the groups.
Has Status	When this option is checked, a service item attribute called status is created automatically. **Note** When this option is enabled, the "Status" attribute is system-controlled and is linked to other service item properties such as Operations. Therefore, you can delete a status only when there is no operation associated with it. To delete a status just uncheck the **Has Status** check box. When you delete status for a service item, the application: • ◦ Deletes the list of statuses defined for the SI ◦ Deletes status and next status associations ◦ Deletes status and associated services associations
Hide in Service Catalog	Check the **Hide in Service Catalog** check box , if you want to hide the Service Item and the associated request in Service Catalog > My Stuff page.
Custom View URL	Enter **Custom View URL** for a service item so that when a service item is selected in the left pane of Service Catalog > My Stuff, the corresponding right pane displays the content of the custom URL. This is an optional field. For more information see, Adding a Custom Web Page For Displaying Service Items

Step 4 Click **Add** and the Service Item that you created is displayed in the Service Items panel on the left under associated group.

Step 5 On the bottom of the page, click **Add** to add the attributes to the Service Item, as shown below. Each attribute, or row, is a service item instance. The Display Name for each instance will become a column header for the grid.

Step 6 On the bottom of the page, click **Save** after you have added all the attributes.
To further configure the service item, see Configuring Service Items.

Configuring Service Items

After you create a service item, you should configure the service item such that the workflow of the service item is managed automatically.

- Define the service items that can be displayed on the Service Catalog page, or create a separate web page to display list of service items on the Service Catalog homepage. See Configuring Service Item Definition.
- Associate multiple services to a service item for displaying associated services when a service item is chosen for ordering. See Associating Services for a Service Item.
- Define the actions to be executed against a service item instance when certain conditions are met. See Understanding Service Items Policies.

- Create custom-defined statuses for the service item. See Configuring Custom-Defined Status For a Service Item.
- Ensure only a certain state transitions occur by defining finite state transition for a service item. See Defining a Finite State Transition for a Service Item.
- Define permissions on which service item types and instances the user can access and which actions (create/update/delete/view) are allowed. See Defining Permissions to Perform Operations for Service Item Types .
- Displaying Out-of-Box service item attributes on My Stuff page in the Service Catalog. See Customizing My Products & Services Page.

Configuring Service Item Definition

After Creating Service Items, customize the service item definitions using the below procedure.

Procedure

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Service Item Definition.**

Step 2 Enter the required information for the service item definition.

- Check the Hide in Service Catalog option, if you want to hide the Service Item and the associated request in **Service Catalog > My Stuff** page.

Note Use this option to hide service items that are meant as infrastructure data elements used in the service provisioning workflow.

- Enter **Custom View URL** to display a web page you have designed to show the Service Items on the **Service Catalog > My Stuff** page, the corresponding right pane displays the content of the custom URL. This is an optional field. For more information see, Adding a Custom Web Page For Displaying Service Items .

Note **Hide in Service Catalog** and **Custom View URL** features are useful if the service item type is maintained for the service delivery orchestration process, and not for the end users to interact with directly.

Step 3 Click **Save**.

You can also configure custom attributes in the **Item Attributes** window when you create a new service item. You can add, update, or delete attributes. The service item attributes define various components of the service item, and also helps you prepare for the service item workflow. For more information on Service Item attributes, seeTable 8: Service Item Attributes table.

Adding a Custom Web Page For Displaying Service Items

By default, when a Service Item type is selected from the tree view, the standard My Stuff grid with Service Item attributes are displayed. You can customize the right pane view to display a web page you have designed to show the Service Items.

To add a custom web-page for the Service Item view in Service Catalog > My Stuff:

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Service Item Definition**.

Step 2 Select the Service Item from the tree view.

Step 3 Enter the required URL in the Custom View URL field.
For example: /RequestCenter/custom/CustomServiceItemView.html

Note As a best practice, place the HTML files in the same machine where the application is installed, in the path /RequestCenter/custom.

Step 4 Click **Save**.

Associating Services for a Service Item

The **Associated Services** sub tab of the **Design Service Items** tab allows service designers to associate a service item with one or more services. Associating a service with a service item allows Service Catalog and My Services users to use shortcuts to order services applicable to service items previously assigned to them. When a service item is chosen, all associated services for which the user has ordering permission are displayed in My Stuff.

You can also add conditions for these associated services. Only if a condition is met, these services will be shown in My Stuff. For more details, see Adding Conditions to an Associated Service.

When you click on an associated service, the service form appears. If service has been configured correctly, the service form is prefilled with information on the user's service item.

All services that include a Service Item Task to create, update, or delete a service item should be associated with that service item type.

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Associated Services**.

Step 2 Click **Add Services**.

Step 3 Select a service from the list of services.

Step 4 Scroll down, if there are multiple services and Click **Add**.

Step 5 Click **Save**.
These associated services are displayed as related services in **My Services > Service items > Related services**.

To delete an associated service:

To delete an associated service for a service item, choose the associated service by clicking on it (use Ctl+Click or Shift+Click to choose multiple services), and click **Delete**.

To control the related services that can be ordered based on status definition:

Choose the associated service and select status definitions check box as required to modify the display of the related services for a service item depending on the status of the service item. For example, if the service item is a virtual machine and the related services are Start VM and Stop VM. The states defined are online and offline. You can Start VM only when the status of the VM is offline.

Note If you have created a status definition for the service item as defined in Managing Status Definition, the status definition check box appears in Associated Service Window.

Adding Conditions to an Associated Service

Using this procedure you can add one or more conditions to an associated service. Only if the conditions are met, these associated services will be shown to a user in My Stuff.

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Associated Services**.

Step 2 Select an associate service from the list of services, and click **Add Condition**.

Step 3 Enter the attribute type , operation, and the value for the attribute type. Only when the attribute has this value, this associate service is displayed.

Note User can specify multiple conditions using OR , AND Operation to build any expression.

Step 4 Click **Add** and save the changes made to the associated service.

Understanding Service Items Policies

Policies are used to define the actions to be executed against a service item instance when certain conditions are met. A typical example is that when a virtual machine is approaching its expiration date, an automatic notification is sent to its owner as a reminder. Other policy actions may include triggering a service request to process the service item, or logging an alert for administrators, or preventing a new service request from being submitted. Policies may be defined to be specific to an account or generic to all accounts.

The policies are categorized into four types, namely: Capacity, Quota, Timebound, and Event-based policies.

Capacity- When you define a capacity policy, you enforce a limit for the usage of an attribute. An action is performed when the usage exceeds the defined limit. Before you define a capacity policy, you must define the service item attribute that has to be managed by the capacity policy. See Managing Service Item Attributes.

When you configure a service item attribute as a capacity attribute, the following attributes are generated to be used for capacity management:

- ◦ Used{AttributeName}, for example, "UsedVCPU" - Used by the application to track how much of the capacity is used. This value is not user-modifiable.
  ◦ Req{AttributeName}, for example, "ReqVCPU" - Used by service designer to pass the capacity change required in service requests or update operations in service item APIs.

You can use capacity policy to track your own resource consumption such as type of storage network, usage of each of the elements in the network, and take necessary actions accordingly if you are a service provider.

Examples of Capacity policy:

- ◦ You may define a policy that sends an email when the capacity of the storage area network reaches 80% of the capacity.

- **Quota -** The quota policy is executed when an Item Attribute reaches the threshold% of maximum value. Each quota policy is specific to an account and cannot be applied to multiple accounts. Before you define a quota policy you must:
  - Define the service item attribute that has to be managed by the quota policy. See Managing Service Item Attributes.
  - Ensure a quota is already defined in an agreement that corresponds to the account being referenced in the quota policy. This is maintained under **Demand Management > Agreement**. The service item that is subject to quota management should already be included in the agreement quota with an appropriate aggregate method (see Configuring Billing Rates Based on Usage (Agreements) for more information).
  - When specifying a quota policy, select the appropriate aggregate method that tracks the quota consumption. The “Sum” aggregate method manages the quota based on the sum of a specific attribute consumed by all service item instances (for example, sum of disk size for all datastores owned by the account). The “Count” aggregate method manages the quota simply based on the service item count (for example, total number of virtual machines owned by the account).

Examples of Quota Policy:

-
  - A policy that sends an email when the count of the virtual machine for an account reaches 70% of the quota.
  - A policy that orders a service when the storage of the virtual disk for an account reaches 75% of the quota.

- **Timebound -** This policy is executed when the date attributes of the service item reaches or exceeds the defined value. A poller evaluates the time bound policies at a time interval of every 30 minutes by default and performs actions as necessary. You can enable/disable the poller in the support.properties file. For more information about configuration files, see Cisco Prime Service Catalog Administration and Operations Guide.

Examples of Timebound Policy:

- A policy that sends an email to the owner of a software subscription one month before the subscription expiration to remind the person to renew the subscription.
- A policy that revokes the subscription to a software on the lease expiration date.

To see the example of how you can terminate a lease using this policy, see Configuring Timebound Policy.

To define this policy on a service item, there must be at least one service item attribute of attribute type “DateTime”.

- **Event based -** Event based policies support three conditions:
  - An action is triggered when a service item attribute reaches a particular value.
  - An action is triggered when a service item attribute reaches a particular value, or when the attribute value has been modified.
  - An action is triggered when the service item attribute satisfies certain conditions. You can define a condition for a service item attribute, to determine if the particular value 'is equal to', 'is not equal to', 'contains' or 'does not contain' based on the options you can select from the logical operator drop-down list to trigger an event. You can either enter the value for the service item attribute or

select from the namespace icon. You can add multiple conditions and more than one service item attribute for a particular value.

Examples of Event-based Policy:

- ◦ Perform action when attribute is updated - Create a policy that records a policy alert when the status of the virtual machine becomes offline.
  ◦ Perform action when conditions are met - Create a policy with conditions that record a policy alert when the RAM value has reached its maximum limit and sends an email when the status of production HANA database goes offline and the backup server is active.

Policy Actions

One or more actions may be performed when the policy condition is satisfied. The predefined actions are:

- Stop submission- Prohibit the requester from making another request to consume the service item. This action is not applicable to the time bound policy type.
- Order Service- Spawn a request for a service within the catalog to allow a supporting operation to take place. If you choose to order a service when a condition is satisfied, then you must also:
  ◦ Select a service from a list of existing service definitions
  ◦ Specify the values to be passed to the service form. Any namespaces referenced will be resolved to the actual values based on the context when the service is ordered. For more information, see Namespace Usage for Policy Configuration.
- Send email- If you choose to send an email when the policy condition is satisfied you must also select an email template from the email template drop-down list. You can define email templates in **Administration > Notifications**, just like any other email templates used for service delivery.
- Policy alert- Log the policy trigger occurrence that serves as an audit trail for administrators to review. If you send a policy alert to the application you must also provide a descriptive message for the alert. You can also use namespaces to specify the service item attribute that will be resolved when the message is displayed.

To view the list of policy alerts when the condition is satisfied, add the Policy Alert portlet to any existing portal pages in Service Portal module. Choose **Service Portal> Edit Page > Add Content > Reserved portlets**. Click **Policy alerts** check box. Policy alerts are logged for Policy alert and Order Service actions.

Configuring a Service Item Policy

Each policy type has one or more policy templates. The policy templates are rules that define when to perform an action. You can further define parameters for the condition and also specify the action to be taken when the conditions are satisfied.

Before You Begin

Understand the types of Service Item Policies in Prime Service Catalog. See Understanding Service Items Policies.

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Policy**.

Step 2 Select a service Item from the list panel and click **Add Policy**.

Step 3 Enter the attributes for the new policy type. The table below describes some of these attributes.

Step 4 Click **Add** on the Actions page and choose an action from the list.

Step 5 Click **Next** and verify the Policy you created in the Policy Summary page and click **Done**.

Note If you choose an account attribute for a policy, then the policy becomes applicable to the service items associated with the account. If not, the policy is applicable to all service items.

Table 9: Service Item Policy Attributes

Field	Description
Policy Type	Select the Policy Types as Capacity, Quota, Timebound, and Event-based policies.
Template	Select the type of actions to be executed against a service item instance when certain conditions are met.
Setup Condition	Define conditions in the **Setup Condition** page. The **Setup Condition** page options vary based on the template you choose. • For capacity and quota policies, the service item attributes appear only if you have defined them in **Service Item Definition** tab > **Add Service Item Attribute** table > **Managed By** field. See Managing Service Item Attributes. For a capacity or quota policy, you need to define it against a service item attribute that is numeric. • For event-based policy, you can configure multiple conditions if you choose the template option **Perform action when conditions are met**. These multiple conditions are evaluated based on the and/or logical operators and the associated action is triggered based on a particular value.

To view the list of policies defined already for a service item, click on a service item from the Service Items list panel and select the **Policy** tab. You can choose to see all the policies for an account using the **Filter by account** field.

Configuring Timebound Policy

Time bound policies can be put in place to enforce lease management. For example, you may set up a time bound policy to ensure that the virtual machine is stopped on the lease expiration date. In this procedure you send an email and stop the virtual machine.

Step 1 Choose **Service Item Manager** > **Design Service Items**.

Step 2 Select a Virtual Machine service item from the list panel that lists the Service Items on left hand side window. Click **Policy** tab.

Step 3 In the Policy tab, click **Add Policy**.

Step 4 In the **Name** field, enter Stop VM on lease expiration, choose Time-bound in the **Type** field. Click **Next**.

Step 5 In the Template page, choose template: Perform action when system date is equal to attribute value". Click **Next**.

Step 6 In the Condition page, do not enter any information in the Account field. Select **LeaseExpiryDate** value in the Service Item attribute field. Click **Next**.

Step 7 In the Action page, Click **Add** and select **Policy Alert**. In the Policy Alert Window, enter a message and click **OK**.

Step 8 Click **Add** again and select Order Service. A new window displays the services. Choose **Stop VM** and click **Add**.

Step 9 Click **Next**. A Policy Summary page that describes the policy type, condition, and the action is displayed.

Step 10 Click **Done**.

Configuring Custom-Defined Status For a Service Item

Status definition tab is displayed only if you click on the Has Status icon when you create or update the service item. You can create custom-defined statuses for the service item. A service item can have multiple statuses. For example the virtual machine service item can be online, offline, commissioned, decommissioned, deleted, and so on.

You can also define a status transition to enforce that the service item could be transitioned from the previous state to the defined state. "Any" is a default state that allows status transition of a service item to any status from the previous state.

Note You can add a maximum of 20 status definitions.

To add a status definition

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Status Definition**.

Step 2 In the **Status Name** window, Click **Add** and an empty line opens at the bottom of the grid, where you can type the status definition.

Step 3 Click **Save**.

Defining a Status Transition

To delete one or more status definition, choose the status definition by clicking on it (use Ctl+Click or Shift+Click to choose multiple attributes), and click **Remove Selected**.

To define a status transition

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Status Definition**.

Step 2 Select the status on the **Status Name** window.

Step 3 Click **Add** in the **Next Status(es) Transition Allowed** window. An empty line opens at the bottom of the grid with a drop-down list that contains the statuses already defined.

Step 4 Choose the next status transition you want to define from the drop-down list. This list contains statuses that you defined in the Status name.

Step 5 Click **Save**.
If the service item already has the **Status** attribute defined and there are service item instances populated with certain values for the Status attribute, you can still enable the **Has Status** setting. When you do so, the system takes over the control of the Status attribute, extracts all status values in use and automatically populates the Status definition tab with a list of the unique values.

Defining a Finite State Transition for a Service Item

Using service items operations, you can define a finite state to ensure only a certain state transitions occur for a service item. For example, an offline VM can only be started, but not stopped.

The default operations that could be performed in the life cycle of a service item are create, update, and delete. You can also add more operations to the service item lifecycle that gets saved in the SIBD.

You can monitor the operations performed in the lifecycle of a service item **under** Service Item Manager > Manage Service Items > History.

Before You Begin

- Create a status definition. See Configuring Custom-Defined Status For a Service Item.
- Add Associated Services for a Service Item. See Associating Services for a Service Item.

To add a Service Item Operation

Step 1 Choose **Service Item Manager > Design Service Items > Operations**.

Step 2 Click **Add** and specify the various attributes for the new operation. The following table describes these attributes.

Step 3 Click **Save**.

Table 10: Service Item Operations Attributes

Field	Description
Name	Enter a name of the operation.
Resulting Status	This field lists the predefined statuses. Choose the status that you want the service item to be in, when the operation or the service item task is executed. The drop-down displays the list of statuses that were added in **Design Service Items > Status Definition** tab. **Note** If you uncheck and then check the Has Status check box in the Service Item Definition window, the Resulting Status will be set to the first available status in the drop-down list.
Associated Services	Select one of the associated services for the service item from the drop-down. The drop-down displays the list of services that were added in **Design Service Items > Associated Services** tab.
Billable	Select this checkbox to trigger a billable event and make an operation available for the billing rate table definition. For example billable event can be “Provision a VM” or “Delete a VM”, which incurs or stops the charges for a subscribing user. For more information on this billing rates, see Configuring a Billable Rates, on page 226.

To delete an operation, select an operation from the Operation Summary table and click Remove Selected.

Defining Permissions to Perform Operations for Service Item Types

The read/write access to service items in Service Item Manager is governed by capabilities and permissions granted to the user. Capabilities control which tabs and panes the user can use. Permissions control which service item types and instances the user can access and which actions (create/update/delete/view) are allowed.

The Role-based Access Control (RBAC) for service items can be maintained in Organization Designer as well as the Permissions tab under Define Service Items. In Organization Designer, RBAC permissions are granted to a role and the role is in turn granted to the user directly (to the person himself) or indirectly (to an organizational unit or group that the person belongs to). In Service Item Manager, the permissions are granted with or without the use of a role. The use of roles allow easier tracking and maintenance and is generally recommended.

To add permissions:

Step 1 Choose **Service Item Manager > Design Service Items > Permissions.**
The Permission Summary window lists the permissions assigned to the selected service item.

Step 2 Click the magnifying glass icon in the Role field, select a role and click the **Select** button.

Step 3 Select a permission from the **Permissions To** drop down list.
For more information on permissions that can be added, see Service Item Manager module in Assigning Permissions section in Cisco Prime Service Catalog Administration and Operations Guide.

Step 4 Click **Add**.

To delete permissions select a role from the **Permission Summary** table and click **Delete**.

Customizing My Products & Services Page

This section describes procedures to customize My Products & Services page of Service Catalog module.

Displaying Out-of-Box Service Item Attributes

All service items have a set of out-of-box service item attributes and you can also configure custom-defined attributes for a service. Custom-defined attributes are displayed along with these out-of-box attributes in the **My Stuff** page. For more information about configuring custom-defined attributes, see Managing Service Item Attributes, on page 34. By default, all the out-of-box service item attributes are displayed in My stuff.

Step 1 Choose **Service Item Manager > Design Service Items > Subscription Attributes Configuration.**

Step 2 If you do not want to display an attribute, clear the **Show in My Stuff** checkbox for the required subscription attributes and click **Save**.

To rearrange the service item attributes, select the attribute and click on the up or down arrows on the right side to place the attribute in the desired order. Click **Save**.

Managing Service Item Attributes

You can create custom attributes in the Item Attributes window of the content panel when you create a new service item. You can add, update, or delete attributes. The service item attributes define various components of the service item, and also helps you prepare for the service item workflow.

Note Do not add attributes maintained as part of the Service Item Subscription or Service Item History to the service item definition. Such attributes, including the Customer ID, RequisitionID and other aspect of service item usage, are described in the Service Item-Based Dictionary, on page 43.

After the definition is changed, existing service items will have blank values for the added attributes. Administrators can use the Manage Service Items option to provide values for these new attributes.

If a Service Item-Based Dictionary has already been created, any changes made to the service item definition need to be applied manually to the dictionary. As long as the name of a field added to the dictionary exactly matches the name of the corresponding attribute in the service item, the relationship between the field and attribute is maintained.

To add a new attribute:

Step 1 Choose **Service Item Manager > Design Service Items > Service Item Definition**.

Step 2 In the **Item Attributes window**, click **Add**. An empty line opens at the bottom of the grid, where you can type the service item definition.

Step 3 Update various attributes by clicking on each column.

Step 4 Click **Save**.

Table 11: Service Item Attributes

Field	Description
Display Name	Enter a name for the attribute. This name is displayed in the **My Stuff** window.
Name	Enter a name for the attribute.
Attribute Type	Choose an **attribute type** from the drop down list. It can be a string data type, integer, string (max), and so on. You can also add a service item type as an attribute type for a service item. This is to support the design of hierarchical service items when designing components for an application stack. Use the option "**Is Reference**" for this hierarchical service item, if you want the child service item to be used by multiple parent. **Note** All the CRUD operations for this hierarchical service item is supported only through REST API. These CRUD operations applies only to each Service Item type. For more information on the REST API for the hierarchical service item, see *Cisco Prime Service Catalog Integration Guide*. Some examples for: • **Hierarchical Service Item Attribute Type** : For a service item of type "Laptop", you can add another service item "Hard disk" as an attribute type. • **Other Attribute Types**: Consider virtual server as a service item; the attribute type for the service item attribute RAM could be integer, and the attribute type for the Expiration date is DateTime

Field	Description
Managed By	If you have planned to define a capacity or a quota policy for the service item attribute, you must select the policy type from the drop down list. This is an optional field. See Configuring a Service Item Policy, on page 29.
Accessible In My Stuff	Select the check box if you wish to make the service item attributes available in Service Catalog > My Stuff settings menu. Using the settings menu you can display or hide the attributes in the My Stuff grid.
Show In My Stuff Grid	Select the check box if you wish to display the service item attributes in the Service Catalog > My Stuff grid by default.
Is Friendly Name	You can select any one of the item attributes as a friendly name. This friendly name is displayed instead of the name attribute value in the Service Catalog > My Stuff grid.
Is Reference	Select this check box if there is an already pre-existing object of the associated type. Keep it unchecked if there is an object that is "owned" by this service item.
Multi-value	Select this check box if this attribute is an array, that is it can store more than one value.

Importing Service Items from an External System

You can import Service item from an automated or semiautomated system external to Service Catalog. Service Catalog can then track service item provisioning and other updates. Service item data can be imported at any time, to synchronize Service Catalog data with other records that may have been maintained externally. In addition to importing service item data, the service item definition itself can be imported, either with or without corresponding data. This automates the definition of the service item, so that it does not have to be done manually.

You can import service items and standards using one of the following methods:

- The Import Data option in Service Item Manager allowing administrators to import service items on demand from file. See Importing Service Item from File, on page 37.
- A request can include a Service Link task to import service items or standards. See Importing Service Items and Standards using Service Link, on page 135.

Importing Service Item from File

You can import data and definition for service items from an external source, using Service Item Manager > **Import from File** option. The file to be imported must use ANSI encoding—either ASCII or UTF-8. Unicode encoding is not supported. You can select either **Data** or **Definition** option to import only those portions of the file, or you can select both. If you choose to import both and the XML file contains only a definition section, the system only imports the definition.

When service item data is imported, an entry is added to the Service Item History table and the Service Item Subscription is created or updated, as appropriate.

The table below summarizes the behavior of the Import Utility when importing service item definitions or data from file. This behavior also applies to a Service Link Service Item task, described later.

Table 12: Import Utility Behavior

Definition/Data	Conflict Resolution	Description
Definition	Overwrite	The import replaces the existing Service Item definition with the definition contained in the import. For an existing service item, existing data is preserved when column names between the old and new versions match.
	Merge	The import can be used to merge new columns into the existing Service Item definition. A new Service Item is created if the item does not already exist.
	Insert	The import fails if the service item already exists. If the service item does not exist, the service item and underlying database table are created.
Data	Overwrite	Update (overwrite) any existing service items with a matching item found in the import; insert any new items in the import file. If the import file does not contain a value for an attribute of the service item, then the value of the attribute is set to null for that service item, even if the service item previously existed and the attribute had a value.

Definition/Data	Conflict Resolution	Description
	Merge	Update any existing service items with a matching item found in the import; insert any new items found in the import file. There are two conditions in which the attributes of an existing service item will NOT be updated: (1) the import file does not contain the property tag for this attribute; (2) the service item attribute already had a non-null/zero value. In either of these cases, the existing attribute value will not be overwritten.
	Insert	Insert any new Service Items found in the import; existing Service Items are unchanged.

Note The description field for attribute does not exist on the UI. It is optional in the SI Import file.

Configuring Status for a Service Item

You can ensure that an associated service for a service item is orderable only when the service item is in certain statuses. Here is an example of how you can achieve that:

Step 1 Enable **Has Status** for the service item.

Step 2 Add custom statuses in the**Status Definition** > **Status window**. For example a virtual machine service item may have these statuses defined: Active, Being Reconfigured, Inactive.

Step 3 Add status transitions to prevent invalid status changes. For example, a virtual machine in Inactive status cannot be turned back to **Being Reconfigured**. See Configuring Custom-Defined Status For a Service Item, on page 31.

Step 4 On the Associated Services sub tab, select the check boxes for the associated services that are allowed to be ordered only when a service item is in certain statuses. For example, the service "Extend VM Lease" is allowed for Active status only.

Step 5 Add operations in the Operations tab and the corresponding resulting status such that it co relates.
For example:

Table 13: Operations tab and corresponding Results

Operation	Resulting Status
Provision	Active
Terminate	Inactive

Operation	Resulting Status
Initiate Reconfiguration	Being reconfigured
Complete Reconfiguration	Active

End User's View of Service Items

Service items are automatically assigned to a person when they are provisioned via service requests. Service items can also be assigned directly to a Service Catalog user through the manual Assign feature in Service Item Manager. Manual assignment is particularly useful if users have been assigned service items in an external system and that data must be brought into Service Catalog.

Service Catalog and My Services offers a service item view (My Stuff or Service Item Portlet) that enable users to see the service items that have been provisioned to them and to request further changes and additions to these items. The My Services Consumer role (the default role assigned to all users, to allow them to submit service requests) does not include the ability for users to view and search on their service items. This functionality is provided by the My Services 360-Degree Consumer role, which includes the capability to "Access Service Item Instance Data" and a personal preference setting allowing each user to show or hide the My Items portlet.

A similar view is available in the Service Catalog module, under the Manage Your Stuff spotlight on the top right-hand section of the home page.

Viewing My Service Items

If you are using the Service Catalog module navigate to Service catalog> Manage My Stuff > My Stuff to access and order services associated with service items.

Viewing Service Items in the Old My Services Option

My Services offers a **Service Items** tab and **My Items** portlet that allow users to access and order services associated with service items. Service items are assigned to a person when they are provisioned via service requests, or when the items are assigned directly to the person using the Service Item Manager module.

The My Items portlet displays the five service items most recently provisioned to the user. Click on the **More** link in the My Items portlet or the **Service Items** tab to see a complete list of service items. In addition, the service items that each user can access depend on the read/write permissions he has been granted. Users are always given read/write access to service items that they own.

Managing Service Items on an External System

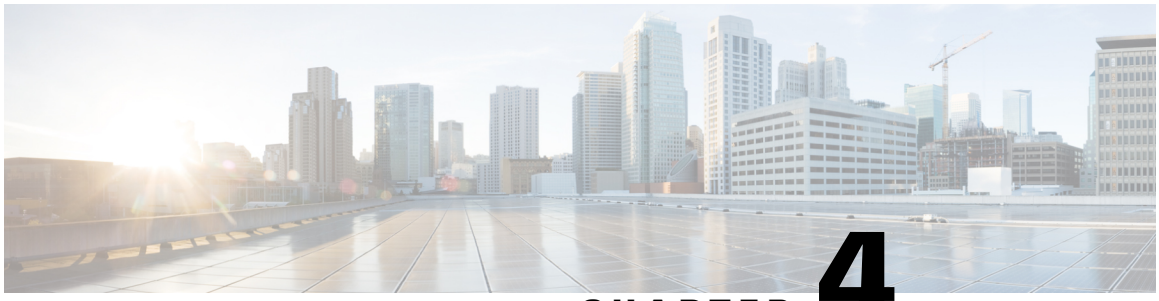
Some service items you create and track in Service Catalog may already be tracked to some degree in an external Asset Management System (AMS), or in a target platform such as a hypervisor. Laptops and workstations, as well as virtual machines and virtual applications, are good examples of such service items. Using an external system depends on number of factors, including the maturity of the external system, the

nature and quality of data in that system, the frequency by which the system changes and whether there are events in the system that you wish to have trigger changes in Service Catalog. Also which attributes (that is, data elements) of service items you want the end users of Service Catalog to see. For example, things like the Manufacturer, Make, and Model of a laptop; or the GUID of a virtual machine, are not going to change and therefore do not necessarily need to be stored and updated in Service Catalog. To get data from an external system use:

- ◦ A data retrieval rule that queries the external database, based upon the Asset ID. For information on data retrieval rules, see Creating Dynamic Data Retrieval Rules, on page 87.
  ◦ A Service Link that provides a "process API" that is easily built into the workflows for services.
  ◦ Web services adapter to get data from the external system in an inbound message.

Best Practices When Using an External System:

1. **Store the Asset ID only in Service Catalog**. The only attribute for the Laptop SI is the Name (into which you write the Asset ID from the external system). The end-user in My Services sees only *that* . When this laptop is referred to in service requests, you use one of the methods to pull the current data (Make, Model, Manufacturer, Disk Space, and Memory) from the external system onto the form:
2. **Store the Asset ID, plus attributes that won't change, in Service Catalog**. Assume the attributes for the Laptop SI are the Name (Asset ID), Make, Model, and Manufacturer. An initial Asset ID lookup brings those attributes onto the form from the external system, and stores them in the Service Catalog database. When this laptop is referred to in service requests, you use one of the two methods mentioned above to pull current data for Disk Space and Memory from the external system onto the service form. (In other words, Disk Space and Memory are never stored in Service Catalog.)
3. **Same as #1 or #2, but enhance the service item data with business-contextual data**. This is really just a variation on the two approaches listed above. The business data gathered during the processes of requesting services and fulfilling those requests is captured and written as service item attributes in Service Catalog. This data is never stored in the external system.
4. **Store all attributes in Service Catalog, the single "system of record"**. An initial load (via Import) is done to populate Service Catalog with all laptops. From then on, everything affecting laptops is handled via Service Catalog service requests. *Now* the question becomes updating the external system. You could do this either manually or via a Service Link agent.
5. **Store all attributes in Service Catalog; changes may occur both in Service Catalog and in the external system**. As in #4 above, but there may be events occurring in the external system that need to be reflected in Service Catalog. For example, you may not want to consult the external system to see the list of laptops currently in inventory (that is, not assigned to anyone); maybe you want all those stored in Service Catalog for easier data retrieval. In this case, you could use the Import API or Service Item Listener Adapter to handle updates from the external system to Service Catalog.



CHAPTER 4

Configuring Forms for a Service

This chapter contains the following topics:

Configuring Forms for a Service

Introduction

A service form is used by an end-user to define their service request. Using Service Designer module, you can configure the service forms, and create data retrieval rules to display specific choices when ordering services. Service Form design uses these components:

- **Dictionaries** are group of fields to enter for a service request. There are four types of dictionaries that can be added in Prime Service Catalog. To know more about these dictionaries, see Configuring Service Form Fields Using Dictionaries, on page 41.
- **Standards** that aid in designing the active form components. To know more about standards, see Defining Standards in Service Design, on page 58.
- **Active Form Components** help in defining service form's appearance and behavior based on user-initiated events. To know more about the Active Form Components, see Configuring Service Form Appearance and Behavior Using Active Form Components, on page 62. You can also create rules to alter the behavior and appearance of a service form based on specified conditions. For more information, see Configuring Dynamic Form Behaviors Using Form Rules, on page 72.

Configuring Service Form Fields Using Dictionaries

The basic building block of a service form is a dictionary, a group of individual data elements (fields) that allow users and service performers to enter and view data required to fulfill the service request. The dictionary should incorporate all fields that may be required in these request.

This principle (of designing a dictionary and form component for reuse) is particularly important for Service Item-Based Dictionary, on page 43, since services may be available for the entire lifecycle of the service

item, not just the initial request to provide a service (or item) to the customer. You may want to design the dictionary to support the tasks that can be included in a service request; create the service item; modify the service item; or remove the service item from its current owner.

Types of Dictionaries in Prime Service Catalog

The two main categories of dictionaries, Internal and External, refer to the way data in the dictionary is stored.

- Internal dictionaries represent data structures that are managed by, and within, Service Catalog. Internal dictionaries are generally preferable, since they minimize database administration required and include additional functionality not available in external dictionaries. Types of Internal dictionaries are:
  - Internal Free-Form Dictionaries, on page 42
  - Internal Person-Based Dictionaries, on page 42
  - Service Item-Based Dictionary, on page 43
  - Automatically Included Reserved Dictionaries, on page 47
  - Automatically Included Integration Dictionaries, on page 49
- External dictionaries, use existing or new data tables outside the Service Catalog requisition. For more information, see External Dictionaries, on page 46.

Internal Free-Form Dictionaries

A free-form dictionary means that service designers are free to specify the fields in the dictionary, the order in which they occur, and the data types assigned to each. To configure the fields in the free-form dictionary, see Table 15: Fields to Configure Internal Dictionary.

Internal Person-Based Dictionaries

Cisco Prime Service Catalog maintains a repository of all persons in the user organization who may need to access Prime Service Catalog applications. Personnel information in Service Catalog is viewable via Organization Designer. The personnel information in this repository is generally populated via directory integration. Any time a user logs in to Service Catalog, or has a service ordered on his/her behalf, his/her personnel information in the repository can be refreshed with their information from an enterprise-wide directory. Service forms typically need to refer to personnel information. For example, a service request always has a customer—the recipient of the service; and an initiator or requestor—the person who sits at the keyboard and requests the service. In most cases, the customer and the initiator are the same person—an employee requests a service for him/herself. Alternatively, an administrator or other authorized employee may initiate a service request on behalf of another person, who then becomes the customer for the service. To configure the fields in this type of dictionary, see Table 15: Fields to Configure Internal Dictionary table.

When to use Person-Based Dictionaries?

If you are only referring to the request's as a customer or an initiator, you do not have to create a person-based dictionary. Instead, use the Customer_Information and Initiator_Information dictionaries that are part of the Reserved dictionaries in Prime Service Catalog.You have to create person-based dictionaries when requesters frequently need to designate another person who may be involved in the fulfillment of the request or needed as a reference.

Use case for Person-Based Dictionary

A frequent use case for person-based dictionaries is when an initiator designate one or more approvers for a request, if the approvers cannot be inferred from the supervisory structure available from the directory integration. In this scenario, the requester selects the approver by searching through a drop-down list of available personnel and the approver's contact information is then included in the service form data for easy reference.

About Person Search Option

When a person-based dictionary is used on a service form, you can choose a single person specifying the search criteria in the “Person Search” dialog box. Once a person is chosen, fields used in the dictionary are automatically filled in with the current values of the corresponding fields in the chosen person’s profile. The name appears in FirstName LastName format.

The “Select_Person” attribute provides the capability to search for people, either within the repository or, if directory integration has been enabled for service forms, in an external directory. This attribute has a data type of “Person,” not text. This data type governs the appearance of the attribute when it is used in a form. For example, the “Name” field on the service form below is defined as the “Select_Person” attribute. In any new person-based dictionary, you will see “Show in Grid” is checked by default for the “Select_Person” since it also checked for use Use by default. “Show in Grid” cannot be unchecked for the Select-Person, just as you cannot uncheck Use. Select the other dictionary attributes to be displayed on the service form by clicking the Use checkbox.

Figure 1: Select Person Attribute

Service Item-Based Dictionary

A Service Item-Based Dictionary is a dictionary whose structure (fields) are based on a service item defined in the Service Item Manager module. This dictionary also holds data about that service item collected during a service request. To see type of data that can be configured for a service item, see section on Defining Service Items for a Service, on page 20.

Fields in that dictionary provide the data stored on the service item and can optionally serve to update the history of the service item and its subscriptions. For more information, see Service Item Subscription and History Fields.

Adding Service Item-Based Dictionary

Using this procedure create a service item-based dictionary based on attributes defined while creating service items,

Before You Begin

Configure Service Item, based on which the dictionary will be defined.

Step 1 Choose **Service Designer** > **Dictionaries**

Step 2 Choose **New** > **New Dictionary** to display the New Dictionary page.

Step 3 In the Add New Internal Dictionary section, in the **Service Item** field, enter one or more characters of the name of the Service Item you want, or enter "*" to search for all Service Items. Click on the Service Item you want.
The values for **Category**, **Service Item Group**, and **Service Item Type** fields are set to the group and service item type you selected. As in previous releases, Service Item Family is not used by Service Catalog and any value may be supplied. This value is available for query and grouping in the Service Catalog data mart. The other fields that can be included in a service item-based dictionary are:

- Fields corresponding to attributes defined in the service item itself. For more information, see Service Item Fields.
- Fields about the service item subscription (history) and delivery history that are available for use in conjunction with the dictionary. These fields provide additional information regarding the service item's current usage and subscription history. For more information, see Service Item Subscription and History Fields.
- User-defined fields. For more information, see User-Defined Fields.

Step 4 Click Save Dictionary.

Note If you want to add a new attribute to the corresponding service item after creating the dictionary, you must manually add the new field. This is because Service Catalog does *not* automatically add a new field corresponding to that attribute to the dictionary. As long as you use the same field name as the attribute name (the Data Name, not the Display Name), Service Catalog will correctly synchronize fields in the dictionary with the attributes in the service item. Similarly, if you create a SIBD and then *delete* an attribute from the corresponding service item, you must manually remove the dictionary field corresponding to the deleted attribute. If you do not, you will end up with a field on the form that has no corresponding data for the service item.

Service Item Fields

Attributes specified in a user-defined service item are, by default, included in dictionaries based on that item. To add additional fields to the dictionary, select the check box on the left of the attribute name under Dictionary Attributes. Any other fields except the **Name** field are optional. The Name field has a reserved data type as "Service Item Identifier". All data types are inherited from the service item's definition. Select the **Encrypt** check box to encrypt sensitive data such as passwords, beneficiary names, account numbers, and so on.

Note Use String (max) field type for service item dictionaries that will be encrypted to ensure that the encrypted string does not exceed the character limit.

The String (max) limit is configurable to 6000 in cnfparam table by the system administrator. However, ensure that the new string limit value does not create performance issues when configured in the service item dictionary. For more information on encrypting sensitive data, see Table 16: Dictionary Attributes table.

Service Item Subscription and History Fields

The next set of fields that appear on the Dictionary page correspond to data maintained for the service item subscription and history. The requisition's subscription and history data is recorded in the Service Item History when the Create or Update Service Item task for the service item is executed. This option for an end user is available in the **My Stuff** page, and for an administrator in the **Service Item Manager > Manage Service Items > History** tab for each item.

Service Item Subscription fields are summarized in the table below. For the pre configured subscription fields, choose the fields you want to include in the dictionary by checking the check box to the left of the field name. The fields listed below are automatically included in the service item's history and subscription data, with appropriate values filled in, even if you do not include them in the dictionary. The reason for including them in the dictionary is to supplement or override the default behavior that Service Catalog uses to provide values to these fields.

Table 14: Service Item Fields

Field Name	Description
CustomerID	ID of the customer for the request/service item.
RequisitionID	The ID of the requisition (shopping cart), which includes the service request that created the service item subscription.
RequisitionEntryID	The ID of the service request that created the service item subscription.
OrganizationalUnitID	The Home OU of the customer to whom the service item is assigned.
AssignedDate	The date and time the service item was assigned to the customer, that is, the date the "Service Item - Create" task was executed.
SubmittedDate	The date and time the request, which included the "Create Service Item" task, was submitted.
ErrorCode	Not used.
ErrorDescription	A textual description of the error code received if an attempt to create or assign the service item failed.
Account ID	By default, the Tenant Account of the Home OU for the customer to whom the service item is assigned.

Note If CustomerID and OrganizationalUnitID are not included in the service item-based dictionary, Service Catalog uses its default logic to supply a value for each field. If a field is included in the dictionary, the value of the form field at the time the service item task is executed is used.

- If CustomerID or OrganizationalUnitID are included in the dictionary, the service designer is responsible for writing a rule or other mechanism to provide a value to the field.

Some sample scenarios for overriding the default behavior of Service Catalog include:

- Create the service item (SI), but with no owner. In other words, no one has a subscription to it. For example, a new laptop has arrived, but it has not yet been assigned to anyone. This would be an alternative to manually creating the service item via the **Manage Service Items** tab or importing the service item from an external source. In this scenario, both the CustomerID and OrganizationalUnitID would be included on the service form (probably hidden by a rule in the Ordering moment), but no value supplied. Add AccountID every time when both CustomerID and OrganizationalUnitID are referenced.
- Allow the request initiator to explicitly choose the customer for the service item (via a person-based dictionary). Copy the customer's information to the CustomerID and OrganizationalUnitID fields included in the SIBD, overriding the default customer (the initiator). Add AccountID every time when both CustomerID and OrganizationalUnitID are referenced. This scenario is explained in more detail in the XREF section.
- Create the SI with no specified owner (a person corresponding to the CustomerID), but with an owning OU. For example, a project team needs a server, but the server is the responsibility of the team, not an individual person. This scenario is similar to the one above, but only the CustomerID needs to be in the dictionary, its contents blanked out by a rule in the Ordering moment. The OrganizationalUnitID can be the OU ID of the initiator or customer, or you could choose an OU/customer via a Person Search function.
- Unassign the service item from someone. For example, when an employee's equipment is temporarily unassigned after he or she leave the company or a project.
- Run a data retrieval rule to display information about when the SI was created (for example, which Requisition and Requisition Entry IDs were on the previous order).

The error description field provides feedback in case a service item task fails. This is a useful for debugging, by giving clues on possible conditional rules that might be required to validate data entry.

Possible reasons for failure to create a service item include leaving the item's Name blank, or attempting to create a service item of the same type with the name of an item that already exists. If such a task fails, the service item is not created, and any changes to the requisition are rolled back. Similarly, a task to update or delete a service item might fail if the referenced item does not exist.

User-Defined Fields

Service item designers may add fields to any service item-based dictionary. Such fields might not be appropriate for the service item itself, but are a critical part of service requests relating to the service item. For example, you may want to include comments on why the customer is requesting the item, or initial or recurring monthly costs associated with item usage. Field names must not match the names of fields from the Service Item History/Subscription available for use, even if those fields are not included in this dictionary.

Such a field just becomes data that lives on the form and in the dictionary data for the form; it will never be recorded as part of the service item and is not viewable via the Service Item History and Subscription queries. However, like any dictionary field, it is viewable in the completed request and reportable via the Advanced Reporting module if the dictionary or a service that includes the dictionary is made reportable.

External Dictionaries

External dictionaries are added and maintained in Service Designer > Dictionaries. Using an external dictionary in read/write mode can result in corruption to the database that you are accessing from the system if you have not set up safe data transfer methods. Read the following tips:

- The Requisition Entry ID generated by the system needs a column and primary key of its own in the external database table. Otherwise, the system can overwrite other data in the table.

- We recommend that your site create an intermediate table with a key column to which the system writes new entries. This table can then work with your existing table.
- Before adding an external dictionary, consult with the external table's DBA or owner to ensure that the system uses the correct columns for the Requisition Entry ID and the primary key.
- The system does not automatically synchronize changes that you make to external database tables with its own external dictionaries that reference those tables. Changes to column names or sizes, for example, will not flow through to external dictionaries, and services ordered that depend on those dictionaries will fail. Once you have created an external dictionary in the system, each time you change the structure of the external database table, you must save the definition page of the external dictionary in the Dictionaries component of Service Designer. When you click Add This Dictionary on this page, the system will synchronize the table structure, and display the new information in the section that describes the external database columns.

Automatically Included Reserved Dictionaries

Every Service Catalog instance automatically includes two dictionaries, provided as seed data for customer and initiator information. These dictionaries can be found in the "Reserved" dictionary group in the Dictionaries component of Service Designer.

There is also a "Reserved" form group in Active Form Components, which contains the Customer-Initiator form. This form includes the Customer_Information and Initiator_Information dictionaries.

Features of Customer_ and Initiator_Information dictionaries:

- Supplement the standard behavior of automatically recording the customer and initiator for all requests.
- All information on the customer and initiator is available throughout the requisition life cycle via Business Engine namespaces, and is displayed as part of the Requisition Summary page for the request in My Services.
- These dictionaries are included in active form components and reusable for all services for the following reasons:
    - These fields reflect the values that were retrieved from the person's profile.
    - The person ordering the service has no opportunity to correct information that may be out of date or to supply additional information needed to fulfill the current request.
    - In addition, for reporting and governance reasons, it may be required to track customer and initiator information as they were when the request was submitted, not reflecting any subsequent changes.
- The dictionaries and group names for reserved dictionaries cannot be modified.

- The reserved dictionaries are included on a reserved form, the Customer-Initiator form. Form contents and appearance can be defined and its behavior manipulated by any form rules or ISF available.

Figure 2: Requisition Summary Page in My Services

The dictionary and group names of the Initiator and Customer Information dictionaries cannot be changed. Other general properties of the dictionaries are editable. The reserved dictionaries are included on a reserved form, the Customer-Initiator form. Form contents and appearance can be defined and its behavior manipulated by any form rules or Interactive Service Forms (ISF) available.

The reserved dictionaries list all available personnel information, and allows to designate which attributes should be part of each dictionary. Choose the attributes to be included in the dictionary by checking the corresponding attribute name. For example, include Supervisor information, since some service requests may require the customer's supervisor's approval. Attribute names and data types cannot be changed. Ensure that you include any attributes that need to be manipulated in a service, even if the field will always be hidden. Choosing the attribute ensures that it is populated with the corresponding value from the person's profile. Designers can then configure the form containing the dictionary to hide the field as appropriate.

The personnel profile includes 10 custom fields (named Custom1 through Custom10) that are not used by Service Catalog. Any of these fields can be included in the Customer or Initiator dictionary. Some of these fields may be mapped to person attributes imported via directory (LDAP) integration. Others may be reserved for assignment or manipulation via the Display Properties or Conditional Rules available when configuring the Customer-Initiator Active Form Component which includes this dictionary.

For a Person Based dictionary and reserved dictionaries (Customer and Initiator dictionaries), you use the Dictionaries component of Service Designer to choose which dictionary fields (attributes) will appear on a service form. Simply navigate to the dictionary, and check or un-check the check boxes in the "Use" column to determine which fields are used, or included, on a form that uses the Person Based dictionary, or on the Customer-Initiator form—the active form component which automatically includes both Customer and Initiator dictionaries.

In addition to a customer or initiator's first and last name, and login ID, it is common to add fields for the person's email address, and home OU. The Person_ID is the unique identifier assigned to the person. Basic form processing does not require use of this field; however, it may be useful in writing data retrieval rules, for example, to dynamically retrieve additional information about the person or his/her supervisor that does not have a corresponding attribute in the dictionary. Similarly, the Supervisor_ID is also selectable for inclusion in the reserved dictionaries. This would allow the service designer to dynamically construct a "chain of command" for approval of chosen requests.

Similarly, you may want to include location information, especially in the Customer dictionary. This information may be required, for example, to determine which queue a task should be routed to, or simply to indicate the person's address, in case contact via an actual physical visit is required.

Automatically Included Integration Dictionaries

The Integration dictionary group is automatically created in all Service Catalog instances. Any dictionary that is created through the Integration Wizard (Service Designer's wizard for creating web services integrations between Service Catalog and external systems) is automatically placed in this group. Once created, integration dictionaries can be moved to any dictionary group. Designers cannot manually place dictionaries in this group.

Creating a Dictionary Group

A dictionary group is a folder that helps organize potentially large numbers of dictionaries. All dictionaries must be placed into a dictionary group. No orphan dictionaries are allowed by the system. Dictionary groups have no behavior associated with them, unlike service groups.

Step 1 Choose **Service Designer** > **Dictionaries**.

Step 2 Choose **New** > **New Dictionary Group**. The New Dictionary Group window opens.

Step 3 Enter the required information in the fields provided and click **Add This Dictionary Group**. The dictionary group is added to the tree menu.

Creating a Dictionary

Before You Begin

- Decide the type of dictionary. Understand the types of dictionaries available in Prime Service Catalog. See Types of Dictionaries in Prime Service Catalog, on page 42.
- For creating external dictionaries, define a data source name for the external database depending upon your application server, so that it can be used as the system dictionary. This is done completely outside of the system.

Step 1 Choose **Service Designer** > **Dictionaries**.

Step 2 Choose a previously defined dictionary on the left, or choose **New** > **New Dictionary** to create a new dictionary.

Step 3 Specify the type of dictionary.

a) To configure Internal Free-Form Dictionaries, click the **Free Form** link in the Add New Internal Dictionary section below the Data Source column.

b) To configure internal person-based dictionary, select **Person Based** from the **Template Based** drop-down list in the Add New Internal Dictionary section below the Data Source column.

c) To configure service item-based dictionary, in the **Service Item** field of the Add New Internal Dictionary section, search for the service item by name, and then click on the service item from the popup window that appears.

d) To configure external dictionaries, in the Add an External Dictionary section, click the name of the data source based on which the dictionary should be created. The New External Dictionary window displays. The login and password for the data source have already been configured on your application server.

Step 4 Add data elements to the dictionary using Table 15: Fields to Configure Internal Dictionary

Step 5 Add dictionary attributes using Table 16: Dictionary Attributes table. To add multiple attributes, click **Add Field**.

Step 6 Click **Save Dictionary**.

To delete a dictionary from the system, choose **Service Designer** > **Dictionaries**, select the specific dictionary you wish to delete and click **Delete Dictionary**.

Note

- While creating a dictionary, values in the Service Form are pre-populated if you have selected **Enable Typeahead** in the Display properties of Active Form Component.

  However, this is applicable only for fields that have DataType as Text and when Input Type field contains either single-select or multiple-select values.

- You are prevented from deleting a dictionary, or a dictionary field, if there are associations with any conditional or data retrieval rules.

Table 15: Fields to Configure Internal Dictionary

Field	Description
Dictionary Name	Enter a name for the dictionary in this field. Allowed characters include: alphanumeric, underscore, and apostrophe. No spaces are permitted.
Group Name	Click the **Update** button to place this dictionary in a Dictionary Group. There is no system behavior related to the association of a dictionary with a dictionary group.
Default Caption	Enter the text you wish to display on the service form section title in this field.
Contact Person	This is the individual who should be contacted to discuss any potential modifications to the dictionary or its use within the context of the organization's site standards and best practices.

Field	Description
Service Item Family	This is an optional text entry field which enables grouping of multiple dictionaries to create a logical entity for the purpose of reporting. For example, three separate dictionaries together may constitute a complete set for Desktop Configuration. **Note** This field is not applicable for external dictionary.
Reportable	When a dictionary is reportable, the ability to modify the dictionary definition is temporarily disabled. The data types of fields in reportable dictionaries cannot be switched between numeric/money, date/datetime, and the character types. If you would like to make any other change to the dictionary, you can switch the Reportable setting to "No", save the dictionary, make the changes, and then restore the setting to its previous value. Set the Reportable field to "Yes" to add the dictionary to those that can be reported against using the Advanced Reporting Ad-Hoc reporting and Report Designer features. If you specify that a dictionary is reportable, then you will see it as a query subject in Ad-Hoc reporting or Report Designer. **Note** There are many implications to be considered in the decision to make a dictionary reportable. See the Cisco Prime Service Catalog Reporting Guide for guidance in the decision to make a dictionary reportable. If you mark an external dictionary as reportable, please be aware that not all data types are supported.
Description	Enter a description of how this dictionary is to be used. This is very helpful information for other service designers who might use this dictionary.
Revision Notes	Ignore this field at setup; later, when making modifications to this dictionary. You can enter revision notes and click **Save Dictionary**.
DBA Notes	Use this field, optionally, to enter database-specific information about the use of this dictionary. This text area is most relevant for external dictionaries, since these connect to external tables which may be maintained by a DBA other than the Request Center DBA.

Field	Description
Dictionary Represents	This field is applicable for configuring External Dictionaries only. Click a radio button to choose either the data in a table (read/write) or the results of a query (read-only). If you click the data in a table (read/write), do the following: **1** Choose the table name from the Table Name drop-down menu. **2** Choose the column in the data table that has been reserved for storage of the requisition number from the Requisition Entry ID Column drop-down menu. **3** Choose the primary search key from the Primary Key Column drop-down menu. If you click the results of a query (read-only), do the following: **1** Enter the SQL statement in the SQL statement field. **2** Click Test SQL Statement to verify the statement.

Table 16: Dictionary Attributes

Field	Description
Name	Name for the field can consist of alphanumeric characters and the underscore and must start with a letter. It cannot contain spaces or other special characters. Reserved words in JavaScript (such as “this”) cannot be field names.
Type	Data Type influences the HTML representation that can be chosen when the dictionary is included in a form. That, in turn, influences how rules and ISF functions may be applied to the field, as well as the usage of the field within the Data Mart, should the dictionary, or a service which includes that dictionary, be made reportable. Choose the data type for the field. See theTable 17: Field Types table for a description of data types and their behavior.

Field	Description
Maximum, Decimals	Enter the maximum length of a string the user can enter into this field. For a number field, enter a number for the number of decimal places to the right of the decimal point. Enter 0 for an integer. For Service Item-Based Dictionary (SIBD), if **Encrypt** is enabled for an attribute the maximum length of a string the user can enter into this field is divided by 2. This is because the SIBD attributes insert form data into a Service Item table and during encryption the extra characters are inserted into the attributes.
Multivalue	Check the check box for Multivalue when you intend to use the INCLUDES operator in an expression to control the execution of a task. The INCLUDES operator compares the values of multivalue data types with other expressions. Multivalue fields map to the following elements: Multi-Select, Checkbox, and Radio button. **Note** DefMultivalue and TxMultivalue records are created for multivalue fields.
Show in Grid	Fields set to “Show in Grid” and “Use” in a dictionary will display as columns in the grid on a service form when the dictionary is set to “Display as Grid”. “Show in Grid” and Multivalue are mutually exclusive, as a grid does not have the capability for a multivalue cell. Grids cannot have more than 20 columns by default, which is defined by “dictionary.attributes.maximum.showingrid.count” property in the newscale.properties file. **Note** Fields in External dictionaries cannot be used in grids and hence do not have the “Show in Grid” column. Also, fields in Reserved dictionaries (Customer_Information and Initiator_Information) cannot be used in grids because those dictionaries inherently represent only one set of data. See Designing Grid Dictionaries for Fields with Multiple Data Instance, on page 70 for more information on grids.

Field	Description
Encrypt	Select the checkbox to encrypt sensitive data such as passwords, beneficiary names, account numbers, and so on. Sensitive information could be exposed while it is being stored, viewed, accessed, logged, or sent to external system and is automatically masked when you encrypt during configuration. The encryption is applicable in the following locations: • Form Data in Service • Service Item Manager > Manage Service Items • Service Catalog > My Stuff • Administration > Utilities > Form Data Viewer • Service Link > View Transactions > Message Type link > nsXML Message / External Message • My Services > Service Items • Server Logs • nsAPI to Read SI data • HTML Source. **Note** You can configure a maximum number of three secure strings for a dictionary. This restriction is because during encryption the resultant value contains some extra characters than the original field value which may impact performance. Therefore, while configuring the maximum length of a dictionary, provide some buffer length to ensure that the encrypted value doesnot exceed the length of dictionary specified. Secure Strings are applicable only for Text field types.

Table 17: Field Types

Type	Description and Active Form Implications
Text	Default data type, supports alphanumeric data; should be used for fields to be rendered as single- and multi-line text.
Number	Data type for all numbers, including integers and whole numbers; it is critical to specify the precision in the Decimals column, as the application will validate decimal precision as well as length.

Type	Description and Active Form Implications
Account	Documentation only; data treated as alphanumeric.
Date	Data compatible with the database's native datetime type, but display restricted to the date; calendar widget supplied for data entry.
Boolean	Data object whose possible values are “true” and “false”, presented as “Yes” and “No”.
Phone	Documentation only; data treated as alphanumeric.
SSN	Documentation only; data treated as alphanumeric.
Money	Data validated to contain only a valid number; monetary symbols or commas cannot be typed; accepts numerical characters and up to 3 decimal places.
Person	Data validated against a person ID in the personnel profiles; a Person Search dialog box is available via a “Search” button automatically rendered on the service form. The Person data type is provided primarily for backward compatibility; if this capability is required, a person-based dictionary should be created.
URL	Data stored as alphanumeric; a saved value is represented both as text and as an HTML representation of the value, providing a link to the specified URL.
Date and Time	Data stored as a date and time; calendar widget supplied for data entry contains a time-selection widget.

Defining Permissions to Edit Dictionaries During Service Requisition

When a dictionary is used in a form, you may specify how access to the dictionary is controlled. Using this procedure you can define permissions per system moment to control which users may view or edit dictionaries in the requisition life cycle. Only those dictionaries in which the initiator must provide the details of the request are typically editable. Any dictionary used solely by approvers, reviewers or task performers have no access assigned. A request may have zero or more Authorization moments, depending on which authorizations and reviews have been configured for that service.

Implications of View capability for a Dictionary

Specifying that a dictionary can be viewed, but not edited, during a particular moment has the effect of rendering all fields in the dictionary as HTML labels (read-only text).

- The text for fields that can have multiple selections (check boxes, multi-select drop-down lists) will show any options previously chosen for those fields as a comma-separated list of values.
- The label for a Person field shows the name of the person previously chosen. The accompanying Search button is disabled. A second (hidden) object contains the unique identifier of the person chosen.
- URLs are rendered as an HTML label that is hyper-linked.

Since HTML DOM objects (that is, <input> tags) do not exist for fields in non editable dictionaries a limited subset of rules or ISF functions can be applied. These limitations are documented in the detailed discussion of conditional rules and ISF functions that follows.

The access levels are applied at the dictionary-level. In a particular moment, the entire dictionary, comprising all its fields, is either hidden, displayed, but with all its displayable fields rendered as boilerplate text; or displayed with its fields rendered as a set of HTML input objects which can be edited by users or manipulated by rules.

Note All access control assignments pertain to all tasks performed as part of the delivery plan. If you need to assign different permissions to dictionaries for different users in different tasks, you will need to do this via a conditional rule. The **Access Control** setting should always grant all permissions that are required on a particular dictionary for a particular user/group. Conditional rules can remove access privileges, but cannot grant them if they are not originally specified via the **Access Control** tab.

Using Access Control Tab for Assigning Permissions

Assigning permissions via the Access Control tab has important consequences for designers specifying active form rules and for ISF programmers:

- **For hidden dictionaries:** Hiding a dictionary via **Access Control** means that no objects defined in that dictionary are present in the service form for the moments and participants for which the dictionary is hidden. Therefore, it is not possible to use rules or ISF to manipulate dictionary or field contents or visibility.

If you need to manipulate the values of fields in a dictionary that are hidden from the user, you can configure the form rules that act upon these dictionaries to be executed on the server side. For more information, see the Defining Form Behavior for Service Item-based Dictionaries, on page 95. For example, you can populate a set of fields with default values without the user being aware of those values.

- **For view-only dictionaries**: Fields in such dictionaries cannot be made editable. However, rules or ISF can be used: to get the current value of a field in a dictionary set to view-only by Access Control; to temporarily hide the field from view; or to apply other functionality provided by the framework.
- **For editable dictionaries**: Full capabilities are available to manipulate the appearance and behavior of the dictionary or fields within the dictionary.

Setting a dictionary's Access Control to “None” (that is, with neither View nor Edit checked) prevents the dictionary from being sent to the browser. However, you can still configure form rules to store data in such a dictionary, provided those rules execute on the server (see the Defining Form Behavior for Service Item-based Dictionaries, on page 95). This is a powerful technique for keeping the service form, as rendered in the browser, as small as possible, while still collecting and storing the results of the browser session in the database.

Administrative Override of Access Control Settings

Access controls assigned via Service Designer are ignored when the service form is run by a user who has been granted the “Manage Service Dictionaries” capability. (This capability is included in the “Site

Administrator" and "Distributed Service Designer" roles.) For such users, all dictionaries are editable in all moments, regardless of the access controls specified.

Tip This capability is provided for ease of developing and testing the form appearance and should be assigned sparingly.

Before You Begin

See Implications of View capability for a Dictionary to understand the behavior of fields in the non-editable dictionary.

- See Using Access Control Tab for Assigning Permissions to understand the implications of assigning edit/view permissions to the dictionaries using Access Control option.
- See Administrative Override of Access Control Settings to see how these dictionaries permission can be overridden.

Step 1 Choose **Service Designer** > **Active Form Components**. Click a form from the Active Form Component tree.

Step 2 Select the form you wish to configure and click the **Access Control** tab.

Step 3 Click on a system moment and click on the dictionary you wish to configure.

Step 4 Click **View** and/or **Edit** for each participant in that moment.

Note If you do not check either View or Edit, then you are effectively granting no permissions to the participant for that dictionary during the specified system moment. This means the participant will not be able to see the dictionary fields at all during the specified system moment.

Step 5 Select the participant type from the **Add Participant** drop-down list. For more information about the participant types and its associated capabilities, see Table 18: Participant Types table.

Step 6 Continue by configuring permissions for all dictionaries during the chosen system moment. Move through the remaining system moments and dictionaries.

Step 7 Click **Save Form**.

You can easily remove a dictionary from an active form component if it is no longer needed, or you can delete a dictionary from the system, thereby removing it from any form components in which it was used.

Table 18: Participant Types

Participant Type	Description
Customer	The user who completes and submits the service order form; typically has View access to dictionaries in the Service Delivery moment, except those that contain confidential information

Participant Type	Description
Service Team	Service performers and managers who are members of the service team OU that "owns" the service group where this service resides. The "Service Team" participant refers to the service team that owns the service group in which the service resides. If you include this form in multiple services, the service team participant for each service may vary, depending on what group the service is in. Therefore, it may be preferable to "**Add Participants**" to the form, explicitly indicating the service teams that are needed to access dictionary data at this time. For services with complicated delivery plans, each task in the plan may need to be performed by a different service team. In this case too, additional participants should be added.
Organizational Unit	Members of the customer's Home OU, such as those who review or authorize service requests.
Financials Team	Refers to the OU, if any, which is configured in Administration as the Financials Team for the purpose of site-wide Financial Authorizations
Ad-Hoc Task Performers	Any performer, or queue members, who receive an Ad-Hoc task

Note The Customer-Initiator form presents additional concerns, since it will likely be used in virtually every service. Therefore, especially if the Customer and Initiator dictionaries should have View only permissions, it might be most efficient to "Add access for Anyone" via the **Additional Participants** option.

Adding Access Control Settings to a Specific Service

Access control that must be granted to a form component used in multiple services may need to vary on a service-by-service basis, depending on the service teams involved in the fulfillment of each request. Additional participants for viewing/editing dictionaries can be added to the service definition. To do so, choose Service Designer > Services > Form, select the dictionary, then add the appropriate Participants. No other changes can be made to Access Control settings at the service level.

Defining Standards in Service Design

Standards aid in the design of active form components, particularly in data retrieval rules that enable customers to drill down to specific answers or choices when ordering services. Standards provide reference data that can be used to validate user entries into a service form or to provide default values for fields on that form. You

can create new standards or import an existing standards data or definitions from a file using Service Item Manager in Prime Service Catalog.

Standards tables perform the same functions as relational database tables maintained in an external datasource—rows can be retrieved either for display (in a drop-down list) or validation (of user-supplied data). Standards tables and external tables differ in these ways:

- Standards tables can be maintained wholly within Service Item Manager—no DBA intervention is required to create the table, modify its structure, or maintain its contents. In addition to providing a user interface for creating, deleting or modifying Standards records, you can also import data from an XML file into a Standards table to create or modify the structure of that table.
- Standards tables can be used directly in table-based data retrieval rules, by choosing the standard as the datasource. They are also available for use in SQL-entry data retrieval rules or in the construction of SQL-based option lists.

Note By default, Catalog Deployer deploys any standards entries when a service deploy a data retrieval that uses the specified standard. You can override this behavior by changing the Administration setting to "Deploy standards entries". This would be desirable, for example, if administrators in the production environment were responsible for maintaining the standards, rather than having all standards defined in a production or test environment.

You can define permissions to view or edit service item standards based on the role assigned to a user. For more information on permissions that can be added, see Service Item Manager module in Assigning Permissions section in Cisco Prime Service Catalog Administration and Operations Guide.

To Define Standards

1. Specify the functional requirements for the Standard table. What data does each table need to contain? How will it be used within Service Catalog?
2. Use Service Item Manager to define the Standard. See Configuring Standards.
3. Populate the Standard table with data relevant to your Service Catalog setup. See Adding Data to the Standards Table.
4. Write data retrieval rules to access the Standard table.

Creating Standards Group

Procedure

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Design Standards**.

Step 2 Click the plus sign (+) on the Standards Table on the left pane and select **New Standard Group**.

Step 3 Specify a Group Name and a description and click **Add**.

Adding Role Based Permissions for Editing Standards

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Permissions**.

Step 2 The Permission Summary window lists the permissions assigned to the selected service item.

Step 3 Click the magnifying glass icon in the Role field, to select a role.

Step 4 Select a permission from the **Permissions To** drop down list and click **Add**.

Configuring Standards

Using this procedure you can create new standard definition and also edit the defined ones.

Step 1 Choose **Service Item Manager** > **Design Service Items** > **Design Standards**.

Step 2 Click the plus sign (+) on the Standards Table on the left pane and select **New Standard**.

Step 3 Add the attributes definitions and click **Add**. The attributes specify the fields (data) that are maintained about the standard. See Table 19: Attributes for Standards.

Step 4 To update existing standard definition, modify the attributes that were defined while creating Standards table and click **Save.**

Step 5 Click **Add in the Column Definitions** pane to add definitions to the table.

- **Display Name:** The name of the attribute that will appear as the attribute label on all Service Catalog pages, including the My Items portlet.
- **Name:** The internal name for the attribute; keywords and reserved words in the underlying database, for example, INTEGER, ORDER, or VARCHAR, cannot be used. Name length is limited to 27 characters.
- **Attribute Type:** The data type for storing attribute data.

All attributes added or updated since the last save are marked by a red triangle at the upper left of the attribute's display name.

If you have standards defined, you can also import these into Prime Service Catalog using **Import Data** tab in Service Item Manager. The details on how to import the standards from an external system, see Importing Standards.

Table 19: Attributes for Standards

Field	Description
Display Name	User-friendly version of the name; it may contain spaces.

Field	Description
Name	Standard name by which the system references the standard and its data. It corresponds to a table that is dynamically created and maintained in the Service Catalog transactional database. A Standard name can contain only alphanumeric characters and the underscore (_), with no embedded spaces. It must begin with an alphabetic character. Service Item Manager creates a database table with the same name as the standard name with a prefix of “St”
Standard Group	Select the group you want to associate the standard table with.
Description	A description about table you are creating

Adding Data to the Standards Table

Service Catalog supports the following ways to add data to standards tables:

- Use the **Manage Standards** tab in Service Item Manager to interactively edit standards data. This tab presents a grid containing all attributes specified for the standard.
- Use the **Import Data** tab in Service Item Manager to import standards data or definitions from a file or a service link. See Importing Standards for details on importing standards from a file or a service link. To know the format of the file that can be imported in Prime Service Catalog, see Import File Format For Service Items and Standards.

Importing Standards

Standards can also be imported using options similar to those available for service items—either the standard definition, the standard entries, or both can be imported. Service Item Manager includes the following methods for importing service items and standards:

- The Import Data option in Service Item Manager allows administrators to import service items or standards on demand from file. See ImportImporting Standards from File.
- A request can include a Service Link task to import service items or standards. See Importing Service Items and Standards using Service Link.

Importing Standards from File

Using Service Item Manager > **Import from File** option you can import data and definition for standards from an external source. The file to be imported must use ANSI encoding—either ASCII or UTF-8. Unicode encoding is not supported. You can select either **Data** or **Definition** option to import only those portions of the file, or you can select both. If you choose to import both and the XML file contains only a definition section, the system only imports the definition.

When importing standards data, all existing data is always overwritten by imported data.

Note The option to import standards data supplements or replaces Catalog Deployer's actions in deploying a service that includes a data retrieval rule that references a standard. By default, Catalog Deployer will deploy both the standard definition and any data previously defined in the source environment to the target environment. This behavior is desirable if standards data does not vary from environment to environment. If this is not the case, you may alter this default behavior by turning off the Administration setting to "Deploy Entries (data) in Standards Tables".

When importing a standards definition, the same conflict resolution options are available that are summarized below.

Table 20: Conflict Resolution for Standards Definition

Definition	Conflict Resolution	Description
Definition	Overwrite	If a standard record exists whose attribute values match all attribute values of the standard being imported then this standard is updated so that standard has values ONLY for the attributes that are specified in the record being imported. This implies that if the standard record being imported has specified values for only some attributes BUT the existing standard in the database has values specified for additional attributes, then those values would be set to NULL.
	Merge	Same as Insert below.
	Insert	If a standard exists whose attribute values match all attribute values of the standard being imported then this standard is not created.

Configuring Service Form Appearance and Behavior Using Active Form Components

A form is a building block for implementing a service. Each orderable service consists of one or more forms. Each form specifies the appearance and behavior of the web page presented to users when they order a service from the service catalog; authorize or review requests for services; and complete the steps required for delivery of the service to its recipient. That web page is called a "service form."

Form rules allow service designers to make service forms a Rich Internet Application (RIA) without having to write code. (RIA means that the application responds immediately to what the user types or what appears

on the screen, without the user having to click a “Submit” button at the end of a screen of input.) The rules allow designers to specify how the service form's appearance and behavior should change in response to user-initiated events.

Some common uses of conditional rules include:

- Enable/disable or show/hide fields based on a radio button (for example, Yes/No selections)
- Mark fields as mandatory based on the value of another field, or during a particular task or moment in the delivery (fulfillment) cycle
- Show/hide entire dictionaries to customize the user experience for different tasks within the service delivery moment
- Set focus on a field to attract the user’s attention
- Validate data for correctness

Dynamic Data retrieval rules provide an online, real-time interface between the service form and information stored in a relational database. These rules allow such data to be displayed in dictionary fields, or interrogated to determine the correctness of items entered by service requester or fulfiller. Some common uses of data retrieval rules include:

- Prefill form data with information maintained via other applications such as a Configuration Management Database (CMDB), ERP, or HR system
- Provide dynamic drill-downs so that the items listed in a drop-down list vary dynamically based on an item previously entered or chosen from another list

Creating an Active Form Group

All active form components must be placed into a form group.

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Choose **New** > **Form Group**.

Step 3 Enter a name for the new form group, and, if desired, a brief description and click **Save**.
The new form group appears at the top of the tree until the page is refreshed, where it is moved to its alphabetical location.

Assigning Group Level Permissions to Design Service Form

Assign object-level permissions for the form group using the Permissions tab. These object-level permissions are types of actions that people, organizational units, groups, functional positions, and roles are allowed to perform. You can grant the following types of permissions:

- DesignForms- Allows the user to design forms for this form group and change group data. These rights are typically reserved for the individuals who design and configure forms in this form group.
- View Forms- Allows the user to view the form group and form definitions, but not change them.

Note New form groups should not be set up to allow "Anyone" to design forms. The ability to "Grant access to Anyone" manually is a time-saving option intended for use once a form group is properly defined and established.

If you had an earlier version of Service Catalog, for each service group, a corresponding form group will be created. The name of the form group will be the prefix "UPGD: Form," followed by the service group name.

Creating an Active Form

Active form components are the building blocks of a service form, and dictionaries, along with active form rules, are the building blocks of a form component.The appearance and behavior of a service form is determined by how the dictionaries and their component fields are configured as part of the active form components that are used in the service definition.

Step 1 Choose **Service Designer** > **Active Form Component**.
Step 2 Choose**New** > **Active Form Component**.
Step 3 Enter a name and brief description for the new form.
Step 4 Click on the form group field and select one of the groups to associate with the form.
Step 5 Click **Save Form**.

After saving the form, you can add dictionaries to the form, and modify and configure form attributes. For more information on how you can add dictionaries and modify form attributes, see Configuring an Active Form.

Configuring an Active Form

Note Reusable form components should only be configured once for use across multiple services. When you modify the form component, the changes are applied to all services using the active form component.

The following attributes can be configured for an Active Form:

- **Form Content**: Dictionaries included in the form and the order in which the dictionaries and fields that comprise the dictionaries are displayed. For more information, see Adding Dictionaries to a Form, on page 65.
- **Display Properties**: How the individual attributes, comprising of each dictionary, are rendered on the web page when a user is working with a service form. For more information, see Defining the Appearance of a Form, on page 66.
- **Access Control**: Which users, or group of users, are able to view or edit specific dictionaries that comprise the service form at a each moment in the requisition life cycle. For more information, see Defining Permissions to Edit Dictionaries During Service Requisition, on page 55.
- **Active Form Rules**: Rules which can conditionally alter the appearance or behavior of the dictionaries or individual attributes displayed on the service form, or can dynamically retrieve data from relational

data sources. For more information, see Configuring Dynamic Form Behaviors Using Form Rules, on page 72.

- **Active Form Behavior**: The events which trigger execution of a conditional rule or a script written using JavaScript in conjunction with ISF. For more information, see Adding Form Rules to a Service Form, on page 93.

Adding Dictionaries to a Form

The first step in configuring a form is to specify the dictionaries that are used in that form, the order and orientation of those dictionaries, and the fields in which each dictionary appears. Use the **Form Content** tab to add dictionaries.

Note When you view dictionaries included on a form on this tab, dictionary names are prefixed with the dictionary group name. Click the plus sign (+) to the left of the dictionary name to display the fields in that dictionary. To change the order in which a dictionary or field is displayed, click the component to move and then click the Up- or Down-Arrow keys at the right of the page until the component is in the desired sequence.

Before You Begin

Following are some design considerations before adding dictionaries to a service form:

- Choose how many, and which, dictionaries to include per form. Associate multiple dictionaries to a single form, only if it is for the same services. For example, if a group of services have a chain of approvals consisting of three approvers, to be chosen by the customer and dynamically determined based on data supplied on the request, the “Approvers” form should include three dictionaries: FirstApprover, SecondApprover, and ThirdApprover.
- Dictionaries do not need to be included in a form in order for rules defined in that form to refer to a dictionary or attribute. Further, a form may contain no dictionaries at all. In this case, the form is a repository for rules. The form must be included in a service that includes other forms which, in turn, include the dictionaries to which the rules refer.
- The same dictionary can be used in multiple forms, provided that only one of those forms is included in a service. However, all active components regarding form appearance or behavior would need to be defined in each form, so this is not an ideal architecture.
- For service bundle:
  - Any rules or JavaScripts attached to form-level events (When the form is loaded or submitted) of the child services are ignored. A form component specifying these rules must be included in the parent service.
  - If a dictionary occurs in multiple child services, it will appear only once in the service bundle. The dictionary's configuration matches the configuration specified in the first child service that includes an Active Form Component which, in turn, includes that dictionary. If the configuration of the

dictionary is different in other services (for example, by the application of service-specific rules), those differences are ignored.

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Click on the form in the Active Form Components tree. Choose **Form Content** and click **Add Dictionaries**.

Step 3 In the Add Dictionaries dialog box, do the following.

a) In the **Name** column, search for available dictionaries and select the one you want to use.

b) In the **Display as Grid** column, check the dictionaries you want to display as a grid. Fields of the dictionary set to **Show in Grid** will display as columns in the grid. Only fields set to **Show in Grid** and **Use** will appear as columns when a dictionary is added with **Display as Grid** checked. Fields set to **Use**, but not to **Show in Grid**, will not appear.

c) Click **Add**.
The dictionaries are added to the form and returns to the Form Content page.

The **Display as Grid** check box on the Content tab appears checked for any dictionary added as a grid. The check box is always dimmed and cannot be changed on the Content tab; its purpose is to show which dictionaries on the form are in grid format. If you inadvertently add a dictionary as a grid, or forget to check **Display as Grid** when adding a grid dictionary to the form, you can remove the dictionary and add it again, with or without **Display as Grid** checked in the Dictionary dialog box.

Step 4 Check the options to show this dictionary when the active form component is used in a bundled service. **Show in Bundle** indicates the dictionary fields appear if the service in this form is attached to is part of a bundle.

Step 5 Click **Save Form**.
Services Used in This Form is a read-only table that automatically populates when the active form component is used by a service. For more information, see Defining Form Behavior for Service Item-based Dictionaries, on page 95.

After a dictionary is added to an active form component, you can configure form fields and form behavior.

If a form uses multiple dictionaries, you can change the order in which they appear on the form by checking the check box to the left of the dictionary name and using the Up- or Down-Arrow keys on the right of the page.

Note The dictionary names on the Form Content tab are a link to the Dictionaries component of Service Designer. If you need to add another field to a dictionary, you can simply Ctrl-Click on a dictionary name from here and go directly to the dictionary.

Defining the Appearance of a Form

The appearance of each active form component is configured on the **Display Properties** tab. On this tab you configure the properties of each dictionary and the fields within. A dictionary field's data type is assigned

when a dictionary is defined, which also defines the field's storage requirements. For more information on fields that can be defined in a dictionary, see Creating a Dictionary, on page 49.

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Select the form you wish to configure and assuming there are dictionaries already added to the form, click the **Display Properties** tab.

Step 3 From the **Dictionaries Used in This Form** section, expand the dictionary whose fields you want to edit.

Step 4 Click on a dictionary name to display its properties on the right.

Step 5 Enter a caption for the dictionary, and click Change Caption, or Set to default caption. If the dictionary has been set to **Display as Grid**, enter the grid settings you want in the Grid Options section.

Step 6 Click on a dictionary field and configure the requirements. For more information, see Table 21: Fields in HTML Representation.

Step 7 Click **Save**.

Step 8 Repeat Step 1 to 7 for all dictionaries configured.

Note To change the order in which dictionaries and fields appear on a form component use the **Form Content** tab.

Table 21: Fields in HTML Representation

Field	Description
Input Type	For a grid dictionary, “Select (multiple)”, “radio”, and “checkbox” are not available in the drop-down list of Input Types, as these controls cannot be rendered in a grid. **Note** Depending on the Input Type chosen, the page will dynamically change the available configuration options. For example, you can choose a maximum or minimum number of characters allowed within a text field, but not for a password field. If the Input Type involves a choice for the end user (for example, radio, checkbox, select), you can also choose whether the choices appear vertically or horizontally on the form, and can configure the choices using the Options List and Options Preselection sub tab. For more information on each of the input types, seeTable 22: Input Types for HTML field Representations table.
Label	Edit the Label. For example, Change "**Select_Person**" to read: "Select Person"

Field	Description
Help Text	Provides direction to the end user as to how to complete the form. For example, "Please enter your 9-digit SSN without any spaces or dashes."
Default Value	Enter a value here if you want to use namespaces to prefill this field.
Generate Unique Value	You can generate a unique ID as the value for any dictionary field with an Input Type of “Text”, “hidden”, or “read-only” by checking the check box “Generate unique value”. When a new service is requested, a universally unique identifier (UUID) is generated and set as the value of the field upon form load for all fields checked “Generate unique value”. Conditional rules and data retrieval rules are executed after this step; so if there is a conditional rule or data retrieval rule that sets the value of this field, then the UUID value may be overwritten by the rule execution.
Validate Range	Check or uncheck this check box if you want the system to validate the user's input. Not all field types support validation.
Mandatory	Check the Mandatory check box to require the end user to complete the field before submitting their order.
Add a Button	Select Add a Button if you want to add a button to the form. Enter a fully qualified URL (including the http://) to take the user to another site when they click the button This control does not appear in a grid dictionary, because you cannot configure buttons in a grid
Editable on server-side	Check the Editable on server-side only check box to set the field to be edited only on the server. Dictionary fields that contain sensitive information, or contain values set by default or with auto-retrieval mechanisms should have this check box checked. This is a security feature that prevents hackers from intercepting and changing data in the browser.
Advanced Formatting	Additional or alternative HTML formatting can be applied to a specific field by using the Advanced Formatting button.This control does not appear in a grid dictionary, because you cannot configure Advanced Formatting in a grid.

Table 22: Input Types for HTML field Representations

Input Type	Description and Active Form Implications
text	Rendered as an HMTL “text box”.
textarea	Rendered as a multi-line text area.
password	Rendered as a password field, where values are displayed or echoed as asterisks.
hidden	Rendered as an HTML text box, but with a display type of “hidden”.
radio	Rendered as an HTML option list; the field has multiple options—the field value is the option currently chosen. Not available for fields set to “Show in Grid”.
select (single)	Rendered as an HTML drop-down box from which the user can choose a single value.
check box	Rendered as a set of HTML check boxes for all the possible options as designed in Service Designer. Not available for fields set to “Show in Grid”.
select (multiple)	Rendered as an HTML drop-down box from which the user can choose multiple values. Not available for fields set to “Show in Grid”.
SSN	Rendered as a single-line text box; “SSN” provides documentation only.
Person	The field is rendered as two objects: a drop-down list displays people and allows users to choose one; a second (hidden) object contains the unique identifier of the person chosen. This HTML representation is automatically applied to the Select_Person field within a person-based dictionary. It could also be applied to a field in a free-format dictionary with a “Person” data type; however, this latter usage is provided primarily for backward compatibility and is not recommended.
URL	Rendered as a single-line text box with a link generated for accessing the URL once the value has been saved.
read-only	Rendered as text—a user will not be able to enter data. The service designer is responsible for providing a value (typically via a form rule or default value display property) to such fields.

Input Type	Description and Active Form Implications
slider	Rendered as a slider component. The Slider component displays the values for text, boolean, money, or number. Also, it displays the maximum and minimum values for money, and number. • **Text**: Use this option to display the values in form of text. • **Number**: Us this option to display numeric values on the slider. Ensure that the **Enable Range Slider** checkbox is unchecked. If it is checked, the values are displayed as range of whole numbers. • **Number Range**: Use this option to display numeric values as range of whole numbers. Check the **Enable Range Slider** checkbox and enter the values for the **Minimum** and the **Maximum** fields. Also, If you want the slider to move only to the specified interval range, enter the value in **Interval** field. For example: if you enter the value 5, you can move the slider only at an interval of 5, 10, 15, 20, 25, and so on. • **Money**: This option is similar to the **Number** option, but you can specify both whole numbers as well as decimal numbers. • **Money Range**: This option is similar to the **Number Range** option, but you can specify both whole numbers as well as decimal numbers. • **Boolean**: Use this option to have only two values on the slider, **On** or **Off**.

Designing Grid Dictionaries for Fields with Multiple Data Instance

A grid is useful when you are designing forms that require multiple data instances of the same fields to be entered in one form. Rather than creating multiple sections of the same fields, you can create one grid with the fields. A grid enables you to create and configure only one dictionary rather than multiple to hold the multiple sets of the same fields.

While configuring a grid, the dictionary is turned ninety degrees so that field labels appear atop the fields. The grid cells are not truly individual fields when rendered in the browser, although they are stored as individual fields in the database.

Difference between grid and non-grid dictionaries

- The Input Types of Radio Button, Check Box, and Select (Multiple) cannot be used.
- Field labels appear as column headers. Advanced formatting cannot be configured for the field labels in a grid.
- Buttons cannot be used.
- Some ISF Dictionary- and Field-Level Functions cannot be used (see Configuring Dynamic Form Behaviors Using Form Rules, on page 72 for details).
- Grid fields cannot be chosen as the “Triggering Field” when creating a data retrieval rule.
- Grid dictionaries and their fields cannot be chosen as the “Triggering Condition” for conditional rules—they can only be “Action” targets.
- A subset of conditional rule Actions are supported; and these apply to the column as a whole, not to individual cells.
- Requisition API (RAPI) cannot be used to submit services containing grid dictionaries.
- The use of grid dictionary fields for Business Engine namespace and Service Link agent parameters is not supported.
- Server-side conditional rules are not supported on a grid.

To Design Grid Dictionaries

Step 1 In **Service Designer** > **Active Form Component** > **Form Content** > **Content tab** choose the fields you wish to have appear in the grid by selecting the **Show in Grid** check box in the dictionary's definition.

- ◦ For a new person-based dictionary, **Show in Grid** is checked by default for the Select_Person, Login_ID, Personal_Identification, Email_Address, and Home_Organizational_Unit fields.

Note “Show in Grid” and Multivalue are mutually exclusive, as a grid does not have the capability for a multi-value cell.

- ◦ Fields in External dictionaries cannot be used in grids and hence do not have the **Show in Grid** column.
- ◦ Fields in Reserved dictionaries (Customer_Information and Initiator_Information) cannot be used in grids because those dictionaries inherently represent only one set of data.

Step 2 Choose **Display as Grid** check box **for the dictionaries** to appear as a grid (except Reserved dictionaries). Only fields set to **Show in Grid** and **Use** columns will appear as columns when this option is selected.

Note The **Display as Grid** check box appears checked for any dictionary you have added as a grid and cannot be changed on the Form Content tab. If you forget to check **Display as Grid** when adding a grid dictionary to the form, you can remove the dictionary and add it again, with or without **Display as Grid** checked in the Dictionary popup window.

Step 3 Set grid options and display properties.

Configuring Dynamic Form Behaviors Using Form Rules

Active Form Rules are defined on the Active Form Rules tab, on the Active Form Components page. The application supports two types of Active Form Rules:

- Dynamic Data Retrieval rules retrieve data stored in a relational database and either display the values returned into fields on the current form, or validate data on the form against the retrieved results. Understanding how
- Conditional rules specify a set of conditions that must be met in order for a set of actions to be carried out. Although referred to as "conditional," these rules are actually applied whenever the specified triggering event occurs

Most of the rule definition is done "declaratively"—that is, rule wizards help you define the rules, by walking you through a series of steps. In most cases, you don't have to write any code at all, just answer a series of questions. When a user orders a service containing the form, the rules are retrieved and code that operates in the context of a web page is automatically generated.

Creating Conditional Rules

Conditional rules allow designers to alter the behavior and appearance of a service form. These are defined in the Active Form Rules tab on the Active Form Components page. After creating the rules, these rules should be attached to a triggering event for these rules to execute.

A conditional rule has the following components:

- The rule name and description
- Conditions under which the actions specified by the rule should be executed (including being unconditional)
- Actions to be taken if all conditions are true

Conditional rules provide a powerful mechanism for applying logic both during the "conversation with the end-user" (that is, during the browser session), and before and after that "conversation" (that is, server-side).

Using this procedure create rules and attach the triggering event to execute the rules.

Before You Begin

- Understand Browser vs. Server-Side Events. See Server-Side Data Retrieval Rule, on page 134.
- Understand how dictionary fields are represented and stored, and how fields in a grid dictionary differ form their non-grid counterparts. See Difference between grid and non-grid dictionaries.

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Select the active form component to which the rule has to be applied, and click **Active Form Rules**.

Step 3 Choose **New Rule** > **New Conditional Rule**.

Step 4 In the first page (Step 1) of the Conditional Rule wizard, formulate a condition by specifying one or more conditional clauses. Each component condition evaluates to true or false. Multiple conditional clauses can be combined, using standard relational rules. You can also make the conditional rule unconditional.

a) Enter a **Rule Name**. This name must be unique on the form component.
b) Click **Add Condition** and choose a type of condition. Depending on your choice, a slightly different .Condition Builder dialog box displays. When choosing dictionaries or dictionary fields, the dictionaries used in your form component are listed first, followed by all dictionaries available in the system. For more information see Table 27: Conditional rules Actions, on page 78 table and Best Practices/Guidelines for Constructing Conditions, on page 76.
c) After each condition is defined, click **OK** to save changes.
If necessary, click **Add Condition** to continue building the statement. When multiple conditions are combined in one rule, you must join them using Boolean operators (or clauses). You may also click on the parentheses () located at the beginning and end of each condition (making the box pink) to clearly define the statement in the full statement condition box below the conditions.

Note If the “is equal to” operator is paired with a person type, the value must be a number corresponding to the person's ID and not the person's name. Do not use “is equal to <blank>” to check for a zero or null value for Number and Money field types. Use “is equal to 0” instead.

d) After all conditions are added, click **Next**.

Step 5 In the second page (Step 2) of the Conditional Rule wizard, add actions that should be executed when the specified conditions are met.

a) Click**Add Action** and choose the type of action. For more information about actions, see Table 27: Conditional rules Actions, on page 78 table.
If the condition set up in the first page of the Conditional Rule wizard is not met, no action will occur. To set up an alternate action, you must create another conditional rule. If you are setting the value of a any field to show a date, the format must be: mm/dd/yyyy. For example, 10/31/2010.

b) After action is defined, click **OK** to save changes and click **Next** to go to the next page.

Step 6 In the last page (Step 3 of 3) of the Conditional Rule wizard, review and save the conditional rule you created. If you need to make edits, click **Previous** to move back through the wizard, or click **Save** and edit the rule later.

a) In the Automatic Associations portion of this page attach the rule to one or more triggering events:

- **Before the form is loaded (server-side):** This server-side rule is executed on the server prior to sending it to the browser.
- **When the form is loaded (browser-side):** This rule is executed when the service form for any service containing this Active Form Component is initially displayed (loaded) in the browser.
- **When the value of any field referred to in the rule's conditions is changed:** In this case, the rule will be attached to either the “When the field is changed” or “When the field is clicked” event, depending on the HTML representation of the field.
- **After the form is submitted (server-side):** This server-side rule is executed on the server after the form is submitted on the browser and sent to the server for processing.

Note Always use the “After the form is submitted (server-side)" triggering event, when the form rule is used, to set the values of form fields that should be strictly controlled based on entitlement or access permissions. This server-side execution of set value action overrides unauthorized data that may be sent to the server through malicious attempts

To edit an existing rule, review or modify the events to which the rule has been attached on the Active Form Behavior tab.

Examples of a Conditional Rule

Here are some conditions that might be useful in a service form, with the actions they might entail:

Table 23: Examples of conditional rule

Condition	Action
Is the current value of the DatabaseType field in the Database dictionary equal to “Other”?	If so, display the Description field in the same dictionary, and require the user to enter additional information.
Are you ordering this service on behalf of someone else?	If so, display additional information about that person (the actual customer for the service).
Is the current user a task performer working on the “Order Memory” task?	If so, display the MemoryDetails dictionary and make entry of all fields required.

Note Grid dictionaries and their fields cannot be chosen as the “Triggering Condition” for conditional rules.

Table 24: Conditions in the Add Conditions list

Parameter	Description/Usage
Dictionary Field	Compare the current value of a field in a dictionary on the service form to a literal field, to the value of another dictionary field, or to a blank or null value. For details on the available operators, see the section on Operators below.
Dictionary	Determine the usage of the specified dictionary in the service form at the current time. A dictionary may be viewable, editable, or hidden (previously hidden by a conditional rule or ISF).
User Name	Compare the login name assigned to the current user to the specified value.
User Role	Check whether the current user has been assigned or not assigned the specified role.
Moment	Check whether the request is in a specific moment in the requisition life cycle. A requisition is also known as service request order. Table 25: Requisition Lifecycle States and Its Values list the values representing requisition life cycle states.

Parameter	Description/Usage
Task Name	Compare the name of the current task in Service Manager to the specified Task Name. The Task Name is set for all authorization and service delivery moments. The task name is blank for moments when a task is not relevant (“Ordering” and “Service Complete”) and when the service form is viewed in Service Catalog, regardless of the current status of the requisition.
Context	The module in which the service form is currently displayed. Valid values are: • Service Catalog • Service Manager • My Services
Service Name	The name of the service
Does unsubmitted requisition exist	A Boolean that returns **true** once the service request has been saved. This is **false** only when the service is initially requested, in the ordering moment. Once the request has been “Added” or Submitted”, it exists.
Is Order on Behalf	A Boolean that returns **true** if the service has been ordered on behalf of another user, and **false** if the requestor is ordering the service for their self.
Customer Role	Check whether the customer for the service has been assigned or not assigned the specified role.

Table 25: Requisition Lifecycle States and Its Values

Service Request Life Cycle Status	Parameter Value
Ordering	ordering
Departmental Authorizations	ouauthorizations
Departmental Reviews	oureviews
Service Team Authorizations	stauthorizations
Service Team Reviews	streviews
Financial Authorizations	financialauthorizations

Service Request Life Cycle Status	Parameter Value
Service Delivery	servicedelivery

Best Practices/Guidelines for Constructing Conditions

The Context Condition: In some cases, the Context may seem redundant—for example, a request is only visible in the Ordering moment in My Services. However, a service form may be viewed in the Service Delivery moment within Service Manager by task performers and within My Services by the user who originally submitted the request. Similarly, a form may be viewable by multiple participants in the Service Delivery moment. You may want to vary the form's behavior or appearance depending on who is doing the viewing, and where.

The Does Unsubmitted Requisition Exist Condition: When a requisition is initiated, it is in the Ordering Moment. At this time any rules or ISF functions that "prefill" form fields with default values are typically executed. The end user may complete data entry and immediately click Submit. In that case, assuming all data in the form is valid, the requisition goes from the Ordering Moment to next moment defined for that service.

In certain circumstances the user may save the requisition without submitting it by clicking "Add and Review". For example, a user may save a requisition if not all data for the requisition is available; if additional services must be added to the same requisition; or if an attachment must be added. The user may edit an unsubmitted requisition by choosing it in My Services.

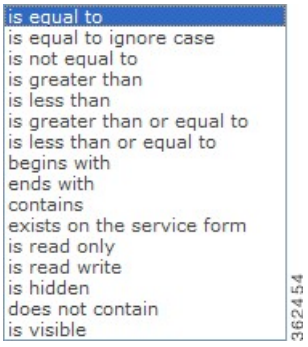
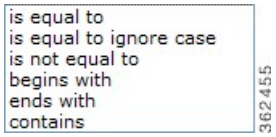
When an existing requisition entry is opened for edit before the requisition is submitted, the "**Does Unsubmitted Requisition Exist**" condition is true; it is false at all other times. Use this condition, for example, to ensure that a rule that prefills default values, is executed only for new requests, not for any requests that have not been saved. This ensures the data the user has previously supplied is not overwritten.

The Moment = Service Delivery Condition: The Service Delivery Moment may encompass many tasks. If a rule is to be conditionally executed for a particular task, use the Task Name condition.

The Task Name Condition: Be careful when referring to a Task Name, since this is a descriptive field that can freely be changed in Service Designer. This problem is minimized if service design guidelines are developed and strictly enforced for naming tasks.

Service designers sometimes include namespaces in task names. For example, the namespace #Name#, referring to the service name, is used to differentiate the same task when it occurs within multiple services. The Task Name used by a conditional rule has all Namespace references properly evaluated. String operations (only checking the beginning or end of the task name, or checking for a phrase contained within the task name) can be used to compare against the portion of the task name that is not derived from the namespace.

Operators in Conditions: The available operators are context sensitive, displayed in a drop-down list that depends on the condition you have chosen. For example, since a field may contain numeric, alphanumeric, or date data, both arithmetic and string operations are allowed, as well as operators to determine the usage of the field. Task and service names are text, so only string operations are appropriate. For conditions that are either true or false, or for which only a limited number of options are possible (such as the current context), radio buttons are available.

Dictionary Field	Task and Service Name	Is Order on Behalf/Does Unsubmitted Req. Exist
		

Most of the operators that can be applied to dictionary fields check the value of that field, comparing it to the value of the specified or literal field. Of these operators, only “is equal to ignore case” is case-insensitive. All other operators are case-sensitive; be sure to take this into account when writing rules such as “starts with” and “contains”, which operate on alphanumeric data.

Other operators, such as “exists on the service form” and “is read only” refer not to the value of the field but to its usage. You do not need to use these rules under most circumstances, since the runtime rule framework automatically checks for the presence of a field on the form and its usage before attempting to execute a rule.

Variables in Conditions: The value of any field in any dictionary may be used in a condition. Since the dictionary may not be included in the current form, the service designer must ensure that forms containing all dictionaries referred to in the rule are included in the service.

Combining Conditions within One Rule: Multiple conditions can be combined and evaluated to determine if the rule’s actions should be executed. In this case, the rule must include Boolean operators (AND, OR) and may include parentheses to alter the normal precedence of these operators.

The conditions essentially build an “if” clause. The Rule Builder does not currently support an “else” clause. If you need to apply if/then/else logic, define two rules with conditions that are mutually exclusive:

Table 26: Conditions versus Actions

Condition	Actions
Rule 1: DatabaseType_OtherDatabase.DatabaseType is equal to Other	Display the Description field in the same dictionary. Make the Description field mandatory.
Rule 2: DatabaseType_DefinedDatabase.DatabaseType is not equal to Other	Hide the Description field. Make the Description field optional. Set the value of the Description field to blank.

Actions are summarized in the table below:

Table 27: Conditional rules Actions

Action	Description of the Action	Notes	Grid Use
Show/Hide	Show/Hide the specified field, grid column, or all fields in the specified dictionary. The elements are displayed with the representations defined in Service Designer.	A frequent use of conditional rules is to show or hide fields or dictionaries based on the current value of another field on the form or any other conditions specified. If you need to hide most, but not all fields in a dictionary, you can hide the dictionary then show only the desired fields. If a field is hidden by a rule, its value is still accessible in other conditional rules.	Applies to entire column

Action	Description of the Action	Notes	Grid Use
Set Value	Set the value of a target field equal to the value of the specified source field, literal, null/blank or the result of an expression. When you set a field value, you need to take into account the field's data type and HTML representation.		Not currently available; use ISF instead

Action	Description of the Action	Notes	Grid Use
		If the value is set to a literal or the result of an expression, the literal or expression can include any numeric or text fields used in the current service. (Date arithmetic is not supported.) The field is represented in the expression using lightweight namespace syntax, that is, *#DictionaryName.FieldName*#. If the expression is invalid (for example, it includes division by zero), the expression is skipped; the value is not updated; and execution of remaining actions or rules for the same event, if any, continues. No error message is displayed to the user. The "Editable only on server-side" setting for a field also affects the behavior of Set Value. A field marked as "Editable only on server-side" is secured against any attempt to intercept its value during the browser session. In the case of person-based dictionaries and service item-based dictionaries with automatic retrieval enabled, the attribute values retrieved from the database will always override the values sent from the browser when the "Editable only on server-side" setting is enabled. For that reason, any Set Value action you wish to apply to such a	

Action	Description of the Action	Notes	Grid Use
		field must be in a rule tied to the post-Submit event—in other words, you can use Set Value to edit the field, but only on a server-side event.	
Set Price	Set the service price or the price of the specified child service within a bundle equal to the value of the specified source field, literal or the result of an expression.	The Set Price action supports dynamic pricing of a service. The price may be set to a constant, the value of another field, or the result of an expression formulated using the same rules as for the Set Value action. Although the Set Price action can be included in a rule triggered by any event, the new price actually takes effect only when the service form is submitted; if the transaction is cancelled, the new price is not recorded. Because the new price takes effect only when the form is submitted, the most appropriate event you should choose for executing a rule containing Set Price is post-Submit. The Set Price action is recorded in the requisition's System History.	Not supported

Action	Description of the Action	Notes	Grid Use
Make Mandatory	Make the specified field mandatory. Making a field mandatory has the same effect as designating a field as mandatory via Service Designer.	• The mandatory symbol appears to the left of the field label. • The user must enter a value in order to successfully submit the form. • If the administrative settings include the form monitor, the monitor will show the dictionary as incomplete until a value is supplied. If you are hiding a field (or dictionary), you must ensure that no fields in the dictionary are set to mandatory; if they are, an error message appears when users try to submit the form. If a rule (or ISF) is used to toggle the field between mandatory and optional, the field's HTML representation should define the field as optional. Using a conditional rule to override a field already marked mandatory in a HTML representation may result in unexpected behavior. If a mandatory field will be toggled between read/write mode, the field should be made optional at the time when it is set to read-only, or disabled to avoid possible confusions to the users.	Not supported
Make Optional	Make the specified field optional.		Not supported

Action	Description of the Action	Notes	Grid Use
Make Read-Only	Make the specified field, grid column, or all fields in the specified dictionary read-only; the field or column cannot be changed by the user, but its value can be changed by a rule or ISF.	In the case of a grid, there is no difference between “Make Read-Only” and “Disable”.	Applies to entire column
Make Writable	Make the specified field, grid column, or all fields in the specified dictionary writeable.	This action is identical to “Enable”	Applies to entire column
Enable	Make the specified field, grid column, or all fields in the specified dictionary writeable.	This action is identical to “Make Writable”.	Applies to entire column
Disable	Disable the specified field, grid column, or all fields in the specified dictionary. Disabling a field or column makes it read-only, like the Make Read-Only action; however, unlike Make Read-Only, it also dims the field or column, and ignores any changes applied to the field via a conditional rule or ISF.	In the case of a grid, there is no difference between “Disable” and “Make Read-Only”.	Applies to entire column

Action	Description of the Action	Notes	Grid Use
Set Focus	Move the cursor to the specified field. This is typically used after an alert or when the form submission is stopped, to direct the user's attention to a problematic field.	Set Focus does not work for flat dictionary right after grid dictionary. Follow these pattern to Set Focus for flat dictionary, • Do some input in the grid dictionary. • Click any field in the flat dictionary. • The first click will not set the focus. • Have to click a second time to set focus on that filed in the flat dictionary.	Not currently available; use ISF instead
Alert	Display an alert box with the specified message; only a literal message can be used—namespaces are not supported. The action does not apply to server-side events in general.	The only valid use of alerts on the server side is when it is coupled with the Stop Submission action in the post-Submit event.	N/A

Action	Description of the Action	Notes	Grid Use
Stop Submission	Does not allow the form to be submitted or updated. Should be used only in conditions where a triggering event is to submit the form.	When the action is executed on the browser side, it does not stop execution of any other actions in the rule, regardless of its sequence in the actions. When the action takes place on the server side, all subsequent actions and rules are skipped and an error message is presented to the end user. An alert action can be defined optionally prior to the stop submission action to provide the appropriate error message. If no alert message is defined, a generic error message is shown to state that the request cannot be submitted.	N/A

Conditional Rules and ISF for Grid Dictionaries

Grid dictionaries are different from nongrid dictionaries in the way they are rendered in the browser and stored in the WDDX generated in the database. Therefore you will need a mixture of conditional rules and ISF functions to manipulate individual cells. This section covers the conditional rule actions and ISF functions that can be used on grids.

Conditional Rule as Applied to Grids

Grid dictionaries and their fields cannot be chosen as triggering conditions for rules; they can only be chosen as the targets of conditional rule actions. The following conditional rule actions which apply to individual fields in non grid dictionaries, apply to an entire column in a grid. All other conditional rule actions are not supported on grid columns or cells.

- Show Fields
- Hide Fields
- Make Read-Only
- Make Writeable
- Enable
- Disable

In case of a grid, there is no observable difference between the Make Read-Only and Disable actions (whereas there is a slightly different user interface effect seen in non grid fields). The same is true for the Make Writable and Enable actions.

ISF in Grids

JavaScript functions cannot be assigned to any field-level event within a grid dictionary. The existing ISF framework has been extended to support grid dictionaries and fields, as described in the tables below.

Table 28: Dictionary-Level Functions:

Function	Usage in Grid
serviceForm.dictionaryName.setVisible (Boolean)	Hides or makes visible the Grid. Has no effect on a dictionary which is hidden via Service Designer.
serviceForm.dictionaryName.isVisible()	Returns true if the Grid is visible and false otherwise.
serviceForm.dictionaryName.getCaption(Boolean stripTags)	Gets the title text for the grid, optionally stripping any HTML.
serviceForm.dictionaryName.setCaption(String newCaption)	Sets the title text for the grid.
serviceForm.dictionaryName.setReadOnly (Boolean)	Sets all the columns in the dictionary to be read-only or read-write; has no effect if in Service Designer the columns were already set to read-only.
serviceForm.dictionaryName.isReadOnly()	Returns true if the dictionary is read-only.
serviceForm.dictionaryName.getGridSize()	Returns the number of rows in the grid.

Table 29: Field-Level (Grid Column-Level) Functions

Function	Usage in Grid
serviceForm.dictionaryName.fieldName. setReadOnly (Boolean)	Makes the column read-only or read-write.
serviceForm.dictionaryName.fieldName.isReadOnly()	Returns true if the column is read-only and false otherwise.
serviceForm.dictionaryName.fieldName.setVisible(Boolean)	Makes the column hidden or visible (displayed). If the column is hidden via Service Designer settings, it cannot be made visible.
serviceForm.dictionaryName.fieldName.isVisible()	Returns true if the column is visible and false otherwise.

Function	Usage in Grid
serviceForm.dictionaryName.fieldName. getInstructionalText(stripTags)	Returns the instructional text for a column, optionally stripping any HTML from the text based on the Boolean stripTags argument.
serviceForm.dictionaryName.fieldName.setInstructionalText(text)	Sets the instructional text for a column.
serviceForm.dictionaryName.fieldName.getPrompt(stripTags)	Returns the column header—optionally stripping any HTML from the prompt based on Boolean stripTags argument.
serviceForm.dictionaryName.fieldName.setPrompt(prompt)	Sets the column header.

Creating Dynamic Data Retrieval Rules

Dynamic Data retrieval rules retrieve data stored in a relational database and either display the values returned into fields on the current form or validate data on the form against those values. They perform this retrieval by executing a SQL query against the source database, returning the values of the columns you have specified to the service form.

Note When a service is ordered through web services, only server-side rules will be executed. If there is any selected field that has options populated by data retrieval rules, the rule should be triggered after submission to ensure that the value passed in the web service request has a list of options to validate against.

Prerequisites to Creating Data Retrieval Rules

- **Define Datasource:** In order for Service Catalog to access a database, a corresponding datasource must be defined. A datasource identifies the database and supplies information for connecting to it, including a valid user name and password. The application comes with one datasource preconfigured, named REQUESTCENTERDS, which provides access to Service Catalog data. A database administrator together with a system administrator needs to define any additional datasources, containing company-specific data, and publish the definitions to the application servers on which Service Catalog is installed. Detailed instructions for configuring and installing datasources are given in the Cisco Prime Service Catalog Installation and Upgrade Guide. Standards and Service Items can be used as a source of data in data retrieval rules. However, their usage is restricted to a rule with the Query Type of "Database Table" are not available for use in rules that use SQL.

When you define a database table lookup, the list of datasources includes Standards (and the Service Items), as well as the transactional and Data Mart databases. If you choose "Standards", the available Standards tables appear that can be chosen as the Table Name for the query. All other aspects of composing the data retrieval rule are identical to those explained previously in this section.

- **Identify the Structure of the Source Data**: You will also need to know the structure of the data you need to retrieve, and tables in which it is stored. An IT specialist knowledgeable in the source system can provide this information.
  - If all of the data you need to retrieve resides in a single table, you can simply specify the name of the table and columns to retrieve. Service Catalog will automatically build a SQL query which retrieves all columns in the table.
  - If you need to retrieve data from multiple tables or to manipulate the values (for example, concatenate values together or perform calculations) for use on the service form, you will need to write and test a SQL query yourself. A database or IT specialist with knowledge of the source system and access to tools for developing SQL are indispensable for this task. Once the query is tested, you can cut and paste it, with minor modifications outlined here, into the Dynamic Data Retrieval Rule Wizard using a Query Type of "Enter Your Own SQL Query".

Use this procedure to define dynamic data retrieval rules.

Before You Begin

- Define the datasource and identify the structure of data to be retrieved as described in Prerequisites above.
- Understand the performance implications of using different data retrieval rules types provided in Service Catalog. For more information, see Performance Considerations Before Choosing Data Retrieval Rules Types.

Step 1 Choose **Service Designer** > **Active Form Components**, and select the active form component to which the rule will apply.

Step 2 Click the **Active Form Rules** tab.and choose **New Rule** > **New Data Retrieval Rule**.

Step 3 In the first page of the Data Retrieval Rule wizard, enter a unique name and a description for the rule, and specify the Rule Type and Query Type.

a) Specify Rule Type:

- Distributing Rule: Choose this option if the primary purpose is to return the results of a query to the form—for example, in a select list, in a set of fields triggered by a selection or data entry in another field, or in a grid. You will be able to attach this type of rule to any number of form- and field-level events.
- Validating Rule: Choose this option if the primary purpose is to validate the data entered on the form against a set of results returned by a query thus ensuring that it conforms to certain standards such as security. This rule can therefore be used to prevent malicious users from intercepting the form and manipulating the content.

Note This rule can be performed on the post-Submit event (server-side) only; you will not be able to attach the rule to any other events.

- Distributing Rule with implicit validation performed on the post-Submit event: This option is checked by default when you create a new rule. Choose this option if the primary purpose is to return the results of a query to the form, but you have an additional need to validate the form data on the post-Submit event. You will be able to attach this rule to any number of form- and field-level events, and a "validating equivalent" of this rule will automatically be attached to the post-Submit event. For example, if you wish to populate a drop-down control with a list of server provisioning locations, and the locations are limited

by the form user's role, using the "Distributing with implicit validation" option means that a malicious user would not be able to specify "Seattle" when the rule excludes "Seattle" as a valid choice for his role.

b) Specify Query Type:

- **Database Table Lookup:** Database Table Lookup is the simple or an easier option. If all the data you want is available within one table (or within a database view that a Database Administrator has created), use this type. The wizard will walk you through a set of screens to define your query, all by filling in dialog boxes. Specify the data source and table in which the data reside. Once you specify the datasource, the drop-down list for the Table Name will be populated with all tables accessible in that data sources.
  - **REQUESTCENTERDS**: Contains the tables in the Service Catalog database.
  - **DATAMARTDS**: Contains the tables in the Data Mart database.
  - **Standards**: Contains the tables you created using Service Item Manager's Design Standards tab or imported using Service Item Manager's Import Data tab.
  - **Service Items**: Contains the Virtual Machine service item, the ServiceItemHistory and ServiceItemSubscription tables, and any service items defined in Service Item.
- **Enter Your Own SQL Query** : Use this query type for more complex query. If you choose this Query Type, you must specify the data source from which the information is to be retrieved, and write the complete SQL SELECT statement to be executed. To reference the value of a field on the form, the query must include the lightweight namespace that refers to that field.

```
For example: select distinct CMN_NAME from ASSET_TRACKING where MODEL = #ST_HARDWAREKIT.ComputerName#
```

Note

- The parser will automatically insert a single quote around the value returned for the namespace at runtime. This is true for all data types except "datetime." The value for a datetime data type must match exactly the datetime format expected by the RDBMS. Each RDBMS may have a different configuration for the datetime format
- If you choose “Database Table Lookup”, the generated SQL query appears at the end of the wizard in the SQL Query field (the field name changes to Generated SQL Query when saved). This gives you a starting-point from which you can then construct your own SQL query, via the second type (that is, you can copy the generated SQL query, then go back to Step 1 of the wizard, choose Enter Your Own SQL Query and paste the generated SQL query into the SQL Statement field).
- REST Web services: Using REST web service dynamic data retrieval rule, the data on the service forms can be fetched from an external system. If you choose a Query Type of **REST Web service**, you must specify the following:
  - Connection Identifier: Select the Unique Identifier or Provide Namespace from the drop-down list.
    Note If you select Provide Namespace, the text box for entering namespace appears.
  - REST URL: Specify the URL of the web resource from where the information has to be accessed using GET and POST operations.
  - REST Method: Use GET or POST method to access the web resource. When using POST method, specify more information in the form of properly formatted xml layouts, or payloads.
  - Username and Password: Basic authentication to access web resources when making a request.
  - Authentication Type: You can access the web resource using the session based and header based authentication, in addition to basic username and password-based authentication mechanism. Session-based name-and-password authentication includes additional authentication token parameter for authenticated HTTP requests.

Step 4 Click **Next**.
If you specified the Rule Type as a **Distributing Rule** or **Distributing Rule** with implicit validation performed on the post-Submit event, you need to choose one or more triggering events.

Note Always use the “Distributing Rule with implicit validation” rule type for form fields that should be strictly controlled based on entitlement or access permissions. The implicit validation prevents malicious attempts to manipulate the field values and to pass unauthorized data to the server. When applying implicit validation to a numeric field that has decimals, you may need to construct the query such that the number of decimals returned by the query will match the user input. Otherwise the validation may fail just because the number of decimal places are different.

Step 5 Select Triggering Event(s) and click Next. The rule can be executed (triggered) by checking the check boxes for the following events. See **Fields on Select Triggering Events page** table.

Note Grid fields cannot be chosen as the “Triggering Field” when creating a data retrieval rule.

Step 6 **Define Lookup Conditions** (**Where Clause**) and click **Next**.

- ◦ For a Distributing Rule or “Distributing Rule with implicit validation performed on the post-Submit event”: Rules with the retrieval type of Database Table Lookup typically need to have an associated “Where” clause as part of the criteria, which specifies which row or rows should be retrieved from the specified table. Any number of conditions may be entered by clicking “Add Where Clause”. As you enter each one, click OK. The condition just specified appears at the top of the page. If you use multiple conditions, all must be true (that is, the conditions will be grouped together using to Boolean operator AND) for a row to be returned.

- If no Lookup Condition is specified, all rows in the specified table will be returned. This typically occurs when you are populating a drop-down list (single-select or multi-select). Rather than attaching such a rule to a Form Load event, it might easier and more efficient to simply define a table-based option list as part of the field's Display Properties.

Note Grid dictionaries cannot be used to create a Lookup Condition.

Step 7 **Define Sort Conditions** and click **Next**.
If you expect the rule to return more than one row, you may want to sort the rows returned. For example, if you are using the rule to retrieve data that is being used to populate a drop-down list, the results should typically be sorted, so they are in an appropriate order. Any number of sort fields (Table Column Name) can be specified, and a Sort Direction (Ascending or Descending) specified for each. Click **Add Sort** to add a new sort field. As you enter each one, click **OK** to add the new field to the Sort Conditions displayed at the top of the page.

Step 8 **Use Lookup results on the Form**
For a Distributing Rule or "Distributing Rule with implicit validation performed on the post-Submit event", at least one distribution target must be defined. A "distribution" defines how the values returned from both table-based lookups and SQL queries are used on the form. Distributions map from column values returned in the query to fields on the service. Each rule may include one or more distributions.

In case of REST web services, services returns JSON response, therefore specify the JSON path from where values are mapped to the dictionary field.

For example, a rule used to populate a drop-down list may have just one distribution (mapping the column to a dictionary field that has the HTML representation of a single- or multi-select element. Alternatively, a rule may have multiple distributions, each mapping from one column to a dictionary field. The target dictionary fields need not be writeable on the form, they can be read-only or hidden. Any number of distributions may be entered by clicking **Add Distribution**. As you enter each one, click **OK**.

A rule cannot distribute results to a combination of grid and nongrid dictionaries, nor to two different grid dictionaries. Once you have chosen one grid dictionary field as a distribution target, any additional targets will be limited to fields in that same dictionary.

The target of the distribution may be a field on a dictionary that is not included in the current form component. It is the responsibility of the service designer to ensure that any dictionary referenced is included in another form component which, in turn, is included in a service with the current form component.

Step 9 Click **Next**.

Step 10 **Validate Field Values** and click **Next**.
If you specified the Rule Type as a Validating Rule, you need to choose one or more fields to validate.

Use this step to choose the fields that will be validated against the columns returned by your query—that is, the query results. The value of each field you specify here will be checked after the form is submitted, against the corresponding query results.If the field's value does not match any of the results, the submission will fail and the end user will receive a message to that effect. Any number of validations may be entered by clicking **Add Validation**. As you enter each one, click **OK**.

A rule cannot validate to a combination of grid and nongrid dictionaries, nor to two different grid dictionaries. Once you have chosen one grid dictionary field as a validation, any additional validations will be limited to fields in that same dictionary.

The validation field may be a field on a dictionary that is not included in the current form component. It is the responsibility of the service designer to ensure that any dictionary referenced is included in another form component which, in turn, is included in a service with the current form component.

Step 11 **Review and Save:**
The rule definition displays on the last page of the wizard. Click Save to save the rule, Cancel to discard the rule (or changes made in this session of the wizard), or Previous to return to a previous page of the wizard. The rule definition may also appear by choosing the rule on the Active Form Rules page.

Fields on Select Triggering Events page

Table 30: Fields on Select Triggering Events page

Field	Description
Form Load	When the service form is initially displayed in My Services or Service Manager
Before the form is loaded (server-side)	This server-side rule is executed on the server prior to sending it to the browser.
After the form is submitted (server-side)	Available for selection only for a Distributing Rule, this server-side rule is executed on the server after the form is submitted on the browser and sent to the server for processing.
Dictionary Field Action	In response to an event that occurs as a user is working with the form, filling out data and moving from field to field.
Dictionary Name, Dictionary Field Name, and Event	If you specify that the Event is a "Dictionary Field Action", you must define that action by choosing the Dictionary Name, Dictionary Field Name, and Event from the drop-down menus. The "triggering" events are similar to, but not identical to, events that web page designers may be familiar with. The list of available events may vary, depending on the Input Type assigned to the field. For example, a radio button has an event "When the item is clicked", which is not applicable to a text field.

Performance Considerations Before Choosing Data Retrieval Rules Types

Though executing a query on the server are faster than queries being called by the browser, it is recommended to execute data retrieval rules on the on-Load event rather than pre-load event. Consider the following if you plan to execute pre-load event than on-load event:

- Anything executing on the server-side (pre-load event) uses cycles on the application server vs. being confined to the user's browser session.
- Executing rules that return number of records affects overall solution performance if executed during a server-side event. Such rules executing on the application server may affect the overall performance of

the solution, for all users. Therefore, refine your query as much as possible, to avoid returning too many results. This will result in the best end-user experience and the best performance.

- Another factor is the perceived performance to the end-user of the form. A complex service form with tens or hundreds of fields will take some time to fully load into the browser window. If that form also contains data retrieval rules that populate drop-downs on the on-Load event, the user is likely to see the form being painted first, followed by the drop-downs being populated. In between the form painting and the drop-downs being populated, the user may see what appear to be "empty" drop-down controls on the form. This effect can be mitigated if you tie the rules populating the drop-downs to the pre-Load event instead. However, moving data retrieval rules to the pre-Load event means that the form does not get loaded—and therefore the user does not see it loaded into the browser—until the rules finish executing. So although the overall performance of the complete loading of the form may be improved, the user's perception may be that the application is slow to respond to clicking the Order button.

You can try to compare the responses by tying a set of rules to the on-Load event and subsequently change them to be tied to the pre-Load event. If you make this comparison while the application is under load, you are better able to see the effects and choose the approach that is most desirable.

Modifying Active Form Rules

In order to modify an existing data retrieval or conditional rule, you must edit the rule, apply the desired changes, and navigate through all pages of the Rule Wizard. The Save button is available only on the last page of each wizard. This process ensures that all aspects of the rule are internally consistent.

Changes to the dictionaries and fields may not automatically be propagated to a rule that uses the corresponding dictionary, field or lightweight namespace. If you change the name of a dictionary or field, you must edit the rule, navigating through all pages. For conditional rules and table-based data retrieval rules, references to the dictionaries or fields are updated automatically as you proceed through each page. For SQL entry data retrieval rules, you must re-enter the SQL code, using correct lightweight namespaces.

If you delete a field or dictionary, you must edit rules to remove references to the deleted object. If a rule that previously worked suddenly stops working, a renamed or deleted object still referenced in the rule is a probable cause.

Adding Form Rules to a Service Form

The Active Form Behavior tab allows designers to attach JavaScript functions and conditional rules to events that occur within the life cycle of a service form. On this tab, you can change the order of conditional rules. JavaScript always executes at the end.

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Click the **Active Form Behavior** tab. The first row in the "Form or Field" column, Active Form Component, is chosen by default. The Triggering Event column lists all form-level events.

Step 3 In the Triggering Event column, choose the form-level triggering event from the available list to which the JavaScript or conditional rule will attach.

Note It is recommended to choose server-side form-level triggering rules as these are secure and less susceptible to interception attacks.

- **Before the form is loaded (server-side):** This server-side rule is executed on the server prior to sending it to the browser. See Managing Service Items on an External System, on page 39 for more information.

- **After the form is submitted (server-side):** This server-side rule is executed on the server after the form is submitted on the browser and sent to the server for processing. See Managing Service Items on an External System, on page 39 for more information.
- **When the form is submitted (browser-side):** The code or rule is executed after the Submit or Update button is clicked, and after any validations specified in the form's Display (such as checking for numeric or mandatory data), but before the form is submitted to the server.
- **When the form is loaded (browser-side):** The code or rule is executed when the service form is initially displayed in Service Catalog, Service Manager, or My Services.
- **When the form is unloaded (browser-side):** The code or rule is executed when the service form is closed.

Note JavaScript functions cannot be attached to server-side triggers.

Step 4 Click **Add JavaScript** or **Add Rules**. A dialog box lists all scripts or rules previously defined.

Step 5 Choose the functions or conditional rules previously defined by checking the checkbox next to them, and then click Add. You can use the Search box, if needed.
In the Behavior column, the functions appear below the JavaScripts section, and the rules appear below the Rules section. Review the order in which the rules are to be executed, and change the sequence if required.

Note Although you can attach multiple JavaScript functions to the same event in a single form, this practice is best avoided because the order in which the functions are specified (and executed) cannot be defined. Therefore, if you need functions to execute in a certain order, create one JavaScript function that contains all functions in the desired order.

Step 6 If required, you can edit arguments for a JavaScript function by clicking on a JavaScript function in the Behavior column, and then clicking **Edit Arguments**.

Adding Form Rules to a Service Form Field

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Click the **Active Form Behavior** tab.

Step 3 Click the field you want in the “Form or Field” column. The Triggering Event column lists all field-level events.

Step 4 In the Triggering Event column, choose the field-level triggering event from the available list to which the JavaScript or conditional rule will attach:

- **When the item is changed:** The user enters something into the specified form field.
- **When the item loses focus:** The user exits from the specified form field, either by tabbing to another field or clicking the mouse in another field.
- **When the item is clicked:** The user clicks on a check box, radio button, or item in a drop-down list.
- **When the item is focused on:** The user clicks in the specified field.
- **When the mouse is moved off the item:** The user moves the mouse off the specified field.
- **When the mouse is moved on the item:** The user moves the mouse onto the specified field.

Step 5 Click **Add JavaScript** or **Add Rules**.

Step 6 Choose the functions or conditional rules previously defined by checking the check box next to them, and then click **Add**. You can use the Search box, if needed.
In the Behavior column, the functions appear below the JavaScripts section, and the rules appear below the Rules section.

Step 7 Review the order in which the rules are to be executed, and change the sequence if required. If required, you can edit arguments for a JavaScript function by clicking on a JavaScript function in the Behavior column, and then clicking **Edit Arguments**.

Note Although you can attach multiple JavaScript functions to the same event in a single form, this practice is best avoided, because the order in which the functions are specified (and executed) cannot be defined. Therefore, if you need functions to execute in a certain order, create one JavaScript function that contains all functions in the desired order.

Defining Form Behavior for Service Item-based Dictionaries

Form rules and other form behavior for service item-based dictionaries (SIBDs) work almost in the same way as free-form dictionaries. Once an SIBD has been defined, it can be included in an active form component. The procedure for doing so is the same as for including any dictionary in a form component—on the Form Content tab, click **Add Dictionaries** and choose the dictionary from the popup search window. If desired, you may change the display order of dictionaries or fields in the form component.

Configuring Display Properties for Service Item-Based Dictionary

An SIBD has one unique property—the ability to automatically retrieve and prefill data about an existing service item instance. For a chosen form, this option is found on the Display Properties tab for the service-item-based dictionary.

"Enable automatic retrieval of service item instance data" should typically be checked if the service is to be used to update or delete an existing service item. To take advantage of this prefill capability, you may need two form components based on the same service item dictionary—one to be included in services where the item is created and the second to be included in services that update or delete the item.

Using Service Items in Dynamic Data Retrieval Rules

Service items are available for use in table-based dynamic data retrieval rules.

Step 1 Create the rule and specify its name, description, and triggering event.

Step 2 Choose "Database Table Lookup" as the Query Type.

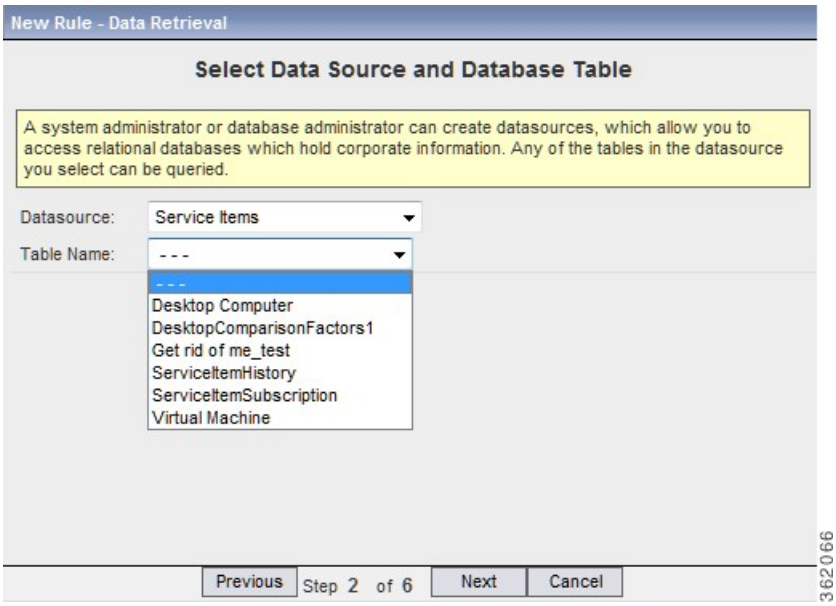
Step 3 The second page of the Rule Wizard appears. You can then choose "Service Items" as the Datasource.

The drop-down list for Table Name is populated to include:

- Any service items defined in Service Item Manager supplied with all Service Catalog installations
- ServiceItemHistory, a table automatically maintained to track the history of a service items
- ServiceItemSubscription, a table automatically maintained to track subscriptions (the current status) of service items

You can then proceed to complete the rule definition as you would for any table-based data retrieval rule. For details on defining rules, see Configuring Dynamic Form Behaviors Using Form Rules, on page 72.

Figure 3: New Rule



You can use a service item in a SQL-entry data retrieval rule by referencing the database table name which the system assigns to the service item. The table name is the prefix “Si” followed by the item name, with spaces removed. An easy way to find out the database table name (without returning to the Manage Service Items page to look up the item’s name) is to define and save a table-based rule using the service item. The generated SQL on the Summary page includes the table name.

Displaying Service Form as a Wizard

In the Service Catalog module, you can customize the display of service form with dictionaries in the form of a wizard. The wizard shows form fields in multiple pages with previous and next navigation controls. By default, all the dictionaries for a service are listed in a single page. You can configure the wizard for an individual service which helps to present the dictionaries in an organized manner, particularly if the service form contains many dictionaries.

The wizard is rendered as steps based on Active Form Components or dictionaries within the service. The wizard configured for the service has a carousel that displays all wizard steps and allows the user to navigate

to any step directly without having to page through the steps sequentially. Navigation controls and pricing information are displayed at the bottom of the page along with the action buttons that exist in service forms.

Step 1 Choose **Service Designer** > **Services** > **General**.

Step 2 For the field **Pagination View Mode**, select one of the pagination view options listed in Table 31: Pagination View Options, on page 97.

Table 31: Pagination View Options

Option	Description
Classic View	This is a default setting, i.e., no pagination is applied to the service form and contains all the dictionaries for the service.
Dictionary Pagination	Displays one wizard step per dictionary present in the service form.
Form Pagination	Displays one wizard step per Active Form Component present in the service form.
Custom Pagination	You can customize the service form to display a set of service pages with selected dictionaries associated to them. One service page with selected dictionaries is displayed as a set of wizard steps.

Step 3 Scroll down and click **Save**.

Customizing the Service Form Using Custom Pagination

To customize the service form using the **Custom Pagination** option perform the following additional steps.

Step 1 Click the icon next to the **Pagination View Mode** (this icon is available only when you select the option **Custom Pagination**).

Step 2 In the pop-up dialogue box, provide a name for the page in the **Page Name** field.

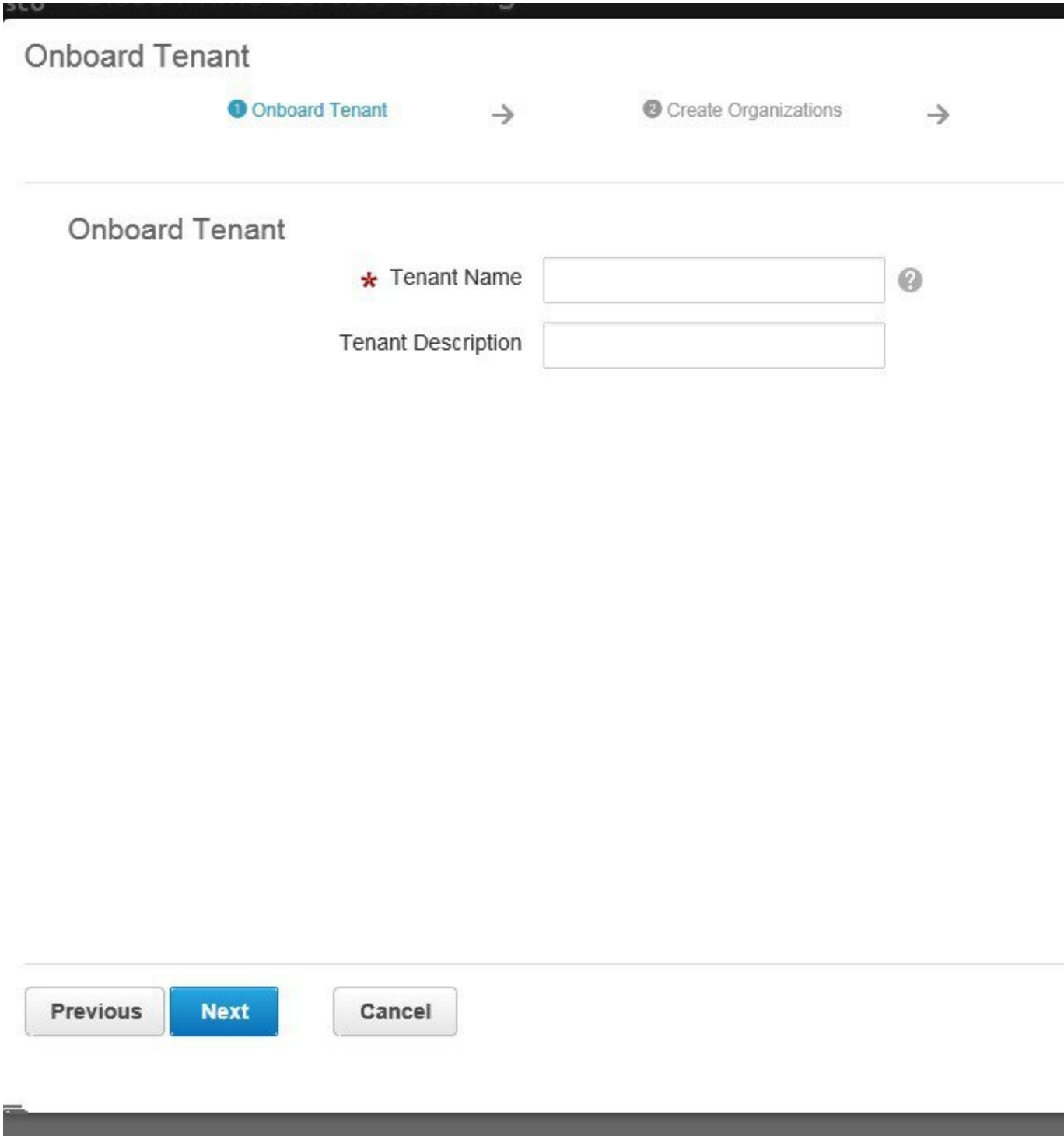
Step 3 Select the required dictionaries from the Available Dictionaries box and click the arrow to move them to the Selected Dictionaries box.

Step 4 If required, add more pages or remove selected page, using **Add New Page** and **Remove Selected**.

Step 5 Click **Save Page** and close the dialog box.

The figure below is an example of a customized service form with customized pagination for dictionaries in the Service Catalog module.

Figure 4: Sample Service Form with Custom Pagination


Onboard Tenant
1 Onboard Tenant
2 Create Organizations
Onboard Tenant
* Tenant Name
Tenant Description
Previous
Next
Cancel

Enabling Order Summary in the Service Form

The show order summary functionality allows you to add a summary page at the end of the paginated service form. This summary page is a read-only page that lists all the fields of the wizard. To enable the summary page:

Step 1 Navigate to **Service Designer > Services > General**.

Note When the Pagination View Mode option is specified other than Classic View, the Show Order Summary option is displayed.

Step 2 Check the **Show Order Summary** option.

Step 3 Scroll down and click **Save**.

Interactive Service Forms (ISF) API Overview

ISF is a set of interfaces and techniques to add JavaScript programming to a service form. JavaScript programming can only be executed when the service data is displayed—when the service is being ordered in My Services or on the Task Details tab in Service Manager.

- When Should You Use ISF and JavaScript?, on page 100
- ISF Components, on page 101

When Should You Use ISF and JavaScript?

JavaScript programming, including ISF, is meant to supplement the capabilities provided by active form rules. The most common behaviors associated with enhancing the interactivity of a service form—dynamically showing and hiding fields or dictionaries; setting field values based on the context or on data previously entered; making fields read-only or editable, mandatory or optional—can be provided by the active form rules. The server-side events (pre-Load and post-Submit) cannot, therefore, execute JavaScript.

Writing ISF and JavaScript requires technical (programming) expertise as well as the use of additional tools to debug your code, access the application server and maintain source code control. Consequently, ISF code is more expensive both to develop and maintain than equivalent active form rules. This technology should be used primarily if the desired functionality cannot be implemented via the declarative rules. Some examples are given in the following sections. These use cases typically fall into the following areas.

- Manipulating objects on the service form not accessible to the rules, such as dictionary captions, field labels, instructional (help) text, and cells in a grid dictionary.
- Performing date or numeric arithmetic—for example, computing a scheduled start date based on the date the service was ordered, or computing a service price, based on components chosen.
- Accessing commercially available, freeware-distributed or custom developed JavaScript libraries for specialized functions such as encryption or decryption, or accessing another application via a web service or server-side (AJAX) code.

Some ISF components duplicate functionality available via the active form rules. If your application's requirements can be completely met by using the form rules, that is usually the most effective way of coding. However, in order to ease maintenance of many, complex rules you might decide to implement some equivalent functionality using ISF in the following scenarios.

- Since the rules do not fully support if/then/else logic (they only support the “if” part), two rules are required for every “if” condition with an “else” clause. Once the conditions get more complicated, for example, you need nested “If statements,” the number of rules required to cover all cases would increase rather markedly. Rather than writing and trying to keep track of all those rules, it might be easier in the long run to write one ISF function, which can include nested if statements.
- Rules are bound to one particular Active Form Component (AFC), while JavaScript functions/ISF are callable from any AFC. ISF could be used for a complex piece of code that needed to be callable from many AFCs, for example, if the same dictionary was used in two different AFCs or the same piece of code needed to apply to two fields.
- If ISF needed to be performed in conjunction with some actions that could be done in rules, for example, change a field label or help text, you might consider coding the entire thing in ISF. This is because of the difficulty in ordering ISF and rules—the rules can be explicitly ordered, but the ISF must follow all the rules for the same event.

You can freely combine ISF and the declarative rules in the same form, even on the same event. The most critical limitation to be aware of in using ISF to replace or supplement actions available in the rules is that JavaScripts must be run after any rules that are attached to the same event.

ISF Components

ISF includes global identifiers as well as a set of public functions.

Global Identifiers

ISF global variables and their possible values are summarized in the table below and discussed in more detail in the following paragraphs. When these identifiers have an equivalent condition in the conditional rules, that equivalence is noted. More details may be found in the preceding sections on Active Form Components.

Table 32: ISF Global Identifiers

Global Variable	Description
Context	The module in which the service form is currently displayed. Equivalent to the “Context” condition.

Global Variable	Description
EditRequisitionBeforeOrdering	A Boolean that returns **true** in the ordering moment when an existing (previously saved) Requisition Entry is being edited and **false** under all other conditions. An alternative way to discover this condition is to evaluate (ReqID==0), since a RequisitionId is assigned when a requisition is saved. Equivalent to the condition “Does unsubmitted requisition exist?”
Moment	The current moment in the requisition life cycle. Equivalent to the “Moment” condition.
ReqCustomerID	The unique identifier of the customer for the service. It may be different from the user that is making the request. The ReqCustomerID value is available in all moments. This is a reference to the Person’s unique identifier in Service Catalog.
ReqEntryID	The Requisition Entry (also known as Service Request) ID. The ReqEntryID is zero (0) until the requisition has been saved.
ReqID	The Requisition ID (also known as Shopping Cart). The Requisition ID is zero (0) until the requisition has been saved.
ReqInitiatorID	The unique identifier of the person that initiated the service request. The ReqInitiatorID value is available in all moments. This is a reference to the Person’s unique identifier in Service Catalog.
ServiceID	The unique ID of the service; included for backwards compatibility; should not be used for services deployed via Catalog Deployer, which does not preserve entity IDs between sites.
ServiceName	The name of the service. Equivalent to the “Service Name” condition.
TaskID	The ID of the task being viewed in Service Manager. The TaskID is set for all authorization and service delivery moments. The TaskID value is zero (0) for moments in which no task is active, typically before the authorization or service delivery begins.
TaskName	The name of the task being viewed in Service Manager, with all Namespace references properly evaluated. Equivalent to the “Task Name” condition.

Global Variable	Description
UserID	The person that is making the request, or, when the context is Service Manager, the person that is working with the request.

Person References

All users must be registered in Organization Designer. The application identifies these users by assigning a unique identifier to their records in Organization Designer. The application tracks the initiator of the current requisition (ReqInitiatorID); the customer for the current requisition (ReqCustomerID); and the user currently working with the requisition (UserID).

In the ordering moment, the ReqInitiatorID is always equal to the UserID. In service delivery and authorization/review moments, the UserID identifies the task performer or reviewer. The ReqCustomerID is different than the ReqInitiatorID if the Order On Behalf (OOB) capability is used; otherwise, these values are identical.

JavaScript Functions

JavaScript functions are built into the ISF framework. ISF is an object-oriented framework. The ISF JavaScript functions are actually methods which are based on the base object serviceForm. To hide a dictionary, for instance, the user calls

```
serviceForm.DictionaryName.setVisible(false).
```

To hide a field the user calls

```
serviceForm.DictionaryName.FieldName.setVisible(false).
```

Tip In the following sections, bold and italicized typeface means that the programmer should substitute the name of the item.

Note JavaScript functions cannot be assigned to any event of grid dictionaries and their fields.

Dictionary-Level Functions

- Dictionary Permissions and ISF Dictionary-Level Functions
- Dictionary Permissions and Administrative Users
- Checking for the Existence of a Dictionary

For grid use, see the Designing Grid Dictionaries for Fields with Multiple Data Instance, on page 70.

Dictionary Permissions and ISF Dictionary-Level Functions

The appearance (and HTML representation) of a dictionary specified as read-only via Service Designer (for a specified moment or set of participants) is different from the appearance (and HTML representation) of a dictionary that is set to read-only via the ISF *dictionaryName***.setReadOnly()** function.

- When the dictionary is set to Edit only via Service Designer, no HTML input tags are generated for the fields which comprise the dictionary; they are rendered on the service form as text.
- When the dictionary's Access Control includes Edit capability and it is set to read-only via ISF or a conditional rule, the dictionary fields are displayed as input objects; however, they are not enterable.

When a dictionary is read-only at design time (that is, does not have Edit permission for the current participant and moment as specified in the Access Control tab for the Active Form Components in Service Designer) it cannot be made writeable through ISF or rules. This is true because when the read-only dictionary is rendered, the resulting HTML includes text with <span ../> tags; HTML <input ..> tags are not present.

Dictionary Permissions and Administrative Users

Dictionary permissions are ignored for a user who has been assigned the “Manage Service Dictionaries” capability. (This capability is automatically granted to any users who are in the “Site Administration” organization and may be included in user- and Service Catalog-defined roles as well.) The dictionary will appear as if it were editable. Care should be taken to not test any ISF when logged in as an administrative user, since the “Manage Service Dictionaries” capability overrides the designated dictionary permissions.

Checking for the Existence of a Dictionary

The ISF expression:

```
serviceForm.dictionaryName
```

is not a function which returns a Boolean. The *dictionaryName* is an attribute of the serviceForm object. The **dictionaryName** attribute has a value of true if the dictionary exists in the service in which the ISF is executed, or undefined if the dictionary does not exist. Therefore, robust code that checks for the existence of one or more dictionaries and takes action only if the dictionaries are present in the current service might be coded as follows:

```
AIT_Server_onLoad() { if (serviceForm.RC_CUSTCODES != undefined)     {RC_CUSTCODES_onLoad();)

  if (serviceForm.RC_PERFORMWORK != undefined)      {RC_PERFORMWORK_onLoad();)
}
```

Field-Level Functions

- Checking for the Existence of a Field
- Getting the Value of a Field
- Setting a Field to Read-Only
- Setting a Field to Visible
- Setting the Value of a Field

Functions that are applicable to fields are summarized in the table below and explained in more detail in the following paragraphs.

Table 33: Field-Level Functions

Function	Usage	Grid Use
serviceForm.**dictionaryName.fieldName**	Returns the field object if the field exists in the form, and undefined otherwise.	No
serviceForm.**dictionaryName.fieldName**.getValue()	Returns the current value of the field.	No
serviceForm.**dictionaryName.fieldName**.setValue(inputValues)	Sets the value of a field.	No
serviceForm.**dictionaryName.fieldName**.getCellValue (RowIndex)	Returns the cell value of the grid column at specified RowIndex.	Yes
serviceForm.**dictionaryName.fieldName**.setCellValue (RowIndex, inputValue)	Sets the cell value of the grid column at the specified RowIndex. inputValue takes a single value instead of an array.	Yes
serviceForm.**dictionaryName.fieldName**.setValue(inputValues, defaultValue)	Sets the value of a field. inputValues has to be an array, the first value of inputValues is assigned to any single-option field. The defaultValue is applicable only for a field with an input type of text or text area. For any other field type the defaultValue is ignored.	No
serviceForm.**dictionaryName.FieldName**.setSelection(inputValues)	Set the value of a field with an HTML representation of select (single), select (multiple), check box or radio button.	No

Function	Usage	Grid Use
serviceForm.**dictionaryName.FieldName**.setMandatory(Boolean)	Makes the field mandatory or optional. If the field is mandatory via Service Designer settings, it cannot be made nonmandatory. When a field is made mandatory, validation is automatically assigned; an error message, "Required: Please fill out this field before submitting" is displayed if the form is submitted with this field blank.	No
serviceForm.**dictionaryName.fieldName** .isMandatory()	Checks it the field has been marked mandatory. Returns true if the field is mandatory, either via the ISF setMandatory() function, a conditional rule, or a Service Designer Display setting.	No
serviceForm.**dictionaryName.fieldName** .setReadOnly(Boolean)	Makes the field or grid column read-only or read-write.	Yes
serviceForm.**dictionaryName.fieldName** .isReadOnly()	Returns true if the field or grid column is read-only and false otherwise.	Yes
serviceForm.**dictionaryName.fieldName** .setVisible(Boolean)	Makes the field or grid column hidden or visible (displayed). If the column is hidden in Service Designer settings, it cannot be made visible.	Yes
serviceForm.**dictionaryName.FieldName**.isVisible()	Returns true if the field or grid column is visible and false otherwise.	Yes

Function	Usage	Grid Use
serviceForm.**dictionaryName.FieldName**.setFocus(Boolean)	Sets focus to the field; applicable to all input types except for radio buttons and check boxes. True: focuses the field; False: blurs the field. This function does not apply for fields that are in dictionaries set read-only via Service Designer or for hidden fields—the call is ignored and no error is shown.	No
serviceForm.**dictionaryName.FieldName**.setFocusViaValidation(Boolean)	Sets focus to the field, applicable to radio buttons and check boxes only.	No
serviceForm.**dictionaryName.FieldName**.getInstructionalText(stripTags)	Returns the instructional text for a field or grid column, optionally stripping any HTML from the text based on the Boolean stripTags argument.	Yes
serviceForm.**dictionaryName.FieldName**.setInstructionalText(text)	Sets the instructional text for a field or grid column.	Yes
serviceForm.**dictionaryName.FieldName**.getPrompt(stripTags)	Returns the prompt for a field or grid column header—optionally stripping any HTML from the prompt based on Boolean stripTags argument.	Yes
serviceForm.**dictionaryName.FieldName**.setPrompt(prompt)	Sets the prompt for a field or grid column header.	Yes

Checking for the Existence of a Field

The ISF expression:

`serviceForm.dictionaryName.fieldName`
does not return a Boolean. The *fieldName* is an attribute of the serviceForm.*dictionaryName* object. The *fieldName* attribute returns undefined if the field does not exist in the current service. (The field may be hidden,

and it is still considered to exist.) Therefore, robust code that checks for the existence of one or more dictionaries and takes action only if the dictionaries are present in the current service might be coded as follows:

```
RC_REQUESTEDBY_onLoad() {  if (serviceForm.RC_REQUESTEDBY.FirstName != undefined)
{serviceForm.RC_REQUESTEDBY.FirstName.setReadOnly();)
  if (serviceForm.RC_REQUESTEDBY.LastName != undefined)
{serviceForm.RC_REQUESTEDBY.LastName.setReadOnly();)
}
```

Checking for field existence is not necessary if code has previously confirmed that the dictionary “exists”.

```
commonOnLoad() {
  if (serviceForm.RC_REQUESTEDBY != undefined) {
    RC_REQUESTEDBY_onLoad();
  }
  ...
}
RC_REQUESTEDBY_onLoad() {  serviceForm.RC_REQUESTEDBY.FirstName.setReadOnly();
  serviceForm.RC_REQUESTEDBY.LastName.setReadOnly();
}
```

Getting the Value of a Field

The getValue() method (serviceForm.*dictionaryName.fieldName* .getValue()) always returns an array. If there is only one item then it is an array of one. Use .getValue()[0] to access the first element. The getValue() method works on all fields, regardless of whether the field is read-only, read-write, or hidden in the current moment, provided that the dictionary is defined with edit access.

- For input types like check box and Select (Multi), getValue() is processed incorrectly if the values contain tabs (for example, if an attempt was made to import data from an external source where tab is used to delimit distinct values in lists). The tab character is represented within the value as \t.
- For complex controls (radio, check box, multi-select, single-select/drop-down) the value returned is the set of “selected values”—meaning only the highlighted values are returned and the “Value” property is used rather than the label or text often seen by the user.

For security reasons, this method does not work for a field with an input type of password. It returns a blank string—no error is shown.

Setting a Field to Read-Only

The setReadOnly() method (serviceForm.*dictionaryName.fieldName* .setReadOnly()) toggles a field between read-only or read-write.

- Fields with the input types of person, date or datetime have an associated button that the user clicks to choose a value for the field. Setting these fields to read-only disables the Select button next to the field so it cannot be clicked. The text field containing the descriptive information (person's name, date, or datetime) is always read-only.
- When a dictionary is marked as read-only for a particular moment in Service Designer, fields appear on the service form as boilerplate text; no HTML input object is generated. Such a dictionary (or fields in the dictionary) cannot be made read-write through ISF. Attempts via ISF setReadOnly() to make the dictionary or a field in the dictionary writeable silently fail. Similarly, attempts via ISF to make the field or dictionary read-only have no effect and do not generate an error.
- If a dictionary or field is made read-only through ISF, the HTML input box is still displayed but field contents cannot be edited and the field is removed from the tab sequence.

Setting a Field to Visible

The setVisible() method (serviceForm.*dictionaryName.fieldName* .setVisible()) toggles a field between being hidden and visible on the service form.

- When a dictionary is marked as hidden for a particular moment in Service Designer, fields in that dictionary cannot be made visible via ISF.
- When a dictionary has been hidden via ISF, attempts to make fields in the dictionary visible silently fail.

Setting the Value of a Field

Two methods are available for setting the value of a field:

- serviceForm.*dictionaryName.fieldName* .setValue())
- serviceForm.**dictionaryName.fieldName**.setSelection())

A third method is the equivalent of the setValue() method, but for cells in a grid:

- serviceForm.dictionaryName.fieldName.setCellValue())

See the Configuring Dynamic Form Behaviors Using Form Rules, on page 72 for more details.

The setValue() method sets the value of the specified field to the specified inputValues. inputValues has to be an array; the first value of inputValues is assigned to any single-option field.

- For security reasons, this method does not work for fields with an input type of password.
- inputValues for Select (single), Select (multiple), check box and radio button input types are set to the display values and this function selects the same. Check box type fields can take an array of inputValues. The function returns without selecting when invalid display values are passed that are not in the field. On saving the service form, the selection is persisted.
- No validation is performed on the inputValues. It is possible for wrong values set through this function to be persisted when the service form is submitted, even for field types like Person, SSN, URL, or Date.
- For Person type fields, see the Specialized Field-Level Functions, on page 109 section below.

The setSelection() method should be used for fields with multiple options—select (single), select (multiple), check box, and radio button. The argument is matched to the values for the various elements in the field. For a radio field or check boxes, this function allows you to set the selection to one or none of the controls. For example: .setSelection(['']) clears all the radio buttons and restores the “pristine state”.

For a multi-select or a single-select input item, setSelection marks the requested items selected. Items that are not found in the list are ignored.

This function does not do anything for fields with an input type of password, text, or textarea, or for fields in read-only dictionaries.

Specialized Field-Level Functions

Some field-level functions are applicable only to fields of particular types. These are summarized in the table below.

Table 34: Special Field-Level Functions

Function	Usage
serviceForm.*dictionaryName.fieldName_disp* .getValue()	Applies to Person type fields only. fieldName.getValue(inputValues) gets the PersonID of the specified field. The display value of the Person field (generally the person's name) can be accessed by suffixing '_disp' to the fieldname: ...fieldName_disp.getValue().
serviceForm.*dictionaryName.fieldName_disp* .setValue(inputValues)	Applies to Person type fields only. *fieldName* .setValue(inputValues) sets the PersonID of the specified field. Use *fieldName_disp* .setValue(inputValues) to set the value displayed in the text box.
serviceForm.*dictionaryName.fieldName_saved* .getValue()	For single- and multiple-select fields, the normal fieldName.getValue() always returns the value that is currently selected. The saved values for Select type of fields can be accessed by suffixing "_saved" to the actual field name –. *fieldName_saved* .getValue() This function can be useful to determine the values that was saved in previous moments for Select lists. This function also returns values previously saved in the same moment. Although *fieldName_saved* .setValue() executes without error, it is nonfunctional, and does not change the saved value of the field. **Note** This function is not available for grid dictionaries.

Person-Based Fields

The Person Data type and HTML representation are designed for the display and validation of person profile data stored in Organization Designer. The field appears on the service form as a single-line text box with an associated control labeled “Select”, as shown below.

Figure 5: Select Option

Clicking the “Select” button opens a popup window allowing the user to search for people in the Service Community, and choose the person of interest. The person’s name and email address (fields configured in Site Administration) are displayed on the service form.

ISF functions operate as follows when applied to a person-type field.

- *personFieldName* .getValue() returns the ID of the person, the unique identifier for the person record in Service Catalog. That ID can be used, with appropriate, customized, server-side code to lookup additional person profile information for display in other fields on the service form.
- *personFieldName_disp* .getValue() returns the name of the person as currently displayed on the service form.
- Using the Select control automatically updates both the PersonField and PersonField_disp fields, so they are kept in sync.
- *personFieldName* .setValue() may be used to set the value of the PersonID. This function can be used in conjunction with *personFieldName_disp* .setValue(), so that the correct name is always displayed for the current ID.

The PersonField.setValue () function expects a valid Person ID, but no validation is done by the client—so assigning an invalid ID does not cause the submit/update action to fail. The display value of the PersonField can be accessed by suffixing ‘_disp’ to the fieldname:

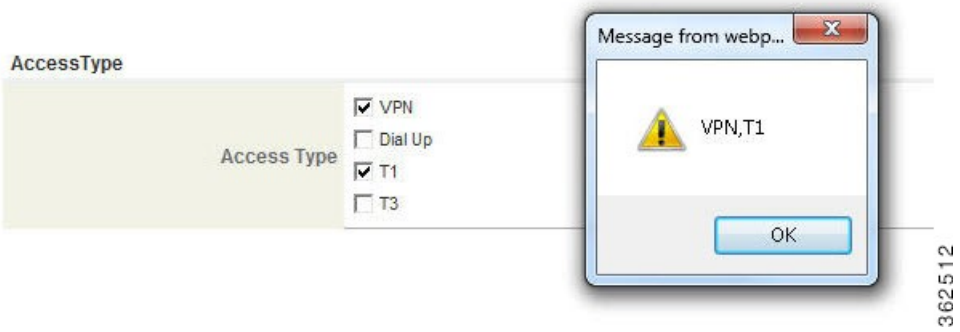
fieldName_disp .setValue(inputValues).

- When the service form is submitted, the Person ID value that was set using this function is retained—the display value is ignored.
- When the Person name changes, the service form does not synchronize the Person display value; that is, it stills show the value that was set using this function.

Fields allowing Multiple Values/Selections

The HTML displays for multi-select fields and for check boxes allows multiple values to be chosen for the same field. The values chosen are represented as a comma-separated list of values, as shown in the following example:

```
alert (serviceForm.EUIT_RemoteAccessDetails.AccessType.getValue([0]));
```



A JavaScript split() method can be used to parse the field value into its distinct elements.

Integrating ISF Code into Service Forms

ISF and service forms implement an event model that is similar, but not identical, to the Document Object Model (DOM) event model. That is, customized JavaScript functions may be invoked to handle events that occur during the processing of a service form.

JavaScript functions are typically invoked as event handlers for processing events that occur as an HTML form is displayed and the user enters data in the form's input fields. Since service forms are generated dynamically, based on the dictionary and form definitions previously stored in the repository, it is not possible for programmers to simply type ISF code into an HTML file. They must rely on the user interface provided by Service Designer to write their functions and to attach these functions to the appropriate event. Therefore, any JavaScript function to be accessed as an event handler must also be defined within the repository. Such functions are defined via the Scripts option of Service Designer, and associated with the appropriate event via the Active Behavior tab for the form.

JavaScript functions written as event handlers can, in turn, call other JavaScript functions. These functions (if they are to have a public scope) cannot be defined as Scripts within Service Designer. Instead, they must be written in a JavaScript library, a text file comprising one or more functions, which resides on a file system accessible to the application server. The Service Designer interface is then used to include these libraries in forms in which their functions are required.

Global Form Settings

Use the global form settings to manage the behavior of JavaScript that is used in all forms. These settings comprise the standard form events:

- When the form is submitted (browser-side)
- When the form is loaded (browser-side)
- When the form is unloaded (browser-side)

Note The server-side triggering events, “After the form is submitted (server-side)” and “Before the form is loaded (server-side)”, are only available for form rules.

Adding JavaScripts

- The General tab allows you to create and maintain a JavaScript function.
- The Libraries tab allows you to include a JavaScript library in the current function and, by extension, in the form to which the function is attached.
- The Active Form Components tab displays the forms to which the current function is attached.

To create a new JavaScript function:

Step 1 Choose **Service Designer** > **Scripts**.

Step 2 Choose **New** > **New Function**. Once the function has been created, choose it for maintenance from the tree structure on the left.

- **Name**: The name of the function. Although this name is just used within Service Designer, to refer to the JavaScript function, it is best practice to use a JavaScript identifier, which will identify the function within the generated code. As a JavaScript identifier, the name is a single, case-sensitive word consisting of letters and numbers and starting with a letter. For guidelines on naming functions, see the Naming Convention in Guidelines for Designing Optimal Service Forms, on page 128.
- **Description**: Optional, but highly recommended description of the function.
- **Add this script to the following events on all forms**: These check boxes provide a way to specify the function as the event handler for the checked event. This “global” attachment would replace the “local” attachment of the function to the event via the Active Form Behavior tab. Global attachment is not recommended for most projects, as discussed in the ISF Coding and Best Practices, on page 123.
- **JavaScript Function**: The actual code for the function, which includes the ISF code. The function signature must not include the “function” keyword (this is automatically added when the function is included in the generated service form), but function contents otherwise follow standard JavaScript rules. For example:

Write this code block in the JavaScript Function	To generate this code in the form:
`CER_Name_onChange() {` `}`	`function CER_Name_onChange() {` `}`

Step 3 Click **Add new JavaScript**.

Adding Arguments to a JavaScript Function

You can add arguments to JavaScript functions (every time the function is called). Function arguments created in Scripts can be overridden by the service designer at the service level.

Step 1 Choose **Service Designer** > **Scripts**.

Step 2 Select a function and choose **General** > **Function Arguments**.

a) In the Argument Name field, enter a name for the argument (follow naming standards for JavaScript identifiers).

b) In the Default Value field, enter a default value.

Step 3 Click **Add** to add the new function argument.

Step 4 (Optional) Enter a description by clicking the Information icon button . A Parameter Description popup window appears where you can enter a description and then click **Set Description**.

Step 5 **In the bottom left of the window,** click **Save**.

Follow the guidelines below in defining arguments:

- Set an unused default value (that is, some default value that will never be overridden) for each of the JavaScript arguments.
  - For example, MyUniqueFunction(arg1, arg2) { .. }: The default value for these two arguments should be - arg1 = 'unused value 1', arg2 = 'unused value 2'.

Tip You must enter a default value for each argument. Otherwise, you will encounter errors.

Tip You must enclose default values intended to be strings in single quotes. Do not use double quotes and do not use two single quotes with no value in between.

- There is no way to mark an argument as a particular data type because JavaScript is type-less.
  - If the intent of an argument is a string value then the default dummy value (the value which will never be possible) can be enclosed in single quotes. For example: 'AAABBBCCCDDD'.
  - If the intent of an argument is a number value then the default dummy value (the value which will never be possible) can be some negative value. For example: –999999999.
- Edit the function arguments after adding the JavaScript function into an Active Form Component.

Associating Libraries with JavaScript Functions

Storing JavaScript in an external JavaScript (library) file has the following benefits:

- Code for many functions is maintained in one place rather than requiring the user to navigate to different screens, as would be the case if each event had a function attached in Scripts.
- Maintaining the code in a file makes it easy to version control the source code.
- Maintaining the code in a text file makes it easy to do global search-replace an execute searches on the file.
- A JavaScript library is simply an ASCII file which resides on the application server. As such, you can use a powerful editor to maintain file contents.
- Since the same piece of code is referenced from one or more functions, changes need to be made only at one location and tested only once.

Prerequisites for Using Libraries

The library approach has the following prerequisites:

- The JavaScript libraries must reside on the web deployment directory (RequestCenter.war) on the application server. By convention the libraries are placed in a directory named “isfcode”.
- The programmer must therefore have access to the file system of the application server. This may entail creating additional logins on that server or providing additional client software (for example, a Remote Desktop service).
- The directory on which the ISF code resides needs to be set with read-write permissions for the ISF programmers.

A decision regarding where to store ISF scripts must be made before detailed design is attempted, as it will affect the design cost. It is significantly more efficient to use the library approach than to embed all ISF in the repository.

If the function calls additional functions which reside in a library, use the Libraries tab of the JavaScripts option to specify the library.

To add a new library:

Step 1 Choose **Service Designer** > **Scripts**.

Step 2 Select a function and choose **Libraries** > **Add Libraries**.

Step 3 In the Add Libraries window, select the check boxes corresponding to the libraries you want to add and click **Add** to include the chosen items in the JavaScript function.
All libraries included in the function will appear on the Libraries page.

You may delete one or more by checking the corresponding check box and clicking **Remove**.

Reviewing Forms With JavaScript

The Active Form Components tab lists the forms to which the current JavaScript has been attached.

To review the forms used for a Javascript

Step 1 Choose **Service Designer** > **Scripts**.

Step 2 Select a function and choose **Active Form Components**.

Step 3 Click on the form name to review the form definition.
This cross-reference reflects only “local” attachment of functions. It does not include any JavaScripts which have been “globally” attached to a service by checking the “Add this script to the following events on all forms” check box on the General tab of the JavaScript function.

Creating JavaScript Libraries

Use this procedure to assosciate library which you have created (or will create) external to Service Catalog.

Procedure

Step 1 Choose **Service Designer** > **Scripts**.

Step 2 Choose **New** > **New Library** to create an entry for your library in the repository.

- **Name**: Any name may be specified for the library, since this is a descriptive field. This name is used within Service Designer to refer to the library.
- **Import from URL**: The location of the library. The library must be located within the web deployment directory (designated as /RequestCenter/). By convention, ISF libraries are placed on the isfcode directory, directly beneath the root directory /RequestCenter.
- **Include this library in all functions**: This, too, is a slight misnomer, as the check box actually includes the library in all service forms, rather than in all “functions”. Checking this check box includes the library (by reference) in the service form, so that any functions defined in the library may be invoked. (A <script> tag for the library is generated into the service form.) Details on this and other options are discussed in the ISF Coding and Best Practices, on page 123.

Step 3 Add the library to the repository by clicking **Add new Library**.

Adding Functions Arguments to JavaScript

To add JavaScript functions to a form:

Step 1 Choose **Service Designer** > **Active Form Component**.

Step 2 Click the **Active Form Behavior** tab.

Step 3 Click the first row in the "Form or Field" column, **This Active Form Component**. The Triggering Event column will list all form-level events.

Step 4 Choose the form-level triggering event to which the JavaScript functions are to be attached.
Available form-level triggering events for JavaScript functions are:

Table 35: Form-Level Triggering Events

Form-Level Triggering Events	Description
When the form is submitted (browser-side)	The code is executed **after** the Submit or Update button is clicked, and **after** any validations specified in the form's Display (such as checking for numeric or mandatory data). These actions are executed **before** the form is submitted to the server. This usually provides for a window to do extra validation, formatting, or any kind of processing before the data is sent to the server. The onSubmit function **must return a Boolean**, true if the submission can proceed, and false to stop it. This corresponds to the HTML event: form ... onSubmit.
When the form is loaded (browser-side)	This is the **preferred** place to add code that populates fields in dictionaries, or that does some kind of form massaging before the form is rendered on the browser. This corresponds to the HTML event: body ... onLoad.
When the form is unloaded (browser-side)	This event is called when the form is being closed. This event can be used to notify the user about data being lost if the window is closed. This corresponds to the HTML event: body ... onUnload.

Note The server-side triggering events, "After the form is submitted (server-side)" and "Before the form is loaded (server-side)", are only available for form rules.

Step 5 Click **Add JavaScript**.
An Add Functions dialog box appears listing JavaScript functions previously defined with the Scripts option.

Step 6 Choose the JavaScript functions you want by checking their check boxes, and then click **Add**. You can use the Search box to search for JavaScript functions, if needed.

The functions appear on the Behavior column below the JavaScripts section.

Note Although you can attach multiple JavaScript functions to the same event in a single form, this practice is best avoided, because the order in which the functions are specified (and executed) cannot be defined. Therefore, if you need functions to execute in a certain order, create one JavaScript function that contains all functions in the desired order.

Adding JavaScript Function to a Field-Level Event

To add a JavaScript function to a field-level event:

Step 1 Edit the form in the Active Form Components component of Service Designer.

Step 2 Click the **Active Form Behavior** tab.

Step 3 In the "Form and Field" column, expand the dictionary node and choose the field to which the function is to be attached.

Step 4 In the Triggering Event column, choose the field-level triggering event that corresponds to the timing at which the function is to be executed.

Figure 6: Triggering Event

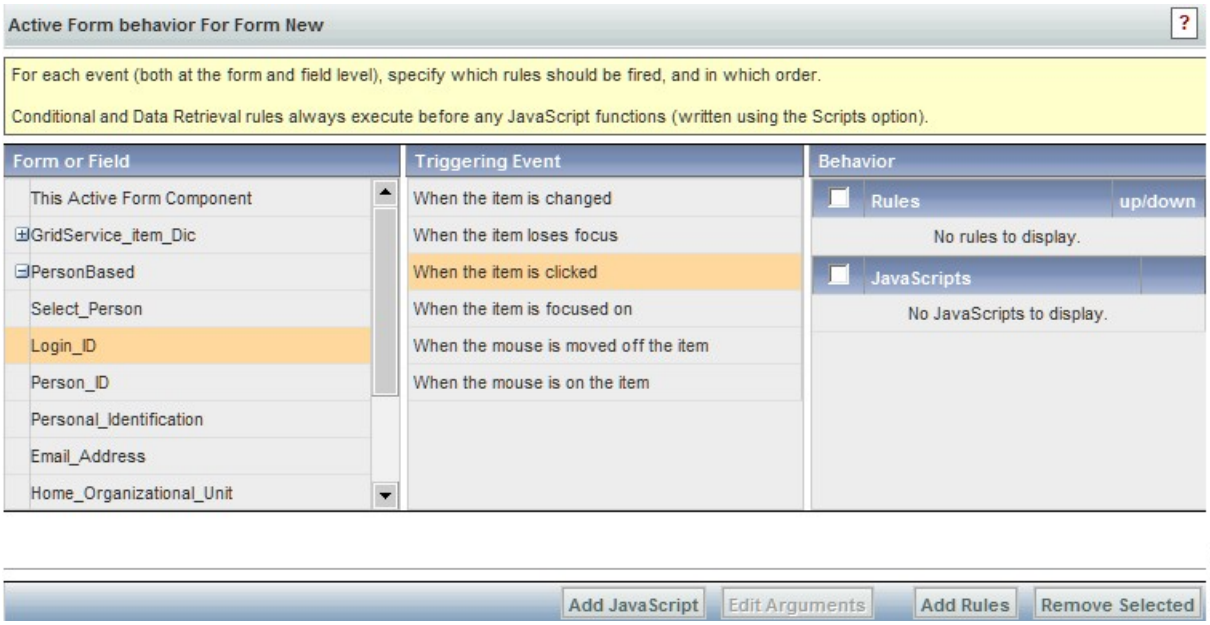


Table 36: Available field-level triggering events are:

Triggering Event Description	HTML Event Name
When the item is clicked	onClick
When the item is changed	onChange

Triggering Event Description	HTML Event Name
When the item is focused on	onFocus
When the item loses focus	onBlur
When the mouse is on the item	onMouseOut
When the mouse is moved off the item	onMouseOver

The **onChange** event is best used to detect changes to Text/Single-select fields. The onChange event for a Person, Date or DateTime field is always triggered, even if the same Person is chosen from the "Select Person" popup window, or the same Date is chosen from the Calendar popup window. Similarly, any typing in a Text field triggers the onChange event, whether the value in the field is actually changed or not.

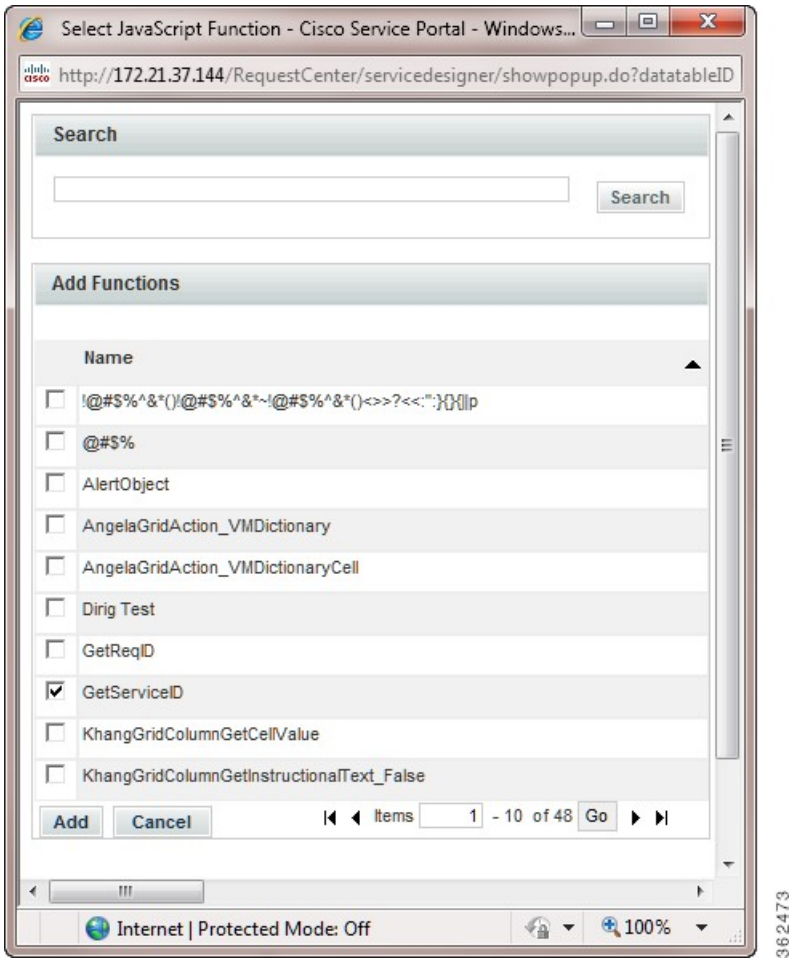
- Fill a field based on the value of the current field (for example, set a flag in a service based on the value of another field entered by the user).
- Choosing "Other" from a drop-down requires displaying an extra text box on the form.

The **onClick** event works best with Radio Button/Check Box controls. The onClick event does not trigger for Person, Date and DateTime fields as text boxes for these field types are always read-only and selection is only possible through the popup window.

Step 5 Click **Add JavaScript**.
An Add Functions dialog box appears listing JavaScript functions previously defined with the Scripts option.

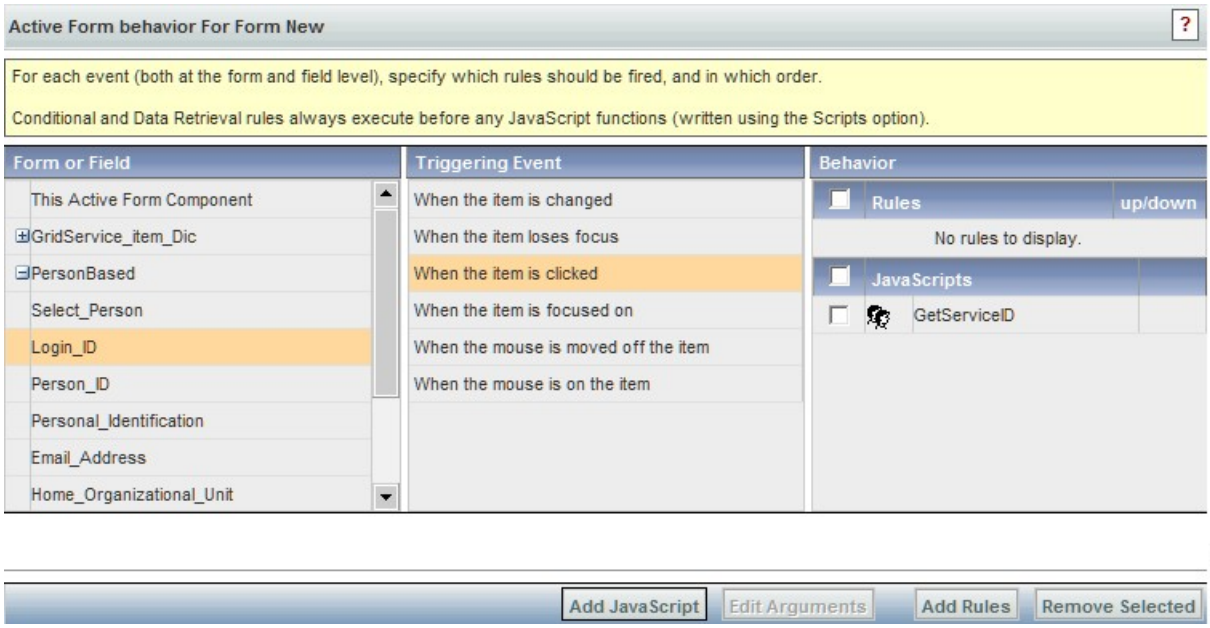
Step 6 Choose the JavaScript function you want by checking its check box, and then click **Add**. You can use the Search box to search for JavaScript functions, if needed.

Figure 7: JavaScript function



The function appears in the Behavior column below the JavaScripts section, as shown below.

Figure 8: Function in JavaScripts Function



Note Although you can attach multiple JavaScript functions to the same event in a single form, this practice is best avoided, because the order in which the functions are specified (and executed) cannot be defined. Therefore, if you need functions to execute in a certain order, create one JavaScript function that contains all functions in the desired order.

Step 7 If required, you can edit arguments for a function by clicking on a JavaScript function in the Behavior column and then clicking **Edit Arguments**. (See the Adding Arguments to a JavaScript Function, on page 114 for more information.)

Using JavaScript

This section describes the usage of Javascript function.

Associated Controls (Buttons and Links)

Any dictionary field may have an associated button, as specified in the “Add a Button” section of the Display Properties for the field:

Figure 9: HTML Representation

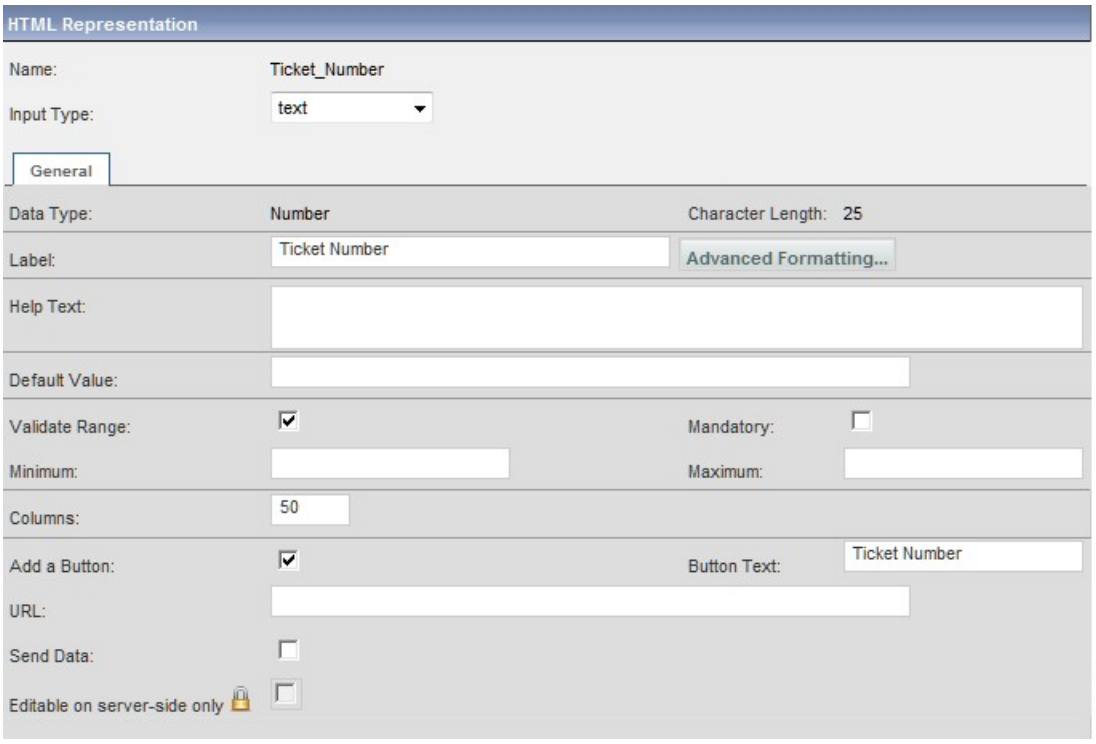


Figure 10: Remedy Ticket Numbers

You can specify a caption (text to appear on the button) and a URL. The URL can point to a JavaScript function or to an external page accessible elsewhere on the network.

When referring to an external web page, use the fully qualified URL.

- The URL may point to a JavaScript function, available in a library, or embed JavaScript code. Form data cannot be included in the parameters, if any, included in a function call. However, the function can include ISF.
- The Send Data option is applicable only to an external web page. This will POST a response to that page including all of the data on the current form.

The disadvantage of using an associated button, rather than placing code in an event handler, is that the user must use an extra keystroke (clicking on the button) to invoke the code. Consequently, buttons are best reserved for events that need to occur on demand, not in the course of regular form processing, or for extended dialog boxes or browsing of external sites.

Simply specifying a URL brings up the specified web page in a separate window. However, the window is not resizable and does not include scroll bars. Therefore, for some applications, it might be preferable to use JavaScript to display the web page in a window that you explicitly specify. Sample code is shown below.

```
javascript:window.open ("http://www.coder.com", "mywindow","status=1,toolbar=1, scrollbars=1,
 width=500, resizable=1");
```

The JavaScript all has to be on one line (no carriage returns). See sample output (primitive, but you get the idea) below. The above line is copied and pasted from the URL entry for the button with the "URL with JavaScript" label.

ISF Coding and Best Practices

Using ISF adds an additional level of complexity to a service catalog project. This section outlines some methodological and technical tips that may be helpful in developing in this environment.

- Install additional tools on your client workstation or reconfigure software previously installed for developing, testing, and debugging JavaScript and ISF efficiently.
- JavaScript does not warn of errors. Therefore, Install a JavaScript DebuggerJavaScript to have more helpful error messages and debugging facilities The Microsoft Script Debugger is available as a free download from Microsoft (find it via a an Internet search). Some web development environments, such as Visual Studio, also include debuggers, which can be used.

In order to use the debugger, you must reconfigure Internet Explorer to enable script debugging in **Tools > Internet Options > Advanced** option.

- Use a text editor that provides syntax highlighting for JavaScript programs. Several such editors are available as freeware or trial versions. Some Java integrated development environments (IDE) also offer support for editing JavaScript files.
- Provide ISF developers read-write access to the directory on the application server (typically, the isfcode directory underneath the RequestCenter.war web archive) on which JavaScript libraries reside. They will also need software for transferring the files between their workstation and the application server.
- Library files are served from the application server. Therefore, page caching must be disabled, to allow revised versions of the libraries to be loaded.

Architecture/Storing ISF Scripts

Service Designer allows JavaScript functions to be stored in Scripts (within the repository) or Libraries (as external files on the file system).

Advantages of Using Libraries

Storing JavaScript in an external JavaScript (library) file has the following benefits:

- Code for many functions is maintained in one place rather than requiring the user to navigate to different screens, as would be the case if each event had a function attached in Script Manager.

- Maintaining the code in a file makes it easier to version control the source code.
- Maintaining the code in a text file makes it easier to do global search-replace and execute searches on the file.
- A JavaScript library is simply an ASCII file which resides on the application server. As such, you can use a powerful editor to maintain file contents.
- Since the same piece of code is referenced from one or more functions, changes only need to be made at one location and tested only once.

Structuring and Using Libraries

In principle, all client code could be included in one library. However, under some circumstances it might be advisable or required to divide the code into multiple libraries.

- One or more JavaScript libraries can be used, grouping the ISF functions so that those that are likely to be used in the same service or group of services are in one library, and those used by another set of services are grouped in another. This allows multiple sets of developers, potentially working on independent projects, to keep their work separate.
- If using multiple, independent libraries, the libraries should never be attached globally to the service forms (using the “Include this library in all functions” check box on the “Library” page of the Scripts option). Instead, the relevant library (libraries) should be included in a function attached to the “onLoad event” of a form (using the “Libraries” tab of the “Scripts” page).
- Similarly, the onLoad event to which the relevant library is attached should not be included in all forms (using the “Add this script...when the form is loaded (browser-side)” option on the Scripts page). Instead, it must be associated with the event using the Active Form Behavior tab for the form in the Active Form Components.
- There is a theoretical advantage in placing ISF functions likely to be used in the same service in one library and those functions that are rarely used, or used under well-defined circumstances, in another. The advantage would be a reduction of memory used (functions not required for a particular service are not loaded). However, since memory is relatively cheap and plentiful these days, and the amount used by ISF functions is minor compared to that used by other components, so no practical advantage have been observed.

Recommended Naming and Coding Standards

If your organization has developed any JavaScript coding standards, you should use a naming and a formatting standard. Although scripts written to support ISF are not generally too long or complex, applying naming and formatting standards helps programmers understand and read one another's code.

JavaScript is case-sensitive so any naming standard needs to specify case to be used for object names.

ISF Function Names

The ISF you write will be in JavaScript functions. If the functions apply to a specific dictionary or field within that dictionary, the function names should reflect this hierarchy. This makes it much easier to track function usage. Function names follow the convention:

- dictionaryName_event

- dictionaryName_fieldName_event

For example,

- RC_REQUESTEDBY_onLoad
- ST_HighProfile_onSubmit
- ST_UserLocation_FieldOffice_onChange
- SVC_Phone_PhoneType_onClick

In addition to dictionary- and field-specific functions, Cisco Advanced Services may create the following site-wide functions, which may be modified as required:

- rc_CommonService_onLoad
- rc_CommonService_onSubmit
- siteRC_REQUESTEDBY_onLoad
- siteRC_REQUESTEDFOR_onLoad

This naming convention makes it easier to understand how each script is used, and to find the appropriate place to create any new functionality. It also allows site-specific code to interface correctly with ISF code that may have been installed in conjunction with a standard service library.

Code Placement

We recommend that ISF JavaScript functions be placed in an external JavaScript library. It is important to order the functions within that file, to facilitate finding a particular function.

Code Formatting

All code should be stored in ANSI text files without tabs, using 2-space indentation for clarity of reading and understanding. Lines should not be longer than 76 characters each.

ISF-Specific Best Practices

Be very careful changing the names of fields in a dictionary. If a field is referred to in a function, that function will stop working.

Every standard property and method in JavaScript starts with a lower case character, and the next word starts with an upper case character. For instance the property "**readOnly**" or the method "**onSubmit**" follow this convention. Although it is not enforced in ISF, following similar conventions is recommended.

Authoring JavaScripts

Service forms are generated automatically based on the specifications you enter interactively in Service Designer. But HTML tags for including libraries in a particular page or pages, or assigning a piece of JavaScript to a particular event embedded in a page are manually. You use Scripts to specify how to include libraries in the generated HTML pages that contain service forms, and the Active Form Behavior tab to instruct the application about which Scripts are event handlers for particular events. You must work outside of Service Catalog to maintain code within the libraries.

Creating a Library

Using the Libraries option under **Service Designer > Scripts**, you can place custom ISF code in one or more JavaScript libraries. Since the library is a text file, external to the application, you need a text editor to maintain the library contents. A JavaScript-aware editor or development environment is highly recommended. Third-party JavaScript libraries may also be used in conjunction with ISF.

Copying the Library to the Application Server

To be accessible by the service form, the library must reside within the directory structure on the application server, mapped to the URL/RequestCenter. By convention, the library is placed on a directory named, isfcode located directly under the root. The physical location of the /RequestCenter site may vary according to how Service Catalog is installed on your server and the application server in use. Please consult your system administrator to determine the physical location on which the ISF libraries need to reside. You also need to ensure that ISF developers have read/write access to this directory and a tool for transferring files to the directory.

Including the Library in Service Forms

A reference to the library (implemented as an HTML script tag) must be included in the generated service form in order for functions in the library to be used in the service form. This reference may be generated in one of two ways.

- Use the Libraries tab of the JavaScripts node of the Scripts option to associate the library with a particular function. The library is then available in all forms that use the function.
- Use the “Include this library in all functions” check box on the Library option of Scripts to include the library in all service forms.

Loading the Library by Including it in a Function

This is the recommended approach to including a library reference in a service form. There should be a function that is invoked as an onLoad event. Libraries are then included in that function. One or more onLoad functions may be coded. Each may have a different set of libraries attached. In this way different teams of developers have control over which libraries are available to their service forms.

Note The loading sequence of the JavaScript libraries is not controlled by the application and can be influenced by known and unknown environmental variables such as Application Server, OS version, and Database Type.

Loading the Library via the Library Check Box

The “Include this library in all function” check box on the Library page is a misnomer; it should be “Include this library in all forms” since the library is only loaded once per form. Using this check box puts the <script> tag for the library at the beginning of the generated HTML page, ensuring that the library and its functions are present when the page-level ISF calls it.

This methodology is not recommended for complex projects.

Note The loading sequence of the JavaScript libraries is not controlled by the application and can be influenced by known and unknown environmental variables such as Application Server, OS version, and Database Type.

Verifying

At this point you have a library (which possibly contains no code) and a specification to use the library for all service forms. You can verify your work by running a service form, looking at the source code, and searching for the name of your library. To view the source code of a service form, you cannot use the "View Source" option of your browser. Because the service form is displayed as a frame within an HTML page, its generated source is not displayed. Instead, move the mouse pointer to within the service form (not within a field) and use the right mouse button option to "View Source". A search shows the name of the library.

Writing Custom JavaScript Functions

Due to the difficulty of debugging in an HTML and JavaScript environment, it is recommended that your write, debug, and test one function at a time while writing the code.You may, of course, edit the library file containing your JavaScript locally; but to test it, it must be copied back to the designated directory on the application server. When initially developing code, or if your access to the application server is limited, it may be easier to write the code within a Script and refactor when you are done, moving the function to the library and leaving only a wrapper function, which calls the library function, under the Scripts. Make sure that you save often and keep backup copies in case you need to revert to a previous version of the code.

Attaching the Function to the Appropriate Events

Once the code is written, it needs to be attached to an event in the form. Since the service form HTML page is generated dynamically, you must use Service Designer to do this. See the Adding Functions Arguments to JavaScript, on page 117 for more information. To attach a field-level event to a dictionary field:

- Write the function in the JavaScript library. Name it *dictionaryName_fieldName_event* , for example, RC_REQUESTORLOCATION_LocationName_OnChange.
- Name the function with a site-specific code prefixed to the name of the function you previously included in the JavaScript library. Service Catalog-provided functions (for example, those used in services included in the Service Catalog libraries) use an "rc" prefix, so the code would appear as shown here:

```
rcRC_REQUESTORLOCATION_LocationName_onChange () {
 RC_REQUESTORLOCATION_LocationName_onChange();
}
```

- Use the Active Form Behavior tab of the Active Form Components option to attach the function to the appropriate field-level event. Choose the field and triggering event, then add the function by clicking Add JavaScript. (If the dictionary is used in multiple forms, the function must be attached to the dictionary field in every form. This is not recommended.)

Testing

ISF code is attached to an active form which can be reused in many services. However, it is not sufficient to test just one service. For example, you may have code that assumes that a field in a dictionary in another form

is also present in the service; if that is not the case, your ISF code will fail with a JavaScript error. Therefore, you need to set up a testing matrix based on the different combinations of forms that can be used. Especially in the initial phases of testing, it is useful to run multiple sessions of Service Catalog simultaneously.

- Start Service Designer in one browser, with the option set to Scripts (to make changes to function code) or Active Form Components: Active Form Behavior (to change JavaScript attachments) displayed.
- Run My Services or Service Manager in the other browser, to test the service form in the moments for which rules or ISF have been defined.
- If you are testing code that resides in a library you will, of course need another window: to edit the library file and upload your changes to the application server.

If a JavaScript error is encountered, the JavaScript debugger is displayed. After you have fixed the error (by editing the function or library code) and dismissed the debugger, there is no reason to exit the current service form and start a new request. Simply refresh the page to cause the current service form to be reloaded with the new ISF code.

If the Browser Cache setting is enabled in the Administration Settings, changes made to the JavaScript libraries will not take effect until the browser cache has been deleted. Therefore in a development environment, it is best to disable browser caching. When modifications to JavaScript libraries are deployed to the production environment where browser caching might be enabled, application users will need to delete their browser cache. To prompt the application users to do so, follow the instructions in the Cisco Prime Service Catalog Administration and Operations Guide to increment the browser cache version.

Guidelines for Designing Optimal Service Forms

Service Catalog enables you to create service forms quickly and easily. Following are some general guidelines and principles for designing optimal service forms.

- **JavaScript code should be specific to a dictionary**. Ensure each function in the code is stand-alone and does not depend on any other function or dictionary. This is to make sure the code functions even if a new dictionary is added or existing dictionaries are removed. For example, assume that two dictionaries used in a particular service have code that must be executed in an onLoad event. Rather than writing one monolithic function, write two dictionary-specific functions, place them in a library, and call them from a wrapper function, which is defined as a Script and attached as the onLoad event to the form:

```
Common_Service_onLoad () { IT_Dictionary1_onLoad(); IT_Dictionary2_onLoad();}
```

- **Create service-independent code**. Testing the code in one service suffices for all services in which that code is used. The code in the previous example is not service-independent. It would fail if the Common_Service_onLoad() function were executed in a service that was missing one or both of the dictionaries. However, this can easily be modified:

```
Common_Service_onLoad () { if (serviceForm.ITDictionary1 !- undefined) {
IT_Dictionary1_onLoad();  }  if (serviceForm.ITDictionary2 !- undefined) {
IT_Dictionary2_onLoad();}
```

Just as the above code tests for the presence of a dictionary in a service before attempting to apply dictionary-specific code, you may need to test for the presence of a particular field before attempting to apply field-specific code. It is best practice to use the dictionary in only one Active Form Component; however, service-specific rules may affect the dictionary's appearance. For example, displaying the supervisor information for the person requesting a service may only be required for those services that require supervisor approval.

Therefore, code that attempts to manipulate the supervisor-related fields should be included in a code block such as:

```
FirstApprover_onLoad () {  if (serviceForm.FirstApprover.SupervisorName !- undefined) {//
   code goes here;  }}
```

A field may be used in a form, but conditionally hidden by a rule or ISF code previously executed. In cases like these, code like the following might be more appropriate, and more robust:

if (serviceForm.FirstApprover.SupervisorName !- undefined) { if (serviceForm.FirstApprover.SupervisorName.isVisible()) {// code goes here; }

- **Send less data to the browser session:** To improve the performance of the form when it is loaded, less data should be sent to the browser session. In previous releases of Service Catalog, any dictionary fields you needed to manipulate had to be sent to the browser in order to be manipulated. Using server-side events, you can manipulate data in dictionaries such that less data is sent to the browser and also it keeps sensitive data under the control of server-side execution and therefore incapable of being intercepted. See Server-Side Data Retrieval Rule, on page 134. For sending less data, you can do the following:
  - Group fields such that fields that do not need to be loaded into the browser session—at least not during the Ordering moment—should be grouped together into one or more "nonloaded" dictionaries. A common example of a "nonloaded" dictionary is one supplying the parameters to an orchestration Agent. The orchestrator often needs more data than the user ever sees on the form. This data is typically derived through conditional rules or data retrieval rules that determine who the user is, what role he has, what OU membership he has, and so on; and all of this is best derived either before the form is loaded in the browser (during the pre-Load event) or once the user has clicked the Submit button (during the post-Submit event).

Seeing the effects of rules operating on "nonloaded" dictionaries will take a bit more testing, since you won't be able to see the values being set until the post-Submit event has occurred. Use of "nonloaded" dictionaries will therefore typically involve checking the form in a subsequent moment (for example, Service Group Authorization or Service Delivery), as a user who has the Manage Service Dictionaries permission and is therefore able to see all the dictionaries defined on the form.

- **Use the pre-Load event rather than the on-Load event**. Generally, conditional rules and data retrieval rules take action on objects that are present in the browser. So a "Hide fields" conditional rule that is tied to the pre-Load event, has in fact nothing to hide, because the fields being hidden do not actually exist yet during the pre-Load event. There is a notable exception to this general rule, as explained in the previous discussion of "nonloaded" dictionaries.

It is possible to manipulate the values of fields that are not loaded into the browser, through either the Set Value conditional rule action, or a data retrieval rule.

The conditional rule and data retrieval rule functionality is highly flexible, enabling you to combine multiple rule actions on multiple targets, and to tie any one rule to multiple events. Any actions that do not have access to their targets—for example, a Set Value action on a field that is not actually present on the service form—are said to "fail silently" or take no effect. If you have a conditional rule that performs a sequence of actions (say, hiding some fields, making others mandatory, and popping up an Alert message; as well as executing a Set Value), you may see most of those actions taking effect in the browser. The effect of a Set Value action, however, may not be visible because either the target field is currently hidden or it is not present in the browser. The conditional rule framework does not stop you from constructing such a rule; it merely ignores any actions that cannot be executed.

The conditional rule and data retrieval rule wizards spell out these behaviors as you construct the rules. Help text embedded into each step explains any limitations you may experience when using certain actions on the server-side events.

- Server-side events provide you with an opportunity to write data directly to the database—specifically, to the WDDX data stored in the database for each requisition entry. The post-Submit event in particular is your window of opportunity for writing the data that was collected or derived from actions in the browser into the database.

An important distinction between the pre-Load and post-Submit events is that pre-Load can write to the database only if the requisition data is already in the database. In other words, if the user has not yet clicked the Order or Add & Review buttons, the data for the requisition does not exist in the database and the pre-Load rules have nothing to manipulate. While the pre-Load event gives you the ability to change the viewable value, it does not persist that value in the database. The post-Submit event provides the persistence.

- Server-Side events provides you with great flexibility for manipulating data that is part of the service form yet protected by the server. What makes this possible is that the "nonloaded" dictionaries are stored in the database and therefore accessible by server-side rules. Although the browser-side rules are generated as JavaScript on the form, the server-side equivalents of those rules are actually generated as Java code—so there are, in effect, two separate manifestations of rules possible. The ISF framework (documented in the Interactive Service Forms (ISF) API Overview, on page 100) enables you to write JavaScript for more complex manipulations of the form data and appearance. This JavaScript can only execute in the browser. Therefore any ISF functions you write cannot be tied to the server-side events.
- This **Editable on server-side only** option, available on the HTML Representation tab for any field defined as Input Type = read-only or hidden, helps you satisfy what are sometimes conflicting requirements: the first is to display helpful data to the requesting user or use "hidden variables" on the form that drive business logic; and the second is to secure data completely from user intervention (including malicious activity).

Two common use-cases for using this flag are:

- ◦ The person-based dictionary, a "template dictionary" you can create in the Dictionaries module
  ◦ The service-item-based dictionary, the structure of which is also automatically created in the Dictionaries module

The two are similar in that they can both exhibit "autopopulation" behavior. In the person-based dictionary, the form user chooses a person (using the Select Person control) and that person's details are automatically displayed on the form. In the service-item-based dictionary, the form user can choose a particular service item he owns or has access to, and the attribute values for that service item instance are automatically displayed.

It is a common service design technique to define most (if not all) of the attributes automatically populated as read-only fields (that is, configured with an Input Type on the HTML representation tab as "read-only"). For example, if your user base is large enough to contain multiple people with the same name, you may use attributes such as the email address to help the form user differentiate among those people and choose the right person. Of course you would not expect the form user to be updating the chosen user's email address. On the other hand, the form user may know this person well and therefore know that he uses his mobile phone exclusively and not the desk phone number obtained through Directory Integration. If this is the case, you may want to make the Work Phone field editable and not just read-only, so that at least the service delivery personnel handling the request have the most reliable way of reaching the chosen person.

Using the "Editable on server-side only" check box ensures that any fields you mark as Input Type = read-only or hidden are entirely controlled by the server and that any manipulation of their values in the browser session

is discarded in favor of the data residing in the Service Catalog database. In fact the only mechanism to override the data residing in the database is a rule (either conditional or data retrieval) that executes during a server-side event, and in this case that rule is actually updating the database value. (In the case of the person-based dictionary, the Person record in DirPerson is not updated, but the WDDX data stored in the database for the requisition entry is.)

In other words, you—the service designer—can manipulate the value of a field marked as "Editable on server-side only," but you can do so only through rules executing during server-side events. This is an important factor to consider, because you may be tempted to think that all you have to do to use these "protected" fields is to check the "Editable on server-side only" flag. That is all that's required if you do not have any reason to manipulate the value returned from the database. But if your business logic requires that you set these values in response to selections the form user makes, you must be sure to tie any conditional rule (for a Set Value action) or data retrieval rule (for a data distribution from a query) to a server-side event.

- **Naming Conventions**: Although it is perfectly legitimate, you should not give a dictionary, form, and field the same name. It results in a very confusing display. Dictionaries should ideally use a prefix notation to denote their usage and perhaps the dictionary group in which they have been placed. For example, a "Reason" dictionary sounds like it is fairly generic, used in many forms. Therefore, it makes sense to place it in a dictionary group named, "Common", and to prefix the dictionary name with a code designating this dictionary group, for example, CMN_Reason. Dictionary groups could reflect the service group in which a service-specific dictionary is used, or perhaps the company department or division for which the dictionary and service have been developed.

An easily understood naming convention for forms is also needed. This is especially critical if you need to differentiate between multiple forms that include the same dictionary or group of dictionaries.

- **Form-Dictionary Relationship**: One form should have only one dictionary. Multiple dictionaries should be included in the same form if:
  - These dictionaries will all be required in the same services.
  - The order in which the dictionaries are presented is the same in all services.
  - No additional dictionaries need to be displayed between the dictionaries in this form. On a service form, all dictionaries in one form component are displayed, in the order specified in that component; then, dictionaries in the next form component included in the service are displayed, in the order specified, and so on.

Examples:

- Assume that a group of services developed for the Facilities department have a chain of approvals consisting of up to three approvers, based on the customer's position or department. In this case the Facilities_Approval form should include three dictionaries: FirstApprover, SecondApprover, and ThirdApprover, as well as rules to determine how many of the three approvals actually need to be collected.
- Assume that most of the services developed for the Facilities department require the standard up-to-three approvers, but a rather expensive service requires an additional approver, at the VP level. You have two options:

One Form	Modify the Facilities_Approval form so that it includes the VP Approver dictionary in addition to the FirstApprover, SecondApprover, and ThirdApprover, and write an additional rule or rules to display and process the VPApprover dictionary only for that one service.
Multiple Forms	Create another form, Facilities_VP_Approval, which includes the VPApprover dictionary and duplicates the HTML, access control, and rules needed to process the FirstApprover, SecondApprover, and ThirdApprover dictionaries.

In most cases, the first solution (one slightly more complex form) should be preferred. You don't have to duplicate work to configure three dictionaries and their associated rules in more than one form. And, if any of those dictionaries or their rules needs to be changed, there is only one place to change them. You should only consider the second solution (two partly redundant forms) when including the additional dictionary would also cause extensive changes to the way the other dictionaries are displayed or processed.

- **Different Renderings for the Same Dictionary**: What if the form's behavior or appearance needs to vary greatly, based on the services in which the form is used? Various scenarios are available. Only you, the service designer, can decide which scenario is preferable, based on the criteria outlined below. For example, virtually every service needs to include that “Reason” dictionary. But sometimes the field label should read “Reason”, sometimes “Justification”, sometimes “Explanation”. Further, the help text associated with the field needs to be substantially different for each service or group of services. In this case, since the dictionary is simple and easily configured, a decision is easy: Create a separate form for each rendering of the dictionary, and include the appropriate form in the corresponding services.

But what if the field or fields that need to be customized on a service-by-service basis were part of a large dictionary with potentially complex rules? You still have the same two options outlined above: one form or multiple forms. However:

One Form	Create one form and customize its appearance using ISF functions.
Multiple Forms	Create a different form for each rendering of the dictionary.

The One-Form option is now complicated by the fact that you cannot customize the dictionary's appearance using conditional rules—conditional rules only modify a field's appearance and behavior, not the contents of the field's label and help text. The Multiple-Forms option has the same drawbacks as before—you need to do the work upfront to create the forms, and the maintenance work to maintain rules in multiple places.

There are two additional options:

- ◦ Get the requirements changed. This is not likely nor a good idea, if it results in the users getting less information about how to supply information in requesting a service.

- Put the problematic field in a separate dictionary that is used in multiple forms, and keep one form for the rest of the dictionary. This is only possible if the field can be displayed before or after all the other fields, not in the middle.

- **Dictionaries, Forms, Rules and Services:** A form typically consists of a set of dictionaries and a set of rules.
  - The rules may affect not only those dictionaries (and their fields) in the same form but dictionaries in other forms.
  - It is very important to test the rules thoroughly in the context of a particular service, to ensure that all referenced dictionaries are in forms included in that service. The Best Practices Reports include a "Dictionaries in Services" report that will facilitate doing this testing and doing an impact analysis to assess potential effects of any proposed changes to dictionaries or rules.

It is also perfectly reasonable to structure Form 2 so that it contains only rules to be applied to a particular service. Many services could include only Form 1, but only the service with extra requirements includes Form 2. This greatly simplified (or potentially eliminates) conditions that would be needed in the rules in Form 2, to ensure they only fire in the appropriate services.

This also eliminates having to write a conditional rule that tests for the service name—a textual element that is subject to change. You would merely have to include the rules-only form in the service affected, sequencing it after the form whose dictionaries are affected. Rules are executed in the order in which the forms are included in the service, and then in the order in which the rules are arranged within the form.

- **Loosely Coupled vs. Tightly Coupled Rules and Dictionaries:** When rules and the dictionaries they affect and refer to are in the same AFC, the rules and dictionaries are said to be "tightly coupled". The example above, with a rule affecting the appearance of a dictionary in another form, shows "loosely coupled" rules and dictionaries. In general, a rule can affect a dictionary in any form, provided both forms are included in the same service. There is, however, one significant limitation on loosely coupling rules and dictionaries—a rule defined in one form cannot be triggered by an field-level event attached to a field in a dictionary in another form. Therefore, loosely coupled rules typically need to be triggered by a form-level event, when the form is loaded or submitted.
- **Rules, ISF and the Requisition Life Cycle**: When a request is submitted, all of the definitions for the dictionaries, HTML representation, access control, and task plan for that service are stored (in a compressed format) with the request data. This ensures that a request can be processed throughout the authorization and delivery moments without any subsequent changes to the form's definition affecting the behavior or appearance of the service form for the in-flight request.

If you change a dictionary, all forms using that dictionary automatically inherit the change and all services which use those forms also inherit the change. Any request for one of the changed services that has been saved but not submitted are marked as obsolete. The user will need to cancel the request or remove the service, read it, and fill in the service form data. Submitted requests are not affected by changes to dictionaries or active form components.

However, rules and ISF functions (whether in a Script or an external library) are always loaded dynamically when a service form is displayed in My Services or Service Manager. Therefore, you must ensure that a current version of the rule/code does not refer to a field no longer on the form or, conversely, that a field on previous versions of the form, but not longer used, can have an associated function. This flexibility is easily accomplished by always checking for the existence of a dictionary or field before attempting to manipulate it via ISF, as discussed previously. If desired, a different version of a library could be attached to the newer versions of the

form so that, for example, the names of functions would not need to change, but their contents could be updated to comply with revised requirements.

- **Changing a Dictionary or Active Form Component**: If you change the definition of a field within a dictionary, all active form components will reflect that change. Similarly, if you change any aspect of the active form component's definition, all service definitions that use that form will reflect that change. You cannot delete/change a dictionary or dictionary field to which a conditional rule is associated, or that triggers a data retrieval rule; you must remove the associations first. This check is not performed for JavaScript functions. Each time the form is changed, the application automatically updates the version number for all service definitions which incorporate that form. You may see a brief delay in saving such changes if the form is used within many services. This may also affect any Requisition API (RAPI) programs implemented for ordering the service, since the RAPI SubmitRequest operations requires the version number of the service to be specified.
- **Coding SQL Entry Data Retrieval Rules:** If you are not familiar with SQL, it may be a bit intimidating to code a SQL Entry data retrieval rule. A "trick" to help familiarize yourself with SQL is to start by configuring a simpler version of the rule as a Database Table Lookup. After you have saved the rule, the rule summary displays the generated SQL. You can copy the SQL statement, paste it into a SQL Entry data retrieval rule, and modify it as required.
- **Using Customer and Initiator Lightweight Namespaces:** The #Customer# and #Initiator# lightweight namespaces are automatically supplied as the default values for fields in the Customer_Information and Initiator_Information dictionaries used in the Customer-Initiator form component. However, these namespaces could be used as default values for any form component.

The design of dictionaries to hold information about customer location could use this technique. Fields about the customer's location could be included in the Customer_Information dictionary. However, such fields may be required on very few services—only those where a service must be delivered to the customer's physical location. A more flexible design may be to configure a separate dictionary (and form component) for the customer location, and include this form component only in services where customer location is relevant.

Multiple namespaces can be specified as a default value. For example, the expression #Customer.FirstName# #Customer.LastName# concatenates the customer's first name and last name, separated by one space. This expression duplicates the appearance of the first field of a person-based dictionary.

Note The use of grid dictionary fields for lightweight namespaces is not currently supported.

Server-Side Data Retrieval Rule

Data retrieval rule's triggering event can be server-side or browser-side. There are two form-level events that execute server-side—pre-Load and post-Submit. You can use these events to execute rules on the server. Such rules are commonly referred to as "server-side rules". As opposed to client-side rules that act upon the service form loaded on the browser, server-side rules act upon any dictionaries in the service form. This includes dictionaries that are never sent to the browser (that is, to which users have neither View nor Edit permissions).

The powerful behavior described above means that if you attach one or more data retrieval rules to the pre-Load event, those rules take effect on all dictionaries, for all system moments. Therefore, if you create a form rule intended to prefill data at the Ordering moment, but be ignored in subsequent server-side events when the dictionaries become read-only, you should not associate it with server-side events. The form is rendered in the browser only after the pre-Load data retrieval completes.

This often provides a better user experience as the time between initial rendering of the service form on the browser and the completion of client-side events can be reduced. However, depending on the size and complexity of the form, the end user's perception of the form load may be better if you minimize the pre-Load data retrieval.

Unlike client-side rules triggered at form load time, form rules that are triggered at pre-Load time are not re-executed when the form is refreshed after hitting validation errors.

Importing Service Items and Standards using Service Link

You can use a Service Link agent to import either service items or standards. This allows you to include the import step as a task in a standard service request. The agent will import a file formatted as described in Import File Format For Service Items and Standards, on page 136, and supports the same import options as are available via the Import from File utility. The message is logged in Service Link as a “SIM Import” message type.

For detailed instructions on using Service Link, see the Cisco Prime Service Catalog Adapter Integration Guide .

To configure a Service Link agent to import service items or standards:

1. Define a new agent in Service Link, specifying a “Service Item” task as the Context Type. At a minimum, you will need separate agents for standards and service items.
2. Specify the “Dummy” adapter as the outbound adapter, and the “File” adapter as the inbound adapter. No transformations are required.
3. Skip the pages on configuring the outbound adapter properties and its parameters.
4. For the inbound adapter properties, specify the names of the directories to be used for processing the import file. The file should be written to the designated “Input” directory.
5. For the inbound parameters, create four parameters and enter the appropriate value in the Dictionary Field as shown in the table below. (Parameter values are case sensitive.)

Table 37: Inbound Parameters and Dictionary Values Table

Parameter Name	Valid Values	Description
Domain	SERVICE ITEM STANDARD	An import file can contain data or definition of service item.
ImportDefinition	TRUE FALSE	True if the definition portion (if any) of the import file should be processed; false otherwise.
ImportData	TRUE FALSE	True if the data portion (if any) of the import file should be processed; false otherwise.
ConflictResolution	INSERT OVERWRITE MERGE	Behavior of the import process in reconciling new data or definitions, in the input file, with data or definitions currently in Service Catalog.

Import File Format For Service Items and Standards

Each import file is an XML file in an industry-standard CIM (Common Information Model) compatible format, version 2.3.1. CIM is based on an object-oriented model and uses technology adapted from Unified Modeling Language (UML). A Document Type Definition (DTD) describing the file format used by Service Catalog can be found at RequestCenter.war\DTD on the application server.

Service Catalog recognizes only a subset of entities that are available under CIM. Those entities that it does not recognize it ignores. In Service Catalog's CIM implementation:

- Each service item or standard is a **class**.
- The details about the service item—its description, display name, and assigned group (classification)—are **qualifiers** for the service item.
- The attributes specified for the service item are **properties** of the service item class.
- Each attribute (property), in turn, has **qualifiers**. These qualifiers constitute the detailed configuration for each attribute—its description, caption, whether it is visible in My Services, and the sequence in which the attribute occurs in the service item definition.
- Each service item class may have many **instances**. Each of these corresponds to one service item instance.
- Each instance of a service item has one **property** for each attribute of the service item. You may also include, optionally, three attributes that comprise the subscription information for the service item during import—Requisition Entry ID, Customer Login Name, and Organizational Unit Name. The value in the XML file updates the value of the corresponding attribute or subscription field of the service item instance.

The previous discussion applies to both service items and standards in general. Note that standards are not "owned" by any individual user or organizational unit per se and therefore you cannot set subscription attributes for them.

Syntax for a Service Item Import File

The Table 38: Syntax for Specifying a Service Item or Standards Import File table summarizes the syntax for specifying a service item or standards import file. The same syntax applies to both service items and standards. Each XML file consists of one SI Import Specification.

The Import Specification (SIImportSpec), in turn, consists of a SIClassDefinition, followed by a list of one more SIInstances for the specified class (Service Item or standard). One or both of the class definition and instances may be included in the file. File content should match the import options specified.

Each SIClass specification consists of the service item/standard definition, embedded within appropriate XML tags, followed by the definition of each of the attributes.

Each attribute definition specifies the name, caption, and other configuration options specified for the attribute.

Each service item instance (SIInstance) specifies the name of the service item/standard to which the instance applies and a list of the values for each of the attributes of the item. If an attribute value is blank, appropriate tags must still be included in the import file, with no value specified for the attribute.

Table 38: Syntax for Specifying a Service Item or Standards Import File

SIImportSpec =	S<?xml version="1.0" encoding="utf-8"?> <CIM CIMVERSION="" DTDVERSION=""> <DECLARATION> <DECLGROUP> SIClassDefinition SIInstanceList </DECLGROUP> </DECLARATION> </CIM>
SIClassDefinition =	<<VALUE.OBJECT> SIClass </VALUE.OBJECT>
SIClass =	<CLASS NAME="Service Item Name"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE></VALUE> </QUALIFIER> <QUALIFIER NAME="Classification" TYPE="string"> <VALUE>Service Item Group</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayName" TYPE="string"> <VALUE>SI Display Name</VALUE> </QUALIFIER> SIAttributeList </CLASS>
SIAttributeList =	SIAttributeDefinition [SIAttributeDefinition]

SIAttributeDefinition =	<PROPERTY NAME="Name" TYPE="string"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>Description of the attribute</VALUE> </QUALIFIER> <QUALIFIER NAME="Caption" TYPE="string"> <VALUE>Caption for the attribute</VALUE> </QUALIFIER> <QUALIFIER NAME="IsVisible" TYPE="sint32"> <VALUE>1 if visible, 0 if not visible</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayOrder" TYPE="sint32"> <VALUE>Sequence of the attribute</VALUE> </QUALIFIER> </PROPERTY>
SIInstanceList =	SIInstance [SIInstance]
SIInstance =	<VALUE.OBJECT> <INSTANCE CLASSNAME="ClassName"> SIPropertyList </INSTANCE> </VALUE.OBJECT>
SIInstanceDataList=	SIInstanceData [SIInstanceData]

SIInstanceData=	<PROPERTY NAME="AttributeName" TYPE=SIDataType> <VALUE>Attribute value</VALUE> </PROPERTY> Optional subscription properties: <PROPERTY NAME="#RequisitionEntryID#" TYPE="sint32"> <VALUE>Requisition entry ID</VALUE> </PROPERTY> <PROPERTY NAME="#LoginName#" TYPE="string"> <VALUE>Customer login name</VALUE> </PROPERTY> <PROPERTY NAME="#OrganizationalUnitName#" TYPE="string"> <VALUE>Organizational unit name</VALUE> </PROPERTY> <PROPERTY NAME="#AccountName#" TYPE="string"> <VALUE>account name</VALUE> </PROPERTY> <PROPERTY NAME="#Operation#" TYPE="string"> <VALUE></VALUE> </PROPERTY>
SIDataType =	{“char16” | “string” | “datetime” | “real64” | “sint32” | “sint64”}

The datatypes do not exactly match the datatypes specified when you define a Service Item or Standard using Service Item Manager. Map the Service Item Manager datatypes to those used in the import file using the Table 39: Import File Datatype for Service Item Manager Datatype table.

Table 39: Import File Datatype for Service Item Manager Datatype

Service Item Manager Datatype	Import File Datatype
STRING(32)	char16
STRING(128)	<not supported>
STRING(512)	string

Service Item Manager Datatype	Import File Datatype
INTEGER	sint32
LONGINTEGER	sint64
DOUBLEFLOAT	real64
MONEY	Use real64
DATETIME	Datetime (must use the following format for date value = yyyy-mm-dd hh:mm:ss, for example 2009-04-15 12:00:00)

Service Item Subscription Processing Rules for Imported File

Unlike create and update operations performed by internal and external service item tasks, the create and update operations handled through the file import mechanism happen outside the context of a service request delivery plan. Hence the import program does not validate the relationship among the Requisition Entry ID, Customer Login Name, and Organizational Unit Name provided in the import file. As long as they are valid IDs/names, the input is accepted.

Here are the high-level processing rules for subscription:

- If no subscription information is provided when a service item instance is created, the item stays unassigned.
- If only Customer Login Name property is specified at the time a service item instance is created, the Organizational Unit owner of the item is set to the home organizational unit of the customer and the account ownership is set to the tenant account derived from the home organizational unit if there is one.
- If Customer Login Name or Organizational Unit name, or Account name properties are specified, the values are used to update the service item subscription. When the value is blank, the corresponding subscription field is set to null. The absence of a property in the import file means no change/override on the property value.
- When a Requisition Entry ID is provided in the properties, the corresponding requisition ID is associated with the service item and displayed in My Services and Service Item Manager.
- Once a Requisition Entry ID is set for a service item, it cannot be modified or reset to null.

The possible combinations for customer and organizational unit assignment and the outcome are summarized in Table 40: Service Item Subscription Table.

Account assignment works in a similar way as organizational unit assignment and the permutations are not enumerated in the table below.

:

Table 40: Service Item Subscription Table

When service item is created			
Property In XML	Resulting Subscription		
Login Name	OU Name	Customer	OU
None	None	NULL	NULL
None	Blank or Invalid Value	NULL	NULL
None	Valid OU Name	NULL	OU provided
Blank or Invalid Value	None	NULL	NULL
Blank or Invalid Value	Blank or Invalid Value	NULL	NULL
Blank or Invalid Value	Valid OU Name	NULL	OU provided
Valid Customer	None	Customer provided	Home OU of Customer
Valid Customer	No Value	Customer provided	NULL
Valid Customer	Valid OU Name	Customer provided	OU provided

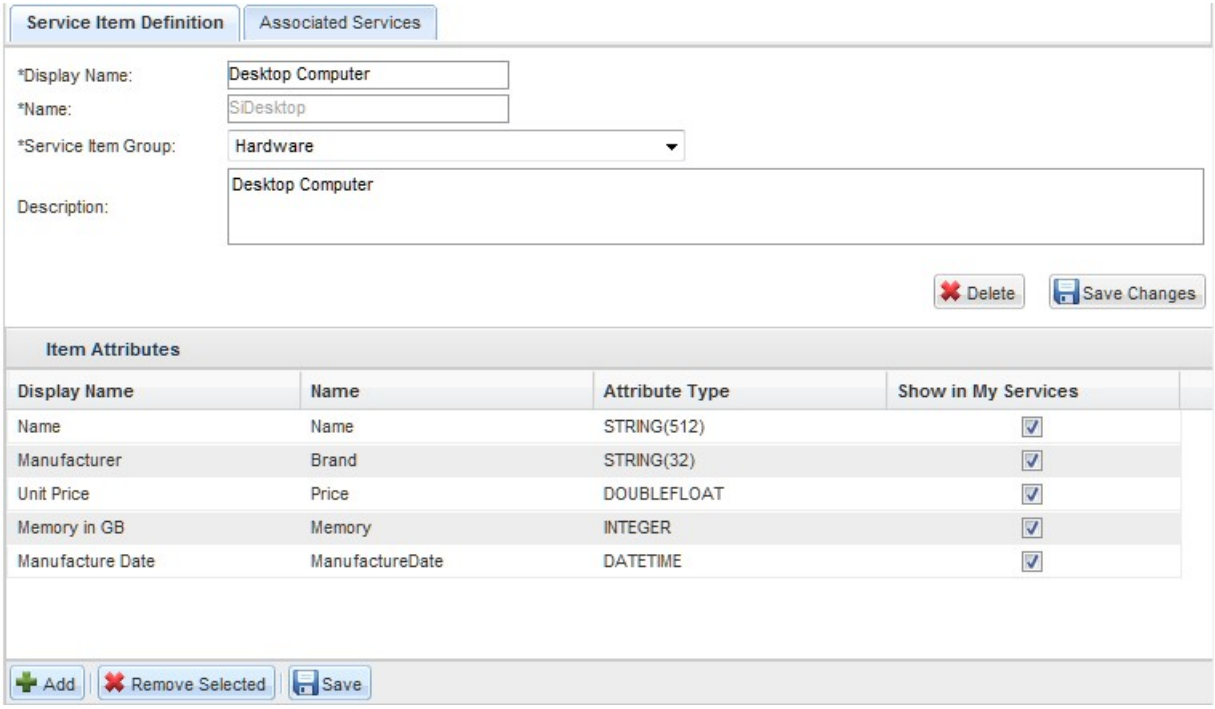
When service item is updated			
Property In XML	Resulting Subscription		
Login Name	OU Name	Customer	OU
None	None	No change	No change
None	Blank or Invalid Value	No change	NULL
None	Valid OU Name	No change	OU provided
Blank or Invalid Value	None	NULL	No change
Blank or Invalid Value	Blank or Invalid Value	NULL	NULL
Blank or Invalid Value	Valid OU Name	NULL	OU provided
Valid Customer	None	Customer provided	No change
Valid Customer	Blank or Invalid Value	Customer provided	NULL

When service item is updated			
Valid Customer	Valid OU Name	Customer provided	OU provided

Example: Service Item Import File

Assume that a “Desktop” service item has been created, with the following definition:

Figure 11: Desktop Service Item



The import file for this service item might look like the sample below:

Table 41: Import File for Service Item Example

Sample XML	Description/Usage
<?xml version=“1.0” encoding=“utf-8”?> <CIM CIMVERSION=“” DTDVERSION=“”> <DECLARATION> <DECLGROUP>	Beginning of SI Import Specification.

Sample XML	Description/Usage
< VALUE.OBJECT> <CLASS NAME="Desktop"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>Desktop Computer.</VALUE> </QUALIFIER> <QUALIFIER NAME="Classification" TYPE="string"> <VALUE>Hardware</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayName" TYPE="string"> <VALUE>Desktop Computer</VALUE> </QUALIFIER>	Properties of an SI—its name, description, display name, and group (classification).
<PROPERTY NAME="Name" TYPE="string"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>The name of the computer.</VALUE> </QUALIFIER> <QUALIFIER NAME="Caption" TYPE="string"> <VALUE>Name</VALUE> </QUALIFIER> <QUALIFIER NAME="IsVisible" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayOrder" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> </PROPERTY>	Definition of the Name attribute of the SI. The Name attribute is required in all SIs; its caption must also be "Name".

Sample XML	Description/Usage
<PROPERTY NAME="Brand" TYPE="char16"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>Brand name of the computer.</VALUE> </QUALIFIER> <QUALIFIER NAME="Caption" TYPE="string"> <VALUE>Manufacturer</VALUE> </QUALIFIER> <QUALIFIER NAME="IsVisible" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayOrder" TYPE="sint32"> <VALUE>2</VALUE> </QUALIFIER> </PROPERTY>	Definition of the Brand attribute of the SI.
<PROPERTY NAME="Price" TYPE="real64"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>MSRP</VALUE> </QUALIFIER> <QUALIFIER NAME="Caption" TYPE="string"> <VALUE>Unit Price</VALUE> </QUALIFIER> <QUALIFIER NAME="IsVisible" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayOrder" TYPE="sint32"> <VALUE>3</VALUE> </QUALIFIER> </PROPERTY>	Definition of the Price attribute of the SI.

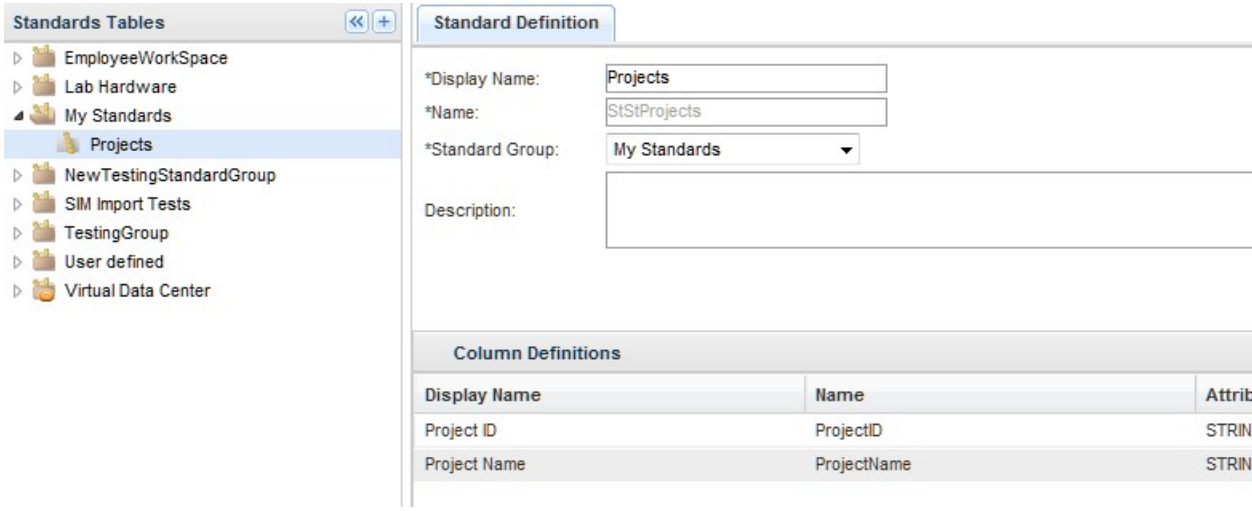
Sample XML	Description/Usage
<PROPERTY NAME=“Memory” TYPE=“sint32”> <QUALIFIER NAME=“Description” TYPE=“string”> <VALUE>Amount of RAM in GB</VALUE> </QUALIFIER> <QUALIFIER NAME=“Caption” TYPE=“string”> <VALUE>Memory in GB</VALUE> </QUALIFIER> <QUALIFIER NAME=“IsVisible” TYPE=“sint32”> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME=“DisplayOrder” TYPE=“sint32”> <VALUE>4</VALUE> </QUALIFIER> </PROPERTY>	Definition of the Memory attribute of the SI.
< PROPERTY NAME=“ManufactureDate” TYPE=“datetime”> <QUALIFIER NAME=“Description” TYPE=“string”> <VALUE>Date of manufacture</VALUE> </QUALIFIER> <QUALIFIER NAME=“Caption” TYPE=“string”> <VALUE>Manufacture Date</VALUE> </QUALIFIER> <QUALIFIER NAME=“IsVisible” TYPE=“sint32”> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME=“DisplayOrder” TYPE=“sint32”> <VALUE>5</VALUE> </QUALIFIER> </PROPERTY> </CLASS> </VALUE.OBJECT>	Definition of the ManufactureDate attribute of the SI.

Sample XML	Description/Usage
<VALUE.OBJECT> <INSTANCE CLASSNAME=“Desktop”> <PROPERTY NAME=“Name” TYPE=“string”> <VALUE>Thinkpad T60</VALUE> </PROPERTY> <PROPERTY NAME=“Brand” TYPE=“char16”> <VALUE>LENOVO</VALUE> </PROPERTY> <PROPERTY NAME=“Price” TYPE=“real64”> <VALUE>899.99</VALUE> </PROPERTY> <PROPERTY NAME=“Memory” TYPE=“sint32”> <VALUE>3</VALUE> </PROPERTY> <PROPERTY NAME=“ManufactureDate” TYPE=“datetime”> <VALUE>2009-04-15 12:00:00</VALUE> </PROPERTY> <PROPERTY NAME="#RequisitionEntryID#" TYPE="sint32"> <VALUE>10</VALUE> </PROPERTY> <PROPERTY NAME="#LoginName#" TYPE="string"> <VALUE>jsmith</VALUE> </PROPERTY> <PROPERTY NAME="#OrganizationalUnitName#" TYPE="string"> <VALUE>Finance</VALUE> </PROPERTY> </INSTANCE> </VALUE.OBJECT>	One instance of a Desktop SI, for a “Thinkpad T60”.
</DECLGROUP> </DECLARATION> </CIM>	End of the SI Import Specification.

Example: Standard Import File Example

Assume that a “Projects” standard has been created, with the following definition:

Figure 12: Standard Import File Example



The Service Link Service Item Task would be defined with the following agent parameters:General

- Outbound Properties
- Inbound Properties
- Outbound Request Parameter
- Inbound Response Parameter
- Internal Parameters

The import file for this standard might look like the sample below:

Table 42: Standard Import File Example

Sample XML	Description/Usage
<?xml version=“1.0” encoding=“utf-8”?> <CIM CIMVERSION=“” DTDVERSION=“”> <DECLARATION> <DECLGROUP>	Beginning of Standard Import Specification.

Sample XML	Description/Usage
< VALUE.OBJECT> <CLASS NAME="Projects"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>Describe it here</VALUE> </QUALIFIER> <QUALIFIER NAME="Classification" TYPE="string"> <VALUE>My Standards</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayName" TYPE="string"> <VALUE>Projects</VALUE> </QUALIFIER>	Properties of the Standard—its name, description, display name, and group (classification).
<PROPERTY NAME="ProjectID" TYPE="string"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>ID of the Project</VALUE> </QUALIFIER> <QUALIFIER NAME="Caption" TYPE="string"> <VALUE>Project ID</VALUE> </QUALIFIER> <QUALIFIER NAME="IsVisible" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayOrder" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> </PROPERTY>	Definition of the ProjectID attribute of the Standard.

Sample XML	Description/Usage
<PROPERTY NAME="ProjectName" TYPE="string"> <QUALIFIER NAME="Description" TYPE="string"> <VALUE>Name of the project</VALUE> </QUALIFIER> <QUALIFIER NAME="Caption" TYPE="string"> <VALUE>Project Name</VALUE> </QUALIFIER> <QUALIFIER NAME="IsVisible" TYPE="sint32"> <VALUE>1</VALUE> </QUALIFIER> <QUALIFIER NAME="DisplayOrder" TYPE="sint32"> <VALUE>2</VALUE> </QUALIFIER> </PROPERTY>	Definition of the ProjectName attribute of the Standard.
<VALUE.OBJECT> <INSTANCE CLASSNAME="Projects"> <PROPERTY NAME="ProjectID" TYPE="string"> <VALUE>001</VALUE> </PROPERTY> <PROPERTY NAME="ProjectName" TYPE="string"> <VALUE>Reporting System Upgrade</VALUE> </PROPERTY> </INSTANCE> </VALUE.OBJECT>	First instance of a Project Standard.

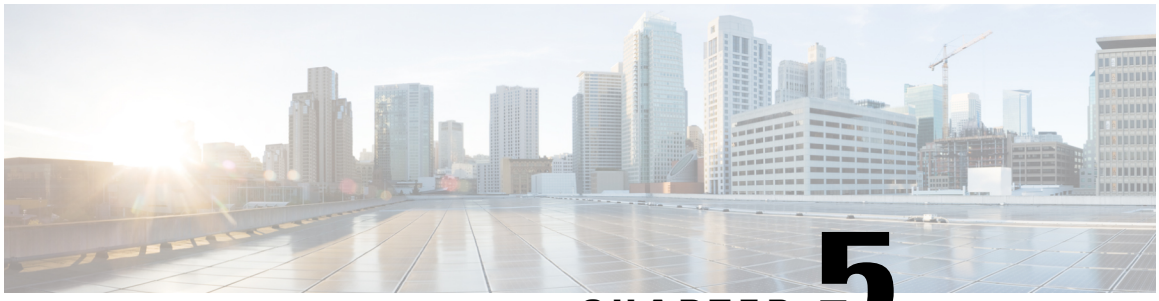
Sample XML	Description/Usage
< VALUE.OBJECT> <INSTANCE CLASSNAME="Projects"> <PROPERTY NAME="ProjectID" TYPE="string"> <VALUE>002</VALUE> </PROPERTY> <PROPERTY NAME="ProjectName" TYPE="string"> <VALUE>Decision Support System Installation</VALUE> </PROPERTY> </INSTANCE> </VALUE.OBJECT>	Second instance of a Project Standard.
</DECLGROUP> </DECLARATION> </CIM>	End of the Standards Import Specification.

Service Item Import DTD

The Document Type Definition (DTD) that describes the format required for a Service Item or Standards import file is available on the Service Catalog application server, under RequestCenter.war\DTD. The DTD is given below.

```
<!ENTITY % CIMName "NAME           CDATA         #REQUIRED">
<!ENTITY % CIMType "TYPE (char16|string|sint32|sint64|datetime|real64)">
<!ENTITY % ClassName "CLASSNAME      CDATA         #REQUIRED">
<!ELEMENTCIM (DECLARATION)>
<!ATTLISTCIM
 CIMVERSION CDATA #REQUIRED
 DTDVERSION CDATA #REQUIRED
>
<!ELEMENTDECLARATION (DECLGROUP)+>
<!ELEMENTDECLGROUP (VALUE.OBJECT*)>
<!ELEMENTVALUE.OBJECT (CLASS | INSTANCE)>
<!ELEMENTCLASS (QUALIFIER*, PROPERTY*, METHOD*)>
<!ATTLISTCLASS
 %CIMName;
>
<!ELEMENTQUALIFIER (VALUE?)>
<!ATTLISTQUALIFIER %CIMName;
         %CIMType;              #REQUIRED
>
<!ELEMENTPROPERTY (QUALIFIER*, VALUE?)>
<!ATTLISTPROPERTY %CIMName;
         %CIMType;              #REQUIRED
>
<!ELEMENTVALUE (#PCDATA)>
<!ELEMENTINSTANCE (QUALIFIER*, PROPERTY*)>
<!ATTLISTINSTANCE
 %ClassName;
```

```
>
<!ELEMENTMETHOD (QUALIFIER*)>
<!ATTLISTMETHOD %CIMName;
>
```



CHAPTER 5

Configuring Services and Service Bundles

This chapter contains the following topics:

Configuring Services and Service Bundles

Services can be designed and grouped to organize a set of similar or related services “owned” by a particular service team in the Service Designer module. Using service groups, you can:

- Configure authorization and escalation processes for services in the group
- Configure permission to order services in the group
- Assign functional positions that are used by the group

Service group folders are different from the categories that end users browse through in My Services. The names of the service group are not displayed to an end user.

For example, to create a “Telephone Services” service group containing the services for acquiring and setting up telephone service for your organization; you need to have a service team assigned to deliver the services and have other people responsible for the management of the Telephone Services group itself. The service group specifies, who can order cellular phones and has an authorization process in place once a phone has been ordered. The Telephone Services group may also have a series of escalation messages configured to alert people when they are late authorizing or reviewing a service.

Creating a Service Group

Similar services can be grouped using Service Designer module. Using groups you can configure authorization processes, ordering permissions, and escalation notifications at the service group level. For information on configuring permissions at a service level, see Defining Service Level Permissions to Order a Service, on page 174.

Before You Begin

Choose an Organizational Unit (OU) or people who will review a service request, approve service request and also handle escalations.

Step 1 Choose **Service Designer**.

Step 2 Click **New** > **New Service Group**.

Step 3 Enter field information as provided in the Table 43: Service Group Field Details and click **Add This Service Group**. To view a service group, choose a service group name to open the details and configuration tabs for that service group, or Expand the + icon next to the service group name to see the services associated with the group.

Step 4 After a service group is created, configure the service group such that a person or a group of people are able to manage the services in the service group. To configure, select the service group and do the following:

- Select a service team and assign functional positions in **General** tab. Refer to the Table 44: Service Group Configuration Table to enter field details.
- Configure authorization structure, review and escalation processes in **Authorizations** tab. Refer to the Table 45: Workflows for Authorization and Reviews to enter fields.
- Assign role based permissions in the **Permissions** tab. The group object-level permissions are types of actions that people, organizational units, groups, functional positions, and roles can perform. Being able to design services in a service group automatically allows the user to view services in that group. See Table 46: Assigning Service Group Permissions to configure permissions at a group level

However, the ability to order services in the group must be independently assigned. For more details on difference between group level versus service level permissions, see Defining Service Level Permissions to Order a Service, on page 174.

You can delete a service group if you no longer need it. You cannot delete a service group in which services still exist. If services do exist, delete or transfer them to other service groups before deleting the group. To delete a Service Group, choose the service group you want to delete and click **Delete** in the General tab.

Table 43: Service Group Field Details

Field	Description
Name	The **name** for the service group. Required. The name should be specific to your organization and needs. End users do not see this name, and the name is editable after service group creation. Sample names include “End User IT Desktop Support,” “End User IT Desktop Software,” or “Identity Management”.
Description	A brief **description** for the service group; optional but recommended.

Service Team	The **Service Team** that is responsible for services in this service group, that is, the team that "owns" the services in the service group.

Table 44: Service Group Configuration Table

Field	Description
Name	Enter the name of Service Group
Description	Enter the description of the Service Group
Service Team	Associate an appropriate service team for the service group. For more information see, Setting Up Services, on page 6. If you change the service team assigned to a service group: • You may need to reconfigure the "Access Control" specified for active form components used in services in this group, to explicitly add the queue associated with the previously specified service team to the list of additional participants. This is required if any tasks have been assigned to that queue. For more information, see Adding Access Control Settings to a Specific Service, on page 58. • All person assignments to functional positions are removed. This is because only members of the service team that owns the service group can fill functional positions in that group. Any such assignments need to be respecified.

Field	Description
Functional Position	**To add a functional position:** 1 Click **Add** below the functional position table and select a functional position. 2 Select the check box next to the n positions you want to add. **To assign functional Position to People within the Service Group:** Click the "**...**" button in the row of the functional position and search for the person using the Search Option. Select the person and click **OK**. **Note** If there are tasks assigned to such functional positions, the tasks will go into the Default Delivery Queue. For more information, see Structuring Your Organization. For description about functional position, see Cisco Prime Service Catalog 11.0 Administration and Operation Guide.
Save	Click **Save** to save the changes on this tab.

Table 45: Workflows for Authorization and Reviews

Workflow	Procedure
Configuring Authorization Structure	Choose one of the following fields under **Authorization Structure For Service Group:** **Use Site Authorization Structure Only:** Uses only site-wide authorization structure and ignores the authorizations and reviews configured at service group or service level. **Use Service Group-level Authorization Structure Only:** (will not use service group-level): Allows you to develop an authorization structure based on a unique scheme of roles, order, or escalations. **Use Both Site Level and Service Group Level Authorization Structures:** Accepts the site-wide scheme of roles, order, or escalations, and allows you to supplement it with a customized scheme that you establish to enhance the authorization process. For more information, see Configuring Site Administration in Cisco Prime Service Catalog 11.0 Administration and Operation Guide.

Workflow	Procedure
Configuring Authorization Types	Choose one of the following **Authorization Type:** • **Service Group Review:** are for information only and a reviewer cannot reject a service or cancel the delivery process. The delivery process does not proceed until the reviewer clicks OK in the review task. • **Service Group Authorization:** The approver can determine if the person requesting the service is eligible to receive it. If an authorization is rejected, the process stops and the service is not delivered authorizations are performed sequentially. It is possible to have several service group authorizations or reviews, each performed by a different person. The authorizations are performed in sequence, in the order specified; if an earlier authorizer rejects the service request, no further work on the service request is done. If you see the message "not currently enabled via the Administration module," then the particular service group authorization/review is not enabled for your site. This may reflect decisions made by the site administrator regarding site-wide standards. This setting can be changed in the Administration module
Defining the Review and Escalation Process	**To define Reviews Concurrent Process:** Click **Add** and enter the **Name**, **Subject**, **Duration** of review, **Effort** for each review, and person to **Assign** a review. Reviews are performed concurrently. Since reviews cannot cancel the delivery of a service, they are performed concurrently, to expedite the delivery process. To Define **Escalation Concurrent Process**, click **Add** and update the number of Hours after which you can send escalation email, and the recipients of each escalation email. For more information, see Configuring Delivery Tasks .

Table 46: Assigning Service Group Permissions

Permission to:	Definition
Design services and change data in this service group	User can design services in this service group and change group data. These rights are typically reserved for the individuals who design and configure services in this service group.
View services and other information in this service group	User can view the service group and service definitions, but is unable to change them.
Order services in this service group	User can order services in this service group. This is how you allow service consumers to see and request these services.
Assign rights to people	User can assign these permissions to other people. This right is typically restricted to the service designer and owner of the service group.

Creating a Service

Use this procedure to add a new service to the catalog and set the context for the service by including descriptive information, as well as to configure the important attributes and settings that affect reporting and display behavior of a service.

Before You Begin

Set up a service group. See Creating a Service Group, on page 153

Step 1 Choose **Service Designer** > **New** > **New Service**.

Step 2 Enter the details in the fields provided.

Note Use **Enter a URL** field to further describe the service by linking to supporting information. The service description in My Services will display a “More information” link to this URL.

Step 3 Click **Add This Service**.

Step 4 After adding the service, you can begin to configure it by entering information on the **General** tab.
The **General** tab is primarily used to influence the consumer experience as the end user browses the service catalog. Use the Table 47: Service Attributes as a reference for completing the fields.

Step 5 Click **Save**.

Table 47: Service Attributes

Field	Definition
Name	The name of the service. This is the name the end user will see in the service catalog. The service name should not exceed 200 characters. For example: Computer Memory Upgrade
Status	A status of **Active** means the service is searchable and available for ordering in My Services. A status of **Inactive** prevents the service from being searchable or orderable within the service catalog. A service may be inactive if it is still in draft form, has expired, or if it is a service you plan to offer at intervals, but not always.
Service Group	The service group to which the service belongs.
Orderable Service	A setting that enables (Yes) or disables (No) the "Order" and "Order For Others" link for the service in My Services. Non orderable services are typically created to provide customers with nonactionable information in the service catalog.
Reportable	This field does not affect the customer. It is for a service designer to specify whether the service data should be loaded into the service Catalog data mart so that it can be used in ad-hoc reports or queries constructed using Ad-Hoc Reports and Report Designer in the Advanced Reporting module. See the Cisco Prime Service Catalog Reporting Guide for additional information and best practices for making services reportable.
Entitlement	Setting this option to "Yes" allows any end user to request this service without requiring an authorization. If a site level, department level, or service group level authorization is in place, setting Entitlement to "Yes" ignores those authorizations for this service.
Service ID	Displays the internal ID of the service.
Compute Price	This field enables you to calculate the price of a service before the consumer orders the service. If you select "**Yes**" in the drop down list, you see a **Compute Price** button when you choose to Order a Service in **My Services**.

Field	Definition
Ordering Mode	This field provides you an option to either add the service to a shopping cart or order the service instantly. The options available are as follows: • **Add Review Enabled:** Provides an **Add to Cart** button (Service Catalog) or **Add and Review Order** button (My Services) for end users to add the service to a shopping cart. Users can review the service and submit the order at a later time. More services of the same ordering mode can be placed in the same shopping cart. This is the default setting. • **Add Review Disabled:** The service can only be ordered in its own cart as a separate request. Only the **Submit Order** button is available. • **1-Click:** Similar to Add and Review Disabled, the service is ordered in its own cart. In addition, user is presented with a simple confirmation dialog to confirm the request submission. This allows the user to see the request as an action. This mode is particularly suitable for requests that do not require any input data from the user, for example, “Stop virtual machine”. **Note** 1-Click feature is available in Service Catalog module only.
Pagination View Mode	This field allows you to present the dictionaries in the ordering page in the form of a wizard in the Service Catalog module, showing form fields in multiple pages with previous and next navigation controls. For more details on how to configure the ordering page for a Service Item see, Displaying Service Form as a Wizard, on page 96.
Is Template	Check the **Is Template** check box, if you want to use the service as a template Integrations.
Template Type	This field appears only when the *Is Template* option is checked. The Template Type field allows you to select the template type when you create custom templates for UCSD or Cloud Center services. The following templates types are available in the drop-down list: • Standard Catalog • Advanced Catalog • Container Catalog • UCSD template • Application
Hide In Service Catalog	Select this field to hide the service in Service Catalog module and the associated requests for this service in Order Status and Completed Orders view in My Stuff. This is useful if the Service Designer does not want these service requests to be visible or cancellable by the end user. Often these are child services that are invoked as part of a more complex service bundle.
Max Quantity	This option allows you to configure the maximum quantity of a service that can be ordered in a single order. For more information, see Configuring Maximum Quantity of a Service, on page 162.

Field	Definition
Description	A text field for entering a succinct description of the product or service. This information is displayed to the consumer in **My Services** or **Service Catalog** and should be as clear as possible so the end user knows exactly what they are ordering.
Service Level Description	This is an optional text field. Anything entered here is displayed to the consumer in My Services or Service Catalog and should be thought of as a promise to the consumer about what level of service they should expect. The description may not exceed 4000 characters.
Standard Duration	The Standard Duration is the standard (typical) delivery time for this service after all authorizations and reviews have been completed. This information is available to the customer before they request a service. *The Standard Duration is not the due date for the service to be completed and is not used to calculate due dates* . This field does not take into consideration the calendars of the service fulfillment team. It calculates hours into days, and is used to calculate the Standard Compliance metric for the service used in Advanced Reporting.
Display Units	The Standard Duration can be displayed in Hours or Business Days. If you choose to display by business days, the number of hours is converted to days using the “Working Hours per Day” setting on the Plan tab. For example, if the standard duration for a service is 3 business days, you would enter 24 hours in the Standard Duration field, assuming an 8-hour workday. As you compile and arrange your service delivery plan, the system keeps track of the total number of hours required to complete all of the plan’s activities. It also uses the “Working hours per day” setting (Plan tab > Tasks subtab; see the Table 50: Fields in Task Subtab) to compute how many working days are required to execute the plan.
Forecasting Method	Choose the method that will be used to forecast the service due date to the end user in My Services after the service is submitted. This forecast should always be considered approximate. No matter which forecasting method is chosen, Service Catalog recalculates all due dates after all authorizations and reviews for the request have been completed. Choosing a forecasting method has both functional implications (what the requestor of the service will see, and when) and performance implications. See the Forecasting Due Dates, on page 162 for a detailed discussion.
Additional URL	The Additional URL field can be used to further describe your service or link to supporting information. For example, if the service is for desktop software, you may want to include a link to an external product site so the end user can read system requirements or ensure they are ordering the correct software. The URL must be fully qualified, beginning with “http://”. The service description will include a “More Information ...” link, to access the specified URL.
Functional Positions	Available functional positions and their corresponding personnel assignments for the service. Functional positions are defined in Organization Designer. You can add positions and assign people to the position, for this service. For more information, see Configuring Functional Positions , in Cisco Prime Service Catalog Administration and Operations Guide .

Field	Definition
Keywords	The search facility in **My Services** returns services containing any word in the service name and service description. Add additional keywords to help facilitate a user search. For example, if the service is “Order a New Laptop” and the user can choose either a Dell or a Lenovo laptop, additional keywords might be: 'Dell' and 'Lenovo'. These are used for customer navigation only, and are not used in reporting. For more information, see Setting Keywords for Search.
Display Categories	The categories in which the service appears in **My Services** or **Service Catalog**. For more information, see Categorizing Services, on page 12.

Configuring Maximum Quantity of a Service

Max quantity functionality allows you to configure the maximum quantity of a service that can be ordered in a single order. You can configure Maximum quantity at Service level and Category level from the Service Designer module.

To set max quantity at service or category level, navigate to **Service Designer > Services / Category > General** and enter a value for **Max Quantity**. The Maximum Quantity field takes any value ranging from 0—999.

Keep in mind

- If max quantity is configured at service level and at category level, then service level max quantity value overrides the category level max quantity value.
- If max quantity is configured at category level and for some of the services max quantity is configured at service level, then for those services alone service level max quantity will be considered.
- If a service belongs to multiple categories and max quantity value is not defined at service level, the least max quantity value of all the categories will be considered.
- If a service is bundled, the max quantity value of the parent service is applicable.

Unless you enable **Allow Update Quantity** option from the **Administration > Settings** My Services section, the max quantity configured in Service Designer is not effective.

The order submission nsAPI also uses max quantity to validate across applications i.e., cart, order management, Soap call, RAPI, and nsAPI.

Forecasting Due Dates

The methods for forecasting due dates are summarized in the table below and explained in detail in the following paragraphs.

Table 48: Forecasting Due Dates

Method	Functionality	Performance Implications
Estimate Due Date from task durations	The system forecasts the due date based on all tasks in the delivery plan, the anticipated duration for each task, and the assigned performer.	All tasks (authorization/review and delivery) are instantiated, and individual queue or person calendars are consulted to determine scheduled task start and end times.
Approximate Due Date using Standard Duration	The system uses the calendar associated with the working hours of the Default Service Delivery queue to approximate due dates, rather than consulting the actual participant assigned to each task.	All tasks (authorization/review and delivery) are instantiated, but individual queue or person calendars are not consulted.
Do not forecast Due Date	No due dates are forecast when the order is submitted. The user sees "TBD" as the Due Date in My Services.	Only authorization/review tasks are instantiated when the request is submitted.

There are both functional and performance implications in choosing the method for forecasting due dates. The following factors affect the accuracy of due dates forecast:

- Any authorizations or reviews associated with the service. Any initial forecast uses the specific duration associated with each of these authorizations. However, due dates are recomputed after all authorizations are completed. Variations in completion time for authorizations and reviews (typically, taking longer than specified in the delivery plan) can give an unrealistic forecast.
- Any conditional tasks included in the delivery plan. Since the conditional expression governing execution of the task can only be evaluated in the service delivery moment, any forecast cannot take conditional task execution into account. Therefore, the estimate may be inflated by included tasks that would actually not be executed. On the other hand, the estimate would always be a worst-case scenario (assuming execution of all tasks), so users could be pleasantly surprised if the service request is fulfilled sooner.

When due dates are forecast (either via Approximation or Estimate), Service Catalog must create (instantiate) all tasks in the delivery plan. This may take a significant amount of server processing time, depending on the number of tasks in the plan. Estimating the due dates will always take longer, since the work calendar assigned to each participant in the plan must be consulted to correctly derive the task start and end dates.

The Administration setting to "Submit, Approve, and Review Asynchronously" affects the perceived performance of Service Catalog when it comes to forecasting due dates. By default, this setting is off, so that "background processing of requisition submit" is disabled. Therefore, Service Catalog instantiates these tasks synchronously; that is, the user submits the order. Service Catalog creates the tasks as instructed in the forecast method, and then control of the page is returned to the user. For complex task plans, the user may wait a significant amount of time to proceed away from the order form.

If "Submit, Approve, Review Asynchronously" is set to on, Service Catalog creates tasks asynchronously; that is, in the background. Consequently, the user does not have to wait until all tasks have been created to exit from the order form and continue using Service Catalog. The wait time is eliminated. The difference in

performance may not be obvious for requests with few (or no) authorizations or with simple delivery plans, but should be apparent for services with more complex workflows. After requisition submission, the status becomes “Ordered” until it is processed by the Business Engine. Afterwards, the status becomes “Ongoing”.

Because asynchronous processing requires additional configuration steps and it might not produce any perceived difference in performance for some installations, configuring Service Catalog to “Submit, Approve, and Review Asynchronously” is optional. Be sure to check with your system administrator to see if this setting has been enabled before deciding on a method for forecasting due dates.

Creating Custom Templates

Prime Service Catalog provides out-of-box templates using which you can then enhance it as per your requirement to create custom templates. These templates are non-orderable services which are used to generate Cloud Center application profiles services or UCSD services. The template defines the appearance of the orderable services created from UCSD or Cloud Center in Service Catalog module. These templates have the basic configurations required for the specific types of UCSD services or Cloud Center application profiles services .

The UCSD out-of-box service templates are located in the *Reserved Services* service group and the Cloud Center templates are located in the *Cloud Center Reserved Services* folder. The available types of out-of-box templates are:

- Standard Catalog
- Advanced Catalog
- Container Catalog
- UCSD Template
- Application

By default, these templates are mapped to the respective types of UCSD Catalogs and container templates. And for Cloud Center connections and application profiles, Application template is mapped. All the custom templates created are also available for mapping the UCSD and Cloud Center services, they are filtered out based on the type of templates.

1. In UCSD the custom templates can be mapped at catalogs or templates level.
2. In Cloud Center the custom templates can be mapped at connection or application profile level.

You must track and map the newly created custom templates appropriately.

Note:

1. Custom templates created based on VACS out-of-box templates must not be mapped to any other services.
2. APIC Container template and Fenced container custom templates must not be mapped interchangeably.

For more information on mapping these templates see sections *Mapping the Custom Templates for UCSD Services* and *Mapping Custom Application Templates for Cloud Center* in Cisco Prime Service Catalog Administration and Operation Guide.

To create custom templates:

Step 1 Go to **Service Designer > Services**.

Step 2 From the Reserved Services folder (which contains the UCSD out-of-box templates), select the required template which will act as a base for the custom template.
For Cloud Center templates go to the Cloud Center Reserved Services folder.

Step 3 In the General tab, click **Clone** to clone the base template.

Step 4 Optionally, enter a new service name and select the group to which the service must belong.

Step 5 Uncheck the Reserved Form to retain the basic configuration settings of the template.

Step 6 Optionally, you can enhance the presentation of the template. For information see Formatting Service Presentation, on page 240 .

Step 7 Add new forms from the **Form** tab.

a) Click **Add Forms** in the bottom left of the window. The Add Form popup window appears.
b) In the **Search field**, enter desired form name.
c) Check the required form from the search results then click **Add**.

Step 8 Additional task can de added such as Email Notifications.

Step 9 Click **Save** to create the custom template.

Configuring Bundles of Related Services

A bundled service is a service that contains one or more related services that are automatically ordered when a customer orders the bundle (parent). Bundling is a convenient way to group services that are commonly ordered at the same time. For example, all new employees might require the following services:

- LAN ID
- Desktop or Laptop PC
- New Phone with Voicemail

You could create a bundle that contains each of these services, rather than expecting users to remember to order all three services for a new employee.

In a bundle, the new service containing two or more related/existing services is referred to as the *parent service* . The services contained by a parent service are referred to as *child services* .

Work flows for Bundling Services

Creating and processing a bundle involves several different modules and tasks within Service Catalog. Here's a brief overview.

- **Create**: After creating service groups and services, use the Service Designer module to create a bundle.
- **Order**: A bundle can be ordered from the Service Catalog module. The users may not realize that a service is a bundle. For example, if they click the Order link when choosing the service from the services catalog, the composite order form for the bundle appears, and if they review the order form, only the

parent service appears. However, if a user clicks the service-name link as it appears in the service catalog, they can then click links for each of the child services and see detailed information. They cannot however, order any of the child services individually this way. After an order is submitted, the Service Order Confirmation page lists all services included in the bundle. If an order for bundle has to be canceled, the complete order must be canceled; as the individual child services included with the bundle cannot be canceled.

- **Schedule**: Service Catalog treats the order like any other set of requisition entries on a requisition: It schedules
  - Authorization and review steps defined for the parent service, which override any authorizations defined for the child services.
  - Delivery plan tasks for each of the services.
  - Plan-monitoring tasks for each of the services.
- **Deliver**: The delivery plan of each child service executes as if the child service had been ordered on its own. The scheduling of the delivery-plan tasks for the child service depends, however, on the position of the included task for the child service within the delivery plan of the parent service. For example, if the parent service lists the child services of LAN ID, Desktop, and New Phone, then the tasks would be completed in that order. (This assumes that you chose to complete top-level tasks sequentially in the Delivery Plan. If you want the delivery plans for the child services to execute concurrently, you could change this setting.)
- **Monitor**: If Service Manager users monitor a plan, they can see the plan-monitoring task for any service on the Plan tab, in Service Manager. The plan-monitoring task for a parent service shows the included tasks for each of the child services. If the site configuration parameter “Show Task Link” (available in the “Personalize Your Site” folder in the Administration module) is set to On (its default), the tasks shown on the Plan tab are links to the actual task forms. For included tasks, these links take you to the plan-monitoring tasks of the included services.

The table below provides you the various tasks that you can do for a service bundle.

Table 49: Tasks for Bundling Services

Tasks	Procedure	Description
Create a Bundle	To create a bundle: **1** Choose Service Designer > Services > Offer > Bundle. **2** Select **This service is a bundle**. The system automatically checks the "This service cannot be included in a bundle" option because Service Designer does not support bundled services within bundles. The "Include Service" button becomes enabled. After you add a service to a bundle (making it a child service), Service Designer disables the two check boxes and instead displays a list of the bundles in which the service is included. **1** Click **Include Service**. The "Select a service" dialog box appears. This dialog box lists only services for which the "This service cannot be included in a bundle" option is not checked. **2** Choose the service that you wish to include in the bundle, and click **Include Service.** The child service appears in the "Includes section on the Offer/Bundle" subtab. The child services appear in "My Services" as part of the detailed information for the parent service (on the Included Services subtab). **3** Repeat Steps 3 and 4 to add other child services to the bundle.	In Service Designer, begin by choosing the service you wish to use as the parent service or create a new service to act as the parent. Once you have chosen the service, click the **Offer** tab to get started. The assumption is that you have already created the child services that you wish to include in the bundle.

Tasks	Procedure	Description
Associate other services with a Bundle	To associate other services with a bundle: **1** Choose Service Designer > Services > Offer > Bundle. **2** Click Add Pre-requisites in the Pre-requisites table. **3** Select a service and click Add Alternative method: **1** Choose Service Designer > Services > Offer > Bundle. **2** Click Add Accesories in Recommended Accessories table **3** Select a service and click Add	Prerequisites and Recommended Accessories allow service designers to associate additional services with the current service. These options may be specified both for bundled and nonbundled services. • **Prerequisites** are other services that are required before ordering this service. For example, a customer must have a computer purchased and installed before ordering software installation. • **Recommended Accessories** are other services that are recommended as add-ons to the service. For example, if the service is for a cell phone, recommended accessories might be a headset and case. If a service has prerequisites or recommended accessories, the corresponding link appears to the right of the service's detailed description in My Services. There is no behavior associated with either the listed prerequisite services or accessories that must be ordered individually. These are optional but can be helpful to the end user.
Prevent Bundling	To prevent bundling a service: **1** Choose the service that you want to prevent from being bundled. **2** Choose Service Designer > Services > Offer > **Bundle**. **3** Choose the **This service cannot be included in a bundle** option.	If you want to prevent other service designers from including a service in a bundle, complete the following procedure. Services marked in this way do not appear in the "Select a service" dialog box when choosing child services.
Change Order of Child Services	To Change order of Child services: **1** In the "Includes" section, check the check box of the service that you want to move. **2** Click the Up- or Down-Arrow icon to move the service.	If desired, you can make one of the subtasks conditional to effect an "opt-out" scenario. In this case, the child service still appears to have been ordered (with all tasks skipped).

Tasks	Procedure	Description
Change an Included Task Name	To change an included task name: **1** Choose Service Designer > Services > Plan. **2** Click the task name. The General tab appears with information about the task. The "Task Type" indicates the task is an Included Service. The "Service Name" identifies the service that the included task represents. (This is important information if you change the name of the included task.) **3** Enter a new task name in the "Task Name" field, and then click **Update**.	
Review Included Participants	To review include participants in a bundle: **1** Choose the parent service. **2** Click the **Plan** tab. **3** Click the **Participants** subtab.	For each child service you add to a bundle, Service Designer automatically assigns participant information into the delivery plan of the parent service. The information on the Participants tab is taken from the Project Manager of the child service. Service Designer automatically assigns both the Performer and the Supervisor of the task using whatever is currently defined as the Project Manager of the child. If you change the Project Manager of the child service, the change is reflected dynamically for the included task. Service Designer names the performer of the included task the Subplan Manager. It names the supervisor the Plan Manager.

Tasks	Procedure	Description
Pricing Bundles		The price of a Bundle includes the price of each included service, as defined in each child service. You set the price of any service on the Pricing subtab of the service's Offer tab. The price could potentially be adjusted by using the Set Price action in an active form component rule, applied either to the bundle as a whole or to its component services. For your convenience as you construct a bundled service, the price of each child service is listed on the parent service's Offer tab, Bundle subtab. If a child service's price is defined as Pricing required, the system schedules a pricing step for that service at the time a My Services user orders the bundle.
Discount Price of a Bundle	To discount a bundle: **1** Choose Service Designer > Services > Offer > **Bundle**. **2** From Services panel, click the parent service. **3** Click the **Offer** tab, and then the **Pricing** subtab. **4** Enter a negative value for the price. For example, to discount the service bundle by $1000.00, enter –1000.00. **5** Click **Save All Settings**.	The price of a bundle can be discounted in either of two ways: • Use dynamic pricing, via the Set Price action available in active form component rules, to reduce the price of a service when it is used in a bundle. • Use the Pricing subtab of the parent service. There you can set a negative price for the parent service, which is subtracted against the total cost of the included services. For example, if the cost of all child services adds up to $5,000 and you enter a negative price of $1000 for the parent service, the net cost for the bundle becomes $4000. By doing this, you encourage the end user to order the bundled service rather than each service individually. Customers can see that the price has been discounted only after they've clicked Order for the bundle. To let them know about the discount up front, you can use the service's Description field on the General tab, or the Description field on the Offer tab/Pricing subtab, to briefly highlight the financial advantage of choosing the bundled service.

Tasks	Procedure	Description
Control Display Parameters on the Order Form	Choose Administration > Home > Personallize your Site> Customizations > Service Manager > Show Bundle Data	When the system creates the composite order form for a bundled service, any duplicate dictionaries are eliminated. This means that if a dictionary is associated with a parent service and one of the parent's child services (or multiple child services), the system will display only the first instance of that dictionary on the bundled service order form. There are two options to display service form data: • **ShowBundleData = On** (default) shows service performers the composite order form for the bundle when processing work items within any of the child services. • **ShowBundleData = Off** shows service performers only the dictionaries from the child service they are working on. The system ensures that the data for the bundled service duplicates the data entered for one dictionary in all instances of that dictionary. So if a dictionary from a child service was suppressed on the original order form (because it duplicated a dictionary in the parent service), any service performer processing just the child service will still see the data entered by the customer.

Tasks	Procedure	Description
Review Included Tasks	To review included tasks in a bundle: **1** From the Service panel, choose the parent service. **2** Choose Service Designer > Services > Plan . The child services are automatically named following this convention: “Deliver Included Service *service-name* .” For example, a service might appear as “ Deliver Included Service Phones.”	For each service you add to a bundle, Service Designer automatically inserts a task into the delivery plan of the parent service. This task is referred to as an included task. Included service tasks behave like most delivery-plan tasks, with the following exceptions: • You cannot delete an Included Service task. The Delete button on the Plan tab is disabled. To delete this task, remove the corresponding child service from the bundle. • You cannot set Duration values. These are calculated as the total Duration from the child service’s delivery plan. • You cannot edit the Task Type. This is set to “Included Service” by default. • You cannot create or associate checklists. There is no Checklist tab for the task. • You cannot edit the information on the Participants tab. Although the task is automatically named “Deliver Included Service *service-nam* e,” you *can* change the task name if desired. You can also arrange included tasks in a task/subtask relationship by indenting one under another. However, the delivery of both services will start at the same time.

Tasks	Procedure	Description
Restricting Cancellation of Child Services in a Bundle	To ensure that the customer cannot cancel the service after a task in the service bundle has started 1 Choose Service Designer > Service > Plan. 2 Select a task from the task table. 3 Click the "Do not allow cancellation of service after task starts" check box on the General subtab.	A customer can cancel a bundle once it has been ordered. Canceling a bundle cancels all included services. The option to cancel individual child services is disabled for the end user on the Edit Requisition page. Service Designer essentially defines "the point of no return" for the customer to cancel a service. This option can be checked for the entire bundle of service or for individual subtasks within. However, once this point has been reached for *any* of the services within the bundle, the customer cannot cancel the bundle. The "Do not allow cancellation of service after task starts" check box on the service's Plan tab, General subtab in Service Designer essentially defines "the point of no return" for the customer to cancel a service. This option can be checked for the entire bundle or for individual child services within, but once this point has been reached for any of the services within the bundle, the customer cannot cancel the bundle. Additionally, when "the point of no return" has been reached for any service or subtask within the bundle, the Cancel button is removed for the service order.

Note For namespace usage, see Namespaces, on page 315.

Customer View of Bundles

To order a bundle in Service Catalog, click the service name to view the service details and then click "Included Services" on the Details page to see all services included in the bundle.

Click on each service name to view the specific details related to that service. At this level, a Return to Service Bundle button displays *in place of* the Order Service button, which means a child service cannot be ordered from within a bundle unless the entire bundle is ordered. To order services separately, the return to the service catalog.

When ordering a bundle, the Order Confirmation page displays the names of all services in the bundle including the Due Date for each child service.

The Track Resolution page displays a list of all child services in the bundle. Click the link to view the order form and to view the Delivery Process for each service as it is performed.

Defining Service Level Permissions to Order a Service

The **Permissions** tab shows which users are allowed to order this service. To grant permission to order a service:

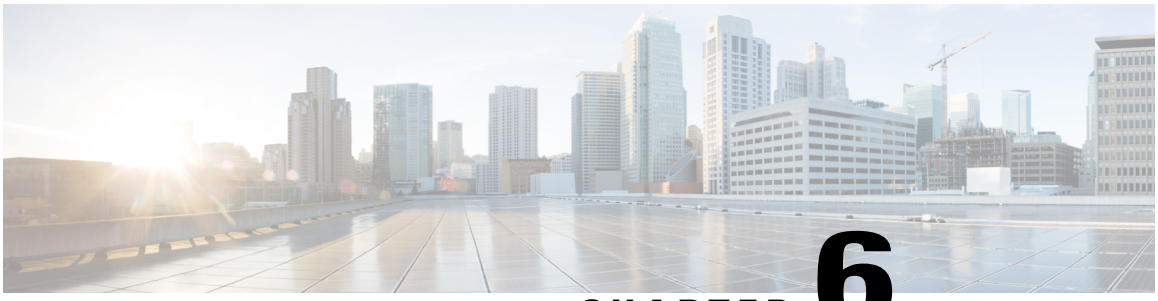
Step 1 Choose Service **Designer** > **Services** > **Permissions**tab of a service you wish to allow participants to order.
"Order service" is the only permission that can be assigned at the service level; it is already displayed.

Step 2 Click **Add Participants** to add People, Organizational Units, Functional Positions, Groups, Roles, or to allow access to Anyone.

Step 3 Search and choose the entity or entities you wish to add.
People, organizational units, functional positions, groups or roles previously created/defined in the Organization Designer module (or via Directory Integration if this is implemented) can be searched and chosen.

Step 4 Click **Add** to grant the chosen entity permission to order the service.

Granting access to "Anyone" allows any Service Catalog user to order this service. Only services that are truly universal in nature, and open to the entire customer base, should have Anyone as a grantee. The "Site Administrator" role, by definition, has permission to order any service.



CHAPTER 6

Designing Plan for Delivering Services

This chapter contains the following topics:

Designing Plan for Delivering Services

A delivery plan comprises one or more tasks that must be completed to deliver a service to a customer. The **Service Designer** > **Service** > **Plan** tab is used to define the delivery plan for a service.

Before You Begin

- Identify what the tasks (or series of tasks) are
- Identify whether the service requires email notifications
- Identify who the performer, supervisor, and email recipients are

Also, have an understanding of the following concepts

- Using Namespace variables and configuring expressions, as detailed in Namespaces, on page 315.
- Configuring and using email templates, as detailed in the Cisco Prime Service Catalog Administration and Operations Guide . For namespace usage in email templates, see Namespace Usage in an Email Template, on page 319.

- Calculation of due dates. See Forecasting Due Dates, on page 162.

Step 1 Configure the Project Manager for the service delivery process.

Step 2 Configure individual tasks, including task name, duration, conditions, priority and other parameters.

Step 3 Specify the performer and supervisor of the task.

Step 4 Specify notification emails related to the task.

Step 5 Configure task instructions.

Step 6 Create a task checklist.

Step 7 Specify the workflow for the tasks, including the sequence for the execution of tasks, whether tasks execute concurrently or consecutively, and potentially grouping tasks that share a common milestone or have common notification requirements.

Configuring Delivery Tasks

Every delivery plan automatically has one overall task, that allows a designated project manager to monitor the delivery plan's progress. Using the **Tasks** subtab you can specify the tasks and detailed activities for each task of the service delivery plan. You define the tasks by listing them in the appropriate order; subtasks and parent tasks are indicated, much the same way as Project Management software shows projects and deliverables within a project.

Step 1 Select **Service Designer** > **Services**.

Step 2 Select the service and choose **Plan > Tasks**. If multiple tasks are listed on the Plan tab, click on the task name you want to configure.
When the task is highlighted in blue/gray, you are ready to begin; each task you want to configure has its own set of subtabs: General, Participants, Email, Task Instructions, and Checklist.

Step 3 Define the task details. For more information, see Table 50: Fields in Task Subtab

Step 4 Add tasks. Ideally you should be able to move through the fields described below in order as you create detailed delivery activities.For more information, see Table 51: Fields in Task Table.

Step 5 Click **General** sub tab and create detailed plan activities for delivery of a task including: the task name, execution order, duration, priority, and conditions.
If your company's installation is integrated with third-party software using Service Link, then you also choose the external action here. If you are using Service Item Manager, you can also choose Service Item tasks here.

For more information, see Defining Delivery Activities for a Task, on page 179.

Step 6 Click **Participants** sub tab and define the details about the performer of the task, including who completes the task and how the work is supervised.

Step 7 Click **Email** subtab and define the optional email templates that the system sends when the task starts, completes, is cancelled, is rescheduled, is reassigned, or when an external task fails. For more information, see Defining Email Notifications for Delivery Tasks, on page 205.

Step 8 Click **Task Instructions** and enter instructions for the task or set links to any task instruction-related URLs. For more information, see Adding Task Instructions, on page 206.

Step 9 Click **Checklist** sub tab and create a list of reminders or detail the procedural steps for completing a task. These appear in Service Manager as a checklist, detailing steps for completing the work. For more information, see Creating Checklist For Completing a Task, on page 206.

The Table 50: Fields in Task Subtab table summarizes the fields on the Tasks subtab that pertain to the "Monitor" task, that is, the overall delivery plan. For more information, see Configuring Bundles of Related Services, on page 165

Table 50: Fields in Task Subtab

Field	Description
Project Manager	Assigns a project manager from a position, a person/queue, or an expression. The project manager is the person, position, or queue that receives the Plan Monitor Task for a service in Service Manager. This task allows to oversee the entire delivery process including making adjustments to staffing, rescheduling and completing tasks, and even cancelling the entire delivery plan, if needed. Clicking on the "..." button on the right pops up a dialog box allowing you to search for and designate a position, a person/queue, or to type in the parameters of your expression. For an expression, click **Set Expression** to save.
Subject for plan monitoring task	Text describing the subject of the monitoring task for the plan. It is recommended that you place the word "Monitor" in the subject. The plan subject may include namespaces, as documented in Namespaces, on page 315. Clicking on the "..." button allows you to edit the subject. Click **Set Subject** to save changes.
Top level tasks execute	Indicates whether top-level tasks in the delivery plan execute concurrently (at the same time) or sequentially (in order, one after the next).

Field	Description
Start and complete plan automatically?	This check box is checked by default. This means that all tasks in the plan are automatically created, and their due dates computed, when the delivery plan begins. If you uncheck the check box, then the Project Manager assigned must take action before the delivery plan goes into effect. When the delivery moment begins only one task is created—the MONITOR task. That task presents a button labeled “Start Plan”. This allows the Project Manager to “staff” the plan (using the Staffing page on the MONITOR task). On that page the Project Manager can reassign any performing position, person, or queue in the delivery plan. When the Project Manager has finished Staffing the plan, they click **Start Plan**, which initiates the first tasks in the delivery plan. Until that button is clicked, tasks are in “Staffing” status.
Allow future delivery	Check this to allow end users to request a service in advance of when service delivery should begin. Once this option is checked, the end user sees a bar across the top of the ordering page which allows them to choose the date they want the service delivery to begin. End users cannot specify the due date/completion date for a service. If the user selects a future delivery date, Service Catalog holds the order in a waiting state until the date chosen by the user. Beginning on that date, the service delivery process begins, and proceeds as configured.
Notify when plan cancelled	Allows the service designer to configure an email to be sent when the Project Manager cancels delivery of a service.
Working hours per day	Used to convert hours to business days for the Standard Duration. See the Configuring Delivery Tasks, on page 176 and Table 52: Configuration Table to Define Delivery Activities for a Task table to set the Standard Duration and Display Units for the service.

Defining Delivery Tasks

The table below summarizes the fields on the Task table.

Table 51: Fields in Task Table

Field	Description/Usage
New	Add a new task to the delivery plan.
Up Down	Change the order of tasks in the task list.
Indent Outdent	Group or ungroup tasks. For example, if you create a task and click **Indent** , the task becomes a subtask of the task above it. **Note** Use the UP, Down, Indent, and Outdent fields to configure the workflow so that tasks execute in the appropriate order and are grouped correctly.
Delete	Delete a task (and all subtasks beneath it).

Defining Delivery Activities for a Task

To define detailed delivery activities for a task or series of tasks:

Table 52: Configuration Table to Define Delivery Activities for a Task

Field	Description
Workflow Type	The default is Internal. This is the only option displayed for a site that has no Service Link integrations defined with external systems and no Service Item tasks. Choose **Internal** if the task is executed within Service Catalog. Choose the appropriate **External** option if the task is executed in an external application (not within Service Catalog). For external tasks only, an ellipsis (...) button appears next to the Workflow Type after you have saved the task. See the Configuring an External Task , on page 192 for more information. Choose Service Item **Task** if you are using Service Item Manager module to design service items and track these items. More information on service item tasks is given in the Defining Service Item Task in a Delivery Plan, on page 188 and in Managing the Services and Attributes, on page 247. Choose Queue Service Request to publish the Service Request data to a rabbitmq message broker server. An external system subscribed to rabbitmq broker and the specific exchange will receive the message and can execute its own work flow. Choose NsApi Task to grant or revoke permissions for a service item or multiple service items that are created using grid dictionaries within the service delivery plan. For more information, see Creating Service Delivery Tasks for Granting and Revoking Permissions for a Service Item, on page 201. Choose Directory Task if the operation is for adding or updating Organization Designer entities (organizational units, people, queues) or Service Link agents. See Defining Delivery Tasks, on page 178 for more information.

Field	Description
Task name	A brief title for the task. This appears as the subject of the task in Service Manager. The delivery task name may include the service name, using the #Name# namespace. Service data (#Service.Data....#) can also be used. Using form data for a task name allows task performers to more easily differentiate tasks in Service Manager. However, it presents challenges in the reporting modules, since the task would no longer be automatically groupable by the task name; report designers would need to use “Custom Groups” to aggregate by such tasks. It may also require administrators to configure Service Manager to allow “contains” searches, which may adversely affect performance. Service designers should think carefully before including form data namespaces as part of a task name.

Field	Description
Create Agent	After you have saved the task, a Create Agent button appears. Click this button to use the Integration Wizard. See the Cisco Prime Service Catalog Adapter Integration Guide for complete details. **Note** The Integration Wizard is available only to those service designers who have been granted a role that allows creation of Service Link agents and transformations. The Integration Wizard automates many of the steps involved in implementing an integration. It is available only for creating integrations between Service Catalog and web services. The Integration Wizard works by retrieving the Web Services Description Language (WSDL) and operation to be invoked by the web service integration. Based on that definition of the integration, the Integration Wizard creates: • The Service Link agent that can be used in an external task to perform the integration • A transformation to transform nsXML into the SOAP message required by the web service • Agent parameters for all data required both in the initial web service request and the response • A dictionary containing fields mapped to the agent parameters • An active form component containing the dictionary created to hold agent parameter values
Subtasks execute	Use the drop-down menu to choose the order in which any “child” tasks of this task execute: one after the other (sequentially) or at the same time (concurrently). Child tasks (subtasks) become active (ongoing) in the delivery plan, and must be completed before their parent task becomes active. If child tasks are executed sequentially, the service duration includes the durations of all such tasks. If tasks are executed concurrently, the service duration includes the maximum duration of any child task.

Field	Description
Duration	The expected length of time between when the system says the task is active (begins) and when the task is completed, rounded to two decimal places. For example, if installing a software program only takes one hour, you might give the performer a duration of 16 hours to finish the task, as the performer's workload and priorities may vary throughout the course of two workdays. The Duration value is not visible to the customer or performer, but is used to calculate the due date for the task.
Effort	The actual length of time it should take the performer to complete the task, rounded to two decimal places. For example, it might take one hour to install a software program. While Duration is “total elapsed time”, Effort is “time on task”. The Effort value is not displayed to any users or used in due date calculations; it is used instead as an Internal Productivity Target.
Priority	Set the task priority to low, normal, or high. No system behavior is tied to the priority for a task, other than a flag which is visible in Service Manager. • If the priority is set to **high**, the system sets a red exclamation point in the Service Manager view where the task is displayed. • If the priority is set to **low**, the system sets a blue down-arrow in the Service Manager view where the task is displayed. • If the priority is set to **normal**, no flag is displayed in Service Manager.

Field	Description
Condition	

Field	Description
	A conditional statement allows tasks to be initiated or skipped based on the whether the expression used in the condition evaluates to true (include this task) or false (skip the task). The expressions are formulated using the namespaces and operators documented in Namespaces, on page 315. After you enter an expression, click **Validate** to make sure that the expression is correct. Validation checks if: • Namespace specified is valid for the current scope (the specific level of review/authorization or task), with the exception of dictionary fields (Data.*DictionaryName.FieldName*). However, it may cause a runtime error: if the specified namespace, for example, Data.EUIT_ACCESS.Access_Type, does not exist. • Correct relational and arithmetic operators are used. The message “unexpected token” indicates that the namespace used is not valid in this context or, perhaps, that you have forgotten to enclose an alphanumeric literal in quotes. If a condition has been entered, you must decide when that statement will be evaluated. The condition for each task (approval, review, or delivery) can be evaluated either: • At the beginning of a phase (Authorization or Delivery moment), or • When the activity becomes active. Use a condition that always evaluates to false (for example, “1=2”) in a conditional statement to specify a task that will automatically be skipped. The most common use cases for this are: • When a service needs to “auto-complete” without having any tasks completed. The skipped task is the only task in the delivery plan. When it is skipped, the requisition is marked as complete. • When an email needs to be sent without having to complete a task or when multiple emails are needed during a given moment in the following

Field	Description
	ways: ◦ Create a parent task. On the Email subtab, choose an email to be sent out at completion. (You may also configure a notification to go out when the activity becomes active.) For more information, see Defining Email Notifications for Delivery Tasks, on page 205. ◦ Create a child task with the condition 1=2. Set the condition to evaluate when the activity becomes active
Allow a scheduled start date	To make the task a delayed task, check **Allow a scheduled start date** .Then,in the Form data for start date field, do one of the following: • Enter a date and time in the following format: “YYYY-MM-DD HH:MM” such as “2006-12-23 13:30” (The quotes are required.) • Or, enter the name of the dictionary field you want to use to hold the date and time of the starting-point for the task. Use the following syntax: ◦ Data.DictionaryName.FieldName for a field from a dictionary in one of the service’s active form components. ◦ ParentData.DictionaryName.FieldName for a field from one of the parent service’s dictionaries (in the case of a bundled service). For more information about delayed tasks, see System Behavior for Scheduled Start Tasks.
Form data for start date	If you have checked the “Allow a scheduled start date” option, you may enter form data for the start date and click **Validate** .

Field	Description
Evaluate condition when delivery phase starts	If you have specified a condition, click the option button to indicate when the condition will be evaluated. Choose this if you want the condition to be evaluated when the delivery moment starts for this service. This means that the information required to correctly evaluate the expression must be available before the delivery moment begins. Each authorization or review has its own moment; all delivery tasks are performed within the Service Delivery moment. Authorization tasks are always serial. You could put one conditional on one task saying if field = “x” and a conditional on another saying field < > “x”. That way, you know one authorization task will always be executed, and if you choose “when authorization phase starts” only one authorization task will appear in the process view. If you choose “when task becomes active” both tasks would be displayed, but one would be skipped. The “**if conditions evaluate to “false”, times will be computed as zero**” statement means that Service Catalog will evaluate the conditions at the beginning of the phase. If these conditions are false, then the corresponding tasks will not be executed, and the **Due Date** for the service will be calculated without including the duration of these tasks.
Evaluate condition when task becomes active	If you have specified a condition, click the option button to indicate that the condition must be evaluated only when the delivery process has reached this task in the delivery plan. Service Catalog evaluates the condition at the beginning of each task. If the condition is false, the corresponding task will not be executed. Durations for any task configured with this option will be used to calculate the due date upon submission.

Field	Description
Re-evaluate expressions as plan advances	This feature enables you to dynamically change the performer assignment for a task, as well as the name of the task, based upon form data. This is useful if you have designated that a task name or a performer assignment should be derived from a namespace expression using form data, and when the form data will be changed during the delivery process. Setting this flag specifies that the expression should be evaluated when the task becomes active (not when the entire delivery plan is initiated). If this option is not checked, all information used in expressions in the authorization task must be present during the Ordering moment. The Re-evaluate Expressions feature is useful for designs in which there are multiple sequential authorizations or tasks. It enables the person performing a task to enter information in the service form that is used to recompute the expression used to assign the performer for a subsequent task. This feature allows dynamic assignment of a task to a user (person or queue) and dynamic adjustment of the task title. Once the task becomes active, the expression is evaluated and the task is assigned appropriately. **Note** The re-evaluation of the performer for a task does not affect the due date for the task. Due dates are not recomputed after the delivery moment begins.
Do not allow cancellation of service after task starts	If you select this check box, the customer will no longer be able to cancel the service after this task becomes active. For example, check this if the task will cause the delivery team to incur substantial costs or take an action that is not easily reversible. The “Cancel” option in My Services is deactivated once the delivery team begins this task.
Display Effort subpage on a delivery task	If you check this box, an “Effort” subtab for a delivery task is shown in Service Manager.

Defining Service Item Task in a Delivery Plan

This section covers the functions of service item tasks. The Service Item Manager module allows designers to designate certain items to be “service items”, history can be tracked within Service Catalog. Typical service

items might be a laptop; desktop; software license; or any corporate asset that can be uniquely identified and whose usage (and ownership) should be tracked. A Service Item Task can be used only when the service includes a Service Item-Based Dictionary (SIBD).

Step 1 Choose **Service Designer** > **Plan** > **Task** > **General**.
a) Choose **Service Item Task** as the Workflow Type and save the plan.

Step 2 Click the ellipsis (...) that appears next to the Workflow Type.
The Service Item Task Parameter popup window appears.

Step 3 Choose the SIBD to use.

Step 4 Select **Update Status Only** check box if the operation should modify only the Status attribute value of the service item.

Step 5 Choose the operation to be applied from the **Operation** drop-down list

Step 6 Click **OK** to save the task definition and dismiss the popup window.

Step 7 If the SIBD is a grid dictionary and the grid rows should be processed conditionally based on certain field values, enter the condition expression in **Process Grid Row**, click **Validate**, and then click **OK**.
See Configuring an External Task , on page 192 for syntax and example for a condition expression.

Service Item Subscription Processing Rules

The customer and organizational unit information within the Service Item Subscription table can be set independently from that of the requisition:

- If no subscription information is provided in the create operation, the item is assigned to the customer of the requisition and that person's Home Organizational Unit.
- If only the Customer ID is specified at the time a service item instance is created, the Organizational Unit ID of the item is set to the home organizational unit of the customer.
- If a value is provided for either the Customer ID or Organizational Unit ID, that value is used to update the service item subscription. If that value is either null or zero, the corresponding subscription field is set to null. The absence of a property in the service item-based dictionary means no change/override on the attribute value.
- Requisition ID and Requisition Entry ID provided in the dictionary are ignored and will not be used to update the subscription record.

The possible combinations for customer and organizational unit assignment and the outcome are summarized as follows:

Table 53: Customer and Organizational Unit Assignment Combinations

When service item is created	
Service Item-Based Dictionary Field	Resulting Subscription

When service item is created			
Login Name	OU Name	Customer	OU
None	None	Customer of Requisition	Home OU of Customer
None	Blank, Inv alid Value, or Zero	Customer of Requisition	NULL
None	Valid OU	Customer of Requisition	OU provided
Login Name	OU Name	Customer	OU
Blank or Invalid Value	None	NULL	Home OU of Customer
Blank, Invalid Value, or Zero	Blank, Invalid Value, or Zero	NULL	NULL
Blank, Invalid Value, or Zero	Valid OU	NULL	OU provided
Valid Customer	None	Customer provided	Home OU of Customer
Valid Customer	No Value	Customer provided	NULL
Valid Customer	Valid OU	Customer provided	OU provided
When service item is updated			
Service Item-Based Dictionary Field	Resulting Subscription		
None	None	No change	No change
None	Blank, Invalid Value, or Zero	No change	NULL
None	Valid OU Name	No change	OU provided
Blank, Invalid Value, or Zero	None	NULL	No change
Blank, Invalid Value, or Zero	Blank, Invalid Value, or Zero	NULL	NULL
Blank, Invalid Value, or Zero	Valid OU Name	NULL	OU provided

When service item is created			
Valid Customer	None	Customer provided	Home OU of Customer
Valid Customer	Blank, Invalid Value, or Zero	Customer provided	NULL
Valid Customer	Valid OU Name	Customer provided	OU provided

Service Item Task Operations

Service item task operations instruct Service Catalog to create, update, or delete a service item. The service item task is executed as specified by its sequence in the service's workflow.

- Creating a Service Item: All custom operations are "Update" in nature. All the points here apply to them as well i.e when a service item task with an operation type of "Create" is executed, Service Catalog:
  - Creates an entry for the service item in the Service Item tables. The entry includes all attributes of the service item for which data has been supplied via the service form.
  - Creates an entry for the service item in the Service Item Subscription table. This table records all service items and their current status.
  - Records the customer, current request, the date and time the request was submitted, and the date and time the service item was created.
  - Creates an entry for the service item in the Service Item History table. This table records the requisition that created the service item.
- Updating a Service Item: When a service item task with an operation type of "Update" is executed, Service Catalog:
  - Updates the existing entry for the service item in the Service Item Knowledge Base tables.
  - Updates the entry for the service item in the Service Item Subscription table to reflect the changed status of the service item.
  - Creates an entry for the service item in the Service Item History table. This table records all operations (and the requisition in which the service item task occurred) that affected the status of a service item.
- Deleting a Service Item: Deleting a service item removes all traces of the service item, erases all references to the item from the system, including its history. In some scenarios, it might be better to include an attribute in the service item to mark it as "Inactive" or "Defunct", and to write conditional rules that prohibit provisioning such items.

For more information on service items and service item tasks, see Chapter 9,Managing the Services and Attributes, on page 247.

Configuring an External Task

This section covers the workflow type for external service item tasks,i.e, the tasks performed outside the Service Catalog.

Defining External Tasks in the Workflow

By default, the drop-down list for Workflow Type contains only one entry, “Internal”, indicating that the task will be performed within the Service Manager module. If integration specialists have used the Service Link module to define “agents” that integrate with external systems, the action associated with each agent also appears in the drop-down list.

For example, the workflow type for the “SAP Integration” agent might be displayed as “Send data to SAP”; the workflow type for an agent that integrates with “Remedy” might look like “Send Ticket to Remedy”.

Tip Contact your site administrator to obtain details about the services and tasks integrated with Service Link.

For external tasks only, an ellipsis (...) appears next to the Workflow Type after you have saved the task.

Figure 13: Ellipsis

If you have permissions to the Service Link module, you can click the “**...**” button to access the Agent Parameter Override dialog box. This dialog box includes settings that show how the data to be sent to the external system is mapped to fields on the service form or other data about the requisition. If you have permission to change these settings, the dialog box is editable, otherwise it is read-only. For more information on permissions for Service Link, see the Capabilities for Service Link section of Cisco Prime Service Catalog Administration and Operations Guide .

External Service Item Task Operations

Once an external service item task becomes active (that is, has a status of Ongoing) in the workflow, Service Link listens for the incoming requests to create, update, or delete service items. The changes made to the service item repository are similar to those of internal service item tasks. The channel ID of the inbound requests allow Service Link to identify the service request to be used for tracking the service item history.

You can perform other task updates that are typical of external tasks—for example, adding comments and sending parameter updates—through the incoming requests as well. When all service item operations are processed successfully, you must be sure to send a “done” action to complete the task so that the next task in the delivery plan can be triggered.

For more information about the Service Item Listener Adapter and the specification of inbound service item messages, see the “Designing Integrations with Service Link Standard Adapters” chapter in the Cisco Prime Service Catalog Adapter Integration Guide is not recommended because the reconfigure action may disrupt processes that are currently running on the machine. Certain actions are, in fact, prohibited if they were to be performed through the VMware Infrastructure Client. Therefore, service designers need to add a “power off” task BEFORE the “reconfigure” task. This ensures that the subsequent request to reconfigure the virtual

machine may proceed. If the virtual machine is already down upon receipt of the power off task, the power off task fails, but this failure is harmless and does not impede the progress of the reconfigure task.

Defining Directory Tasks in the Workflow

Directory tasks can be used to create or update people, organizational units, or Service Link agents using form data. If the operation fails, the task is marked as complete and an error message is written to the service form.

Any free form dictionaries that contain the required fields can be used with the directory task. The dictionary should also include a text field named "**ErrorDescription**" for capturing any errors returned from the operation. Consider hiding this field while ordering, and using the value of this field to conditionally trigger manual tasks for error handling.

Directory tasks can also be used with grid dictionaries to process multiple occurrences of the same object type (e.g., create multiple organizational units). The task loops through all rows in the grid dictionary and performs the operation for each of them. If exceptions are encountered for some of the occurrences, the error messages are logged into the ErrorMessage field for those grid rows and do not affect the successfully completed ones.

You can also design condition expressions using operators such that the application evaluates the condition on each row and proceeds with the operation if the condition is satisfied.

Syntax for condition expression: Data.<DictionaryName>.<FieldName>

Example of a condition expression: (Data.ouDict1.Type = "Service Team" and Data.ouDict1.Description="abc") or (Data.ouDict1.Type = "Business Unit" and Data.ouDict1.Description="xyz")

Although the syntax used here is identical to that for delivery tasks, only dictionary namespaces are supported for the Process Grid Row conditional expressions. Requisition, task or service-specific namespaces are not applicable to this context.

To configure a Directory Task:

Step 1 Choose **Service Designer** > **Plan** > **Task** > **General**.

Step 2 Choose **Directory Task** as the Workflow Type, enter a task name, and save the plan.

Step 3 Click the ellipsis (...) that appears next to the Workflow Type.
The Directory Task Parameter popup window appears.

Step 4 Choose the dictionary to use, and the operation to apply.

Step 5 Enter a **condition** expression, click **Validate**, and click OK.

The following operations are supported for directory tasks. The required and optional fields for directory operations are listed in Table 54: Operations Supported for Directory Task:

Table 54: Operations Supported for Directory Task

Operation	Dictionary Fields	Remarks
• Create a new queue • Update existing queue • Create or Update existing Queue	**Input**: (* Mandatory for Create operation) Name* – Text(100) Email_Address* – Text(100) Home_Organizational_Unit* – Text(200) Timezone* – Text (50) **Output**: Person_ID – Text(50) Organization_ID – Text(50) ErrorDescription – Text(80)	Create a new queue or perform an update if the queue already exists. Organizational units or groups referenced are created automatically if they do not exist in the system. The value for the Timezone field should be in the form of the short name as seen in the Time Zone selections in Organization Designer, for example, “America/Los_Angeles”. The values for Person_ID and Organization_ID fields are updated back to the request form when the queue or Home OU are created.
• Activate a person • Deactivate a person	**Input**:Login_ID – Text(50) **Output**: ErrorDescription – Text(80)	Modify the status of a person to “Active” or “Inactive” accordingly
Create New Organizational Unit	The Dictionary fields supported are: Organization_ID, Name*, Description, Type*, Parent_Name, Parent_Type, Billable, Parent_Description The ID of the created Organization is returned to the form in the Organization_ID field.	If you select this operation in the directory task workflow, you can enable the user to create a new organizational unit as a task during the service workflow. The user does not need access permissions for the Organization Designer Module. While the task is being executed, Parent_name field already exists in the order form where you can add a parent name. Application either creates a new parent if the parent does not exist or associates it with the existing parent. The ID of the created Organization is returned back to the form in the Organization_ID field.

Operation	Dictionary Fields	Remarks
• Create a new person • Update existing Person • Create new person / Update existing person		Organizational units or groups referenced are created automatically if they do not already exist. The values for the Person_ID and Organization_ID fields are returned to the service form when the person or Home OU are created during the operation. The value for the Timezone field should be in the form of the short name as seen in the Time Zone selections in Organization Designer; for example, “America/Los_Angeles”. The input type for the Organizations, Groups and Roles fields should be set to “select (multiple)” in the Active Form Component used. In an Update operation, the Login_ID field is used to look up the person record. Person attributes absent from the dictionary are excluded from the update. If an user’s account is locked due to retry policy violation the IsLocked field is enabled automatically. To unlock the user account, disable the IsLocked field and then reset user password. Enter the LoggedIn_UserPwd only if you are creating a user or changing a user password.

Operation	Dictionary Fields	Remarks
	Input: (* Mandatory for Create operation) First_Name* – Text(50) Last_Name* – Text(50) Login_ID* – Text(50) Password* – Text(50) Email_Address* – Text(100) Home_Organizational_Unit* – Text(200) Timezone* – Text(50) Is_Locked LoggedIn_UserPwd Organizations – Text(4000) Groups – Text(4000) Roles – Personal_Identification – Text(512) Title – Text(100) Social_Security_Number – Text(100) Notes – Text(4000) Company_Code – Text(200) Division – Text(200) Business_Unit – Text(200) Department_Number – Text(200) Cost_Center – Text(200) Management_Level – Text(200) Region – Text(200) Supervisor_ID – Text(200) Employee_Type – Text(200) Location_Code – Text(200) Company_Street_1 – Text(100) Company_Street_2 – Text(100)	

Operation	Dictionary Fields	Remarks
	Company_City – Text(100)	
	Level – Text(100) Cubicle – Text(100) Personal_Street_1 – Text(100) Personal_Street_2 – Text(100) Personal_City – Text(100) Personal_State – Text(100) **Output**: Person_ID – Text(50) Organization_ID – Text(50) ErrorDescription – Text(80) Personal_Postal_Code – Text(100) Personal_Country –Text(100)	External User Authentication (EUA) allows to override default authentication mechanism within Prime Service Catalog and authenticate a user against an external source. If EUA is not enabled or if you are using a backdoor url the authentication for password change in user profile, the LoggedIn_UserPwd fields must contain the database password of the logged in user trying to change the password. If EUA is enabled you must use the LDAP password in the LoggedIn_UserPwd field. An error is returned if: • If a person with the same login name already exists while you create a person or • If the person does not exist while you update a person • If the password does not conform to the password policies.
Update Organizational Unit	Name* Type* Description Status Billable ErrorDescription If you need to modify the type of OU, then add a field named '"Change_Type"'.	**Note** Type field identifies the Organizational Unit that will be acted on. Also all Organizational Unit related operations can define the Organization_ID that will return the ID of the Organizational Unit.
Rename Organizational Unit	Name* New_Name* Type* ErrorDescription	

Operation	Dictionary Fields	Remarks
Activate/Deactivate organizational unit	Name* Activate* Type* ErrorDescription	The Activate field should be set to "true" or "yes" for updating the organizational unit status to “Active”, or to “false” or “no” for updating the status to “Inactive”.
Update existing agent properties	Name*	This feature enables you to order a service with a directory task operation that updates inbound and outbound agent properties. **Note** Ensure that names of agent properties match with names in the free form dictionary. However the character “.” has to be replaced with “_” because the application does not accept “.” character while creating a active form dictionary. During the update process, the application ignores incorrect entries in the dictionary. For example if the agent property is FileOutbboundAdapter.FileLocation the dictionary field name should be FileOutbboundAdapter_FileLocation
Create account / update account	Ensure that the dictionary you associate with the directory task workflow for this operation has the following fields: Name* AccountType* Description BillingRateGroup OrganizationalUnit ErrorDescription	1) The maximum length for Name is 100 characters. 2) AccountType value should be “Project Account “or “Tenant Account”. 3) The maximum length for Description is 500 characters. 4) The Organizational Unit field takes the input of a multi-select box and the values are assumed to be of type “Business Unit”. For tenant accounts, only one organizational unit may be specified. 5) Functional positions and custom attributes cannot be specified at this time.

Operation	Dictionary Fields	Remarks
Create agreement	Name* AccountType* Account* AgreementTemplate* Description ErrorDescription	

Configuring AMQP Tasks for Publishing Service Request to an External System

Prime Service Catalog allows you to publish the service request to another application or system using Advanced Message Queuing Protocol (AMQP). AMQP provides standards on how messages should be structured and transmitted between platforms or systems.

Prime Service Catalog using RabbitMQ, an open source software, implements AMQP and perform message routing between the systems. An example of the systems consuming the service request can be the Cisco Process Orchestrator, which picks up the request and executes a fulfillment workflow.

Note To prevent Poodle security attack, Prime Service Catalog in 11.0 and later, supports connection to RabbitMQ only via TLS 1.2 protocol, hence consumers are also required to use TLS1.2 to connect and consume the messages. Therefore, to connect and consume these messages from RabbitMQ server, use only TLS1.2 .

A task type called Queue Service Request is available under the workflow type, which is used to publish the service request to another application or system.

Queue Service Request consists of a pre-task, main task, and a post-task that are executed in order. The Service Designer provides the option to enable or disable a pre task and post task that will be executed before and after the main AMQP task. A main task alone can be executed without having a pre and post task. For more information on AMQP, see *Cisco Prime Service Catalog Integration Guide*.

Step 1 Choose **Service Designer** > **Plan** > **Task** > **General**.

Step 2 Choose **Queue Service Request** as the **Workflow Type**.

Step 3 Enter the **Task Name**.

Step 4 **Save** the plan.

Step 5 Click the ellipsis (...) that appears next to the Workflow Type.

Step 6 In the Queue Service Request Parameter popup, do the following:

a) For the Primary Task, enter the exchange name in the **Topic** field and select a **Payload Type** from the drop-down list.

b) (Optional) Select the **Execute Pre Task** and **Execute Post Task** check boxes, if you want the Pre Task and Post Task to be executed. Clear these check boxes to skip these tasks.

c) (Optional) Select the **Auto Complete** check box for a task to auto complete a task. When a task is complete the next task will begin automatically.

d) (Optional) Select the transformation from the **Transformation Outbound** drop-down list to change the format of the outgoing messages.
e) (Optional) Select the transformation from the **Transformation Inbound** drop-down list to process the format of the incoming messages.
f) Select the **Connection Identifier** from the drop-down list. This option lists the AMQP connections added in the Administration > Manage Connections > AMQP page. Based on this option, the message format is populated.
g) (Optional) Select the **Public Key** from the drop-down list.
h) (Optional) Select the **Message Format** from the drop-down list for this task. This selection overrides the default message format configured for all the messages in the Administration module, Manage Connections > AMQP. The message format describes format in which the outbound message has to be sent or has to process the inbound message. Transformation type is changed based on the message format selected. For example, for XML transformation type is XSL and for JSON it can be either JOLT or FTL. FTL is supported only for outbound transformation.
i) Click **Save** to save the task definition. This option will create tasks topic and queue on the Rabbit MQ server only when a service is ordered.
Click **Create** to create tasks topic and queue on the Rabbit MQ server at the service definition time.

Note
- The AMQP Public Key created in the **Administration > Settings > Public/Private Keys** will be available for selection for every new AMQP task that is created.
- If the RabbitMQ Cluster is down, the AMQP task will not be completed and the message will not be published to the server. When the RabbitMQ Cluster or one of the nodes in the cluster will be active, then the message will be republished automatically to the server, and the task will be marked as completed.
- The message transformations are created in the Service Link module, **Manage Integrations** > **Transformations** are available for selection in the Transformation Outbound and Transformation Inbound fields. For more information, see section Managing Transformations in the Cisco prime Service Catalog 12.0 integration guide.

Configuring AMQP Tasks for Publishing Request to Cisco Process Orchestrator

Step 1 When a service is ordered in Prime Service Catalog, it generates nsXML or JSON output and publish this output to the AMQP Server.

Step 2 AMQP server secures this nsXML message with Public key. For more information on this, see Encrypting AMQP Tasks.

Step 3 Cisco Process Orchestrator reads this message from AMQP server using the private key. This nsXML is passed into Cisco Process Orchestrator northbound web services, which starts a process at Cisco Process Orchestrator.

Step 4 Cisco Process Orchestrator makes web service calls or connects to Cisco Prime Service Catalog through an adapter to send the acknowledgment of the message received from Cisco Prime Service Catalog.

Note Cisco Process Orchestrator generates either Modulus and Exponent or Public Key along with Private Key.

Encrypting AMQP Tasks

The Service Catalog supports encrypting every AMQP task with a public key selection. The AMQP Public Key is used to encrypt the data in a secure string format.

In the dictionary field, when the 'is-secure' attribute is set to true, the value of those dictionary fields will be converted to secure string using the public key. The format of the secure string is based on the **AMQP Secure String Format** that you select.

Procedure

Step 1 Choose **Administration > Settings**.

Step 2 Select **Public/Private Keys** option from the right hand side tab.

Step 3 Click **Add**.

Step 4 Enter the **Name** of the public key, the encrypted data in the **Modulus** field and the **Exponent** value.
Based on the values specified for Name, Modulus and Exponent, the system generates a GUID that cannot be modified/edited. This GUID is used for adding external layer of security for password and token.

Step 5 Click **Save** and **Close** to exit.

Step 6 Select a **AMQP Public Key** from the list for a particular AMQP task in Service Designer

Step 7 Select a **AMQP Secure String Format** drop-down list. This will encrypt the secure string to the selected format.

Step 8 Click **Update** to save the changes.

Creating Service Delivery Tasks for Granting and Revoking Permissions for a Service Item

You can create a service delivery task to grant or revoke permissions for a service item or multiple service items that are created using grid dictionaries within the service delivery plan. This was possible only via REST call earlier.

The dictionary attributes for grant/revoke permissions must be as follows:

- recipientName
- recipientType
- recipient
- object
- objectName
- scope
- permission
- instanceName
- domain
- ErrorDescription

For more information about granting or revoking permissions for possible field values using APIs, see "Grant and Revoke Permissions API Table" in the Cisco Prime Service Catalog Adapter Integration Guide .

To create a **NsAPI** task to grant SIM permissions:

Step 1 Choose **Service Designer** > **Plan** > **Task** > **General**.

Step 2 Choose **NsAPI** as the **Workflow Type**, enter a task name, and save the plan.

Step 3 Click the ellipsis (...) that appears next to the Workflow Type.

Step 4 Choose the dictionary to use and the **NsAPI Task Type** to apply.

Step 5 Click **OK**.

Creating a Scheduled Start For a Delivery Task

By default, a delivery plan becomes active (and the clock starts ticking on whether the service performers are completing tasks on time) as soon as all authorizations for the service request are completed, or, if there are no authorizations, as soon as the service request is submitted. Sometimes, this is not the desired behavior. For example, you may enter a request on behalf of a new employee scheduled to start work in a few weeks; or a development team may request a new server for a project scheduled to ramp up in the future. For such cases, Service Catalog allows you to create a Scheduled Start task, which does not become ongoing (active) until the specified start date is reached.

You can specify that a scheduled start task is executed on:

- A date and time specified on the service form by the customer or by someone performing an approval or review.
- A fixed date and time that you specify in the service's design, on the Plan tab in Service Designer

To create a scheduled start task by using a field on the service form:

Step 1 In the service definition, go to the **Form** tab for active form components used in the service, and identify the dictionary field that will hold the data (for example, NewHire.StartDate). The field's data type can be either Date or Date and Time.

Step 2 If required, go to the Active Form Component that contains the dictionary field identified in the previous step. Click the Access Control and Display Properties tabs, and set the appropriate dictionary permissions and the HTML representation for the field to be used.

Step 3 Click the **Plan** tab for the service, and then identify or create the task that you want to delay.

Step 4 On the General subtab for the task, check **Allow a scheduled start date**.

Step 5 In the "Form data for start date" field, enter the name of the dictionary field you want to use to hold the date and time of the starting-point for the task. This is the same field you identified in Step 1. Use the following syntax:

- Data.*DictionaryName.FieldName* for a field from a dictionary in one of the service's form components.
- ParentData.*DictionaryName.FieldName* for a field from one of the parent service's dictionaries (in the case of a bundled service).

System Behavior for Scheduled Start Tasks

The system maintains both a Scheduled Start Date and an Actual Start Date for each delivery-plan task. You can see these dates on a task form in Service Manager.

The system does not automatically start a scheduled start task after the preceding task is completed. The scheduled start task has a *Scheduled* status until the specified starting-point is reached. At that point, the system changes the task status to *Ongoing* , and the task is then available for processing by the assigned performer. The “Scheduled Start” and “Started on” fields are the same for a scheduled start task.

If the starting-point for a scheduled start task is defined from form data, and you use a Date field to define this point, the system sets the time to the working hours of the task’s performer. For example, if the task is to be performed by the HR Group queue and that queue’s working hours are from 8:00 to 16:00, the system sets the time to 8:00 on the date entered by the user.

The system schedules all delivery-plan tasks after the Service Delivery phase of the workflow process begins. (The Service Delivery phase begins after the Authorization phase completes.)

The system does not evaluate the starting-point specified for a scheduled start task until it is scheduling all the delivery-plan tasks at the start of the Service Delivery phase. You may want to take advantage of this when designing services, as it means that the starting-point may also be specified by the performer of an authorization or review step (as opposed to the customer who orders the service).

As a service designer, you cannot control what the customer enters as the starting-point for the scheduled start task. For example, the customer might enter a future date that ends up occurring before preceding tasks in the service are completed. If the start date specified by the customer for a scheduled start task is earlier than the earliest possible starting date (with respect to the rest of the plan), the system ignores the specified start date and treats the task as if it were not a delayed task. This is logged in the System History with a comment indicating that the start date is ignored.

- Also if the start date specified is not a mandatory field on the service form and the customer/approver does not specify a start date, Service Catalog treats the task as if it were not a delayed task.

Defining Task Participants

Step 1 Choose **Service Designer** > **Services** > **Plan** > **Tasks** > **Participants**.

Step 2 Specify who completes the task and, optionally, how the work is supervised. For more information, see Table 55: Fields to Define Task Participants.

Step 3 Click **Save** to save your changes on this tab.

Table 55: Fields to Define Task Participants

Field	Description
Role	Task Performer Role: The task performer is the person, queue, or position to whom the task is assigned. Task Supervisor Role: The task supervisor provides a measure of control over task performance. When combined with the Administration setting "Allow Task Supervisor to cancel task", it allows the designated person, queue, or functional position to cancel (skip) the task
Name	Name of the Performer Role, meaning the entity who will perform the task. This name appears in the "By" column in the task list in the Plan tab. **Or/And** Name of the person in the supervisor role, meaning the supervisor of the person who will perform the task/deliver the service.
Assign	Choose how the performer/supervisor of the task is assigned: • **From a position:** The person currently filling a functional position defined in Organization Designer. • **A person/queue** : People or queues as defined in Organization Designer. • **From an expression** : A conditional expression shown in the " Assign to" field. • **None.**

Field	Description
Assign to	**Assigning Roles Based on an Expression** As with authorizations, the service team or person assigned to tasks in the delivery plan may depend on data on each requisition, such as a customer's location. Expressions can be used to intelligently route tasks, rather than creating different forms or workflows for each possible scenario. Performer, Supervisor, and Project Manager roles can all be dynamically assigned. • If you choose "From a position" or "A person/queue", click the "**...** " button to bring up a select dialog box to choose who the task is assigned to. Click **Add** or **OK** to save your selection. • If you choose "From an expression", click the "**...**" button to bring up the Edit Expression dialog box to enter the expression to be evaluated for assigning the task. Click **Set Expression** to save your selection.

Defining Email Notifications for Delivery Tasks

Several events are associated with each task. By associating an email with each event, you can notify the service's customer, task performer, or other groups or individuals of the current status of the request.

Choose Service Designer > Services > Plan > Tasks > Email subtab to specify which notifications should be sent in response to which event. The notification must be defined using the Notifications component of the Administration module. Although Service Catalog is shipped with some default email templates, most sites prefer to design custom notifications.

For each task in the delivery plan (and for each authorization or review), the designer can choose the number of tiers in the escalation structure to be used by that task.

For example, you might configure three escalation tiers:

- Tier 1: 1 hour after the task becomes late
- Tier 2: 8 hours after Tier 1
- Tier 3: 16 hours after Tier 2

The Maximum Tier setting determines how many of the escalation tiers are to be used by that particular task, starting with the first tier.

- As much as there are in escalations = all tiers
- Specified As = the number of tiers to use for this task; any number less than or equal to the number of tiers defined.

In this case, for example, if the maximum tier is Specified As 2, notifications would be sent 1 hour and 9 hours after the task becomes late; the third tier escalation would not be applied.

Adding Task Instructions

Choose **Service Designer** > **Services** > **Plan** > **Tasks** > **Task Instructions** subtab, to give instructions for the task or set the link to any task instruction-related URLs.

Table 56: Fields in Task Instruction Subtab

Field	Description
URL	Enter the full URL beginning with “http://” you wish to link.
URL Description	Enter a brief description to explain what the user will see when they link to the URL, or to include HTML in the checklist item. For example: Click Here for more information on how to complete your task.
Task Instructions	Use this text field to enter task instructions, if desired.

The HTML editor can be used to introduce HTML into the URL Description and Task Instructions fields.

Creating Checklist For Completing a Task

You can create a list of reminders or detail the specific procedural steps for completing a task using checklists. These appear in Service Manager as a checklist, detailing procedural steps for completing the work. Checklists do not affect due dates and are not reportable.

Step 1 Choose **Service Designer** > **Services** > **Plan** > **Tasks** > **Checklists**.

Step 2 Click **Add New** to add new items to the list by entering the checklist item in the Item Properties box to the right.

Step 3 Use the Table 57: Fields in Checklist tab table to further configure checklists.

Table 57: Fields in Checklist tab

Field	Description
Items	The items in this list comprise the checklist the performer sees in Service Manager. Use the **Up** or **Down** Arrows to rearrange the item order. Items at the top should be performed first.
Name	Enter the name for the checklist item.
Make checklist items mandatory for task completion	Check this option to make all the items mandatory for completion of the task. This means the person performing the task will not be able to close out the work for the task until they have checked all of the items in the checklist in Service Manager.

Configuring Escalation Email for Delayed Tasks

You can use the **Plan > Escalations** subtab to specify the email notifications sent to performers, supervisors, and customers when an activity is late.

To define escalation tiers:

Step 1 Choose **Service Designer** > **Plan** > **Escalations**.

Step 2 Click **Add**.

Step 3 Select **After (hours)** checkbox to define the number of hours after the activity is triggered the application is required to send an escalation email.

Step 4 Enter the email address of the recipients

Step 5 Choose the activity from the drop down list.

Step 6 Perform the above steps to configure escalations for sequential activities

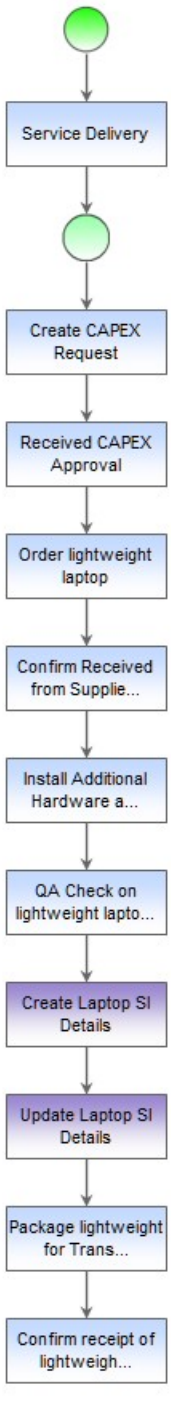
Designing Service Delivery Workflows

Service Catalog offers two approaches to design the workflow of tasks that comprise the delivery plan:

- Use the Task area at the middle of **Plan** tab to enter tasks, in conjunction with the buttons available (Indent, Outdent, Up, Down) to configure the workflow so that tasks execute in the appropriate order and are grouped correctly.
- Use the Graphical Workflow Designer, available from the **Graphical Designer** tab, to draw the workflow by dragging and dropping tasks, connectors, and subtask groupings onto the drawing area.

The diagram to specifies the tasks and subtasks and their relationship. It comprises the delivery plan, the sequence in which they are executed, and whether execution is sequential or concurrent.

A workflow can be configured using either tool, and, in fact, service designers can switch back and forth between Graphical Designer and the dialog boxes provided on the Task tab. The Graphical Designer provides the workflow for the tasks, and its property sheet allows users to enter most general information about the task as well as the performer's role. Other individual task details, such as emails associated with task fulfillment, checklists, and task participants must be supplied by using the subtabs of the Plan tab.



Task
Service Delivery
Create CAPEX Request
Received CAPEX Approval
Order lightweight laptop
Confirm Received from Supplier
Install Additional Hardware and Software
QA Check on lightweight laptop
Create Laptop SI Details
Update Laptop SI Details
Package lightweight for Transport and Hand off to driver
Confirm receipt of lightweight laptop

.

About Graphical Workflow Designer

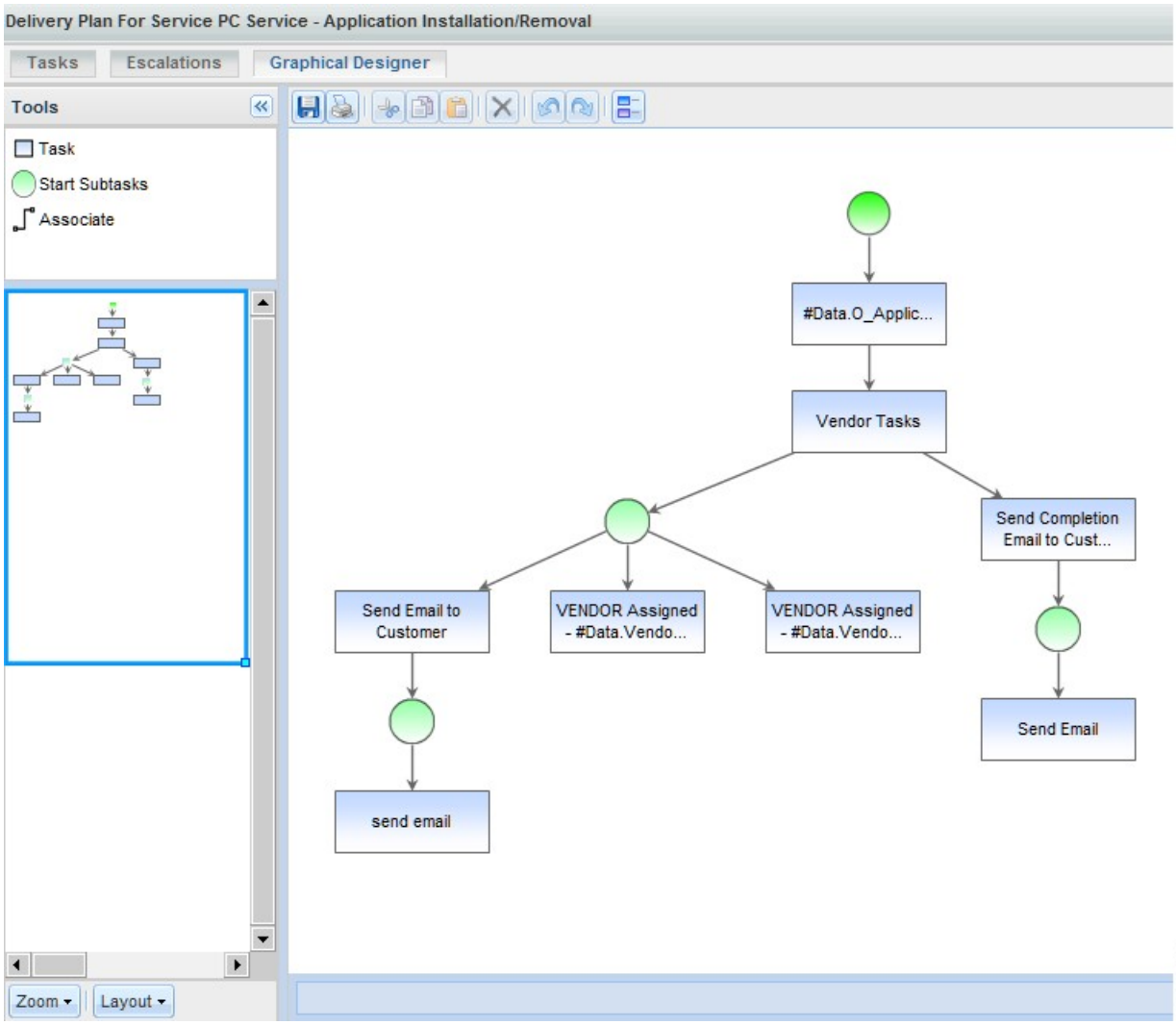
The Graphical Workflow Designer allows service designers to drag and drop tasks, name them, and connect them. The Graphical Designer also provides the ability to define parent/child relationships among tasks. An accompanying Task Properties sheet allows specification of additional details about the task definition and execution.

The Graphical Designer provides a visually oriented alternative to designing the service's delivery plan by using the dialog boxes provided in the General tab for the delivery plan. Designers can define tasks and specify the workflow between them using either method, and freely switch back and forth between the Graphical Designer and the dialog boxes.

The Graphical Designer provides a simplified view of the delivery plan. To configure the more advanced details of a plan (such as those available on the Participants, email, Task Instructions, and Checklist sub tabs), switch to the Plan tab. For more information see, Configuring Delivery Tasks, on page 176.

The Graphical Designer pane has the following sections:

Figure 14: Graphical Designer Pane



1	Tools Pane	4	Outline Pane
2	Toolbar	5	View Tools
3	Work Area		

- **Tools Pane**, at the top left, contains building blocks of the workflow designer. These building blocks can be dragged and dropped into the work area and connected to create a complete workflow.
- **Toolbar** contains the buttons for saving the delivery plan; printing the plan; and manipulating the contents of the work area.

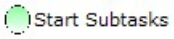
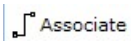
Note Legend lists the color coding used to represent objects included in a workflow diagram

- **Work Area** holds the workflow diagram. When the Designer is invoked for a new delivery plan, the work area is blank except for the bright green circle, which denotes the start of the plan.
- **Outline Pane**, in the middle of the left side of the page, gives a high-level overview of the diagram.
- **View Tools: Zoom/Layout** buttons, at the bottom left, allow the designer to magnify or reduce the size of the workflow diagram or to alter the diagram's orientation.:

About Tools Pane

The Tools Pane includes tools for adding the basic types of objects that comprise a workflow to the diagram.

Table 58: Tools Pane

Option	Explanation
	A workflow consists of a series of tasks, arranged in sequence. Use the Task tool to add a task to the workflow.
	Tasks can be grouped as child tasks to a parent task.
	Use the Associate tool to specify the sequence of tasks and to associate parent with child tasks.

In addition to the options in the Tools Pane, context-sensitive task cursors are available to help manipulate the content of the diagram. As you move the mouse over tasks previously placed on the diagram, the cursor changes shape, to indicate which actions are available:.

Table 59: Cursor

Cursor	Explanation
	Once an object has been selected, the select-cursor appears. Any objects marked with the select cursor are subject to the next Copy, Move, or Delete command.
	The connect cursor allows you to associate the highlighted task with another task. With the connect cursor displayed, drag the mouse to the task to be connected to the current task. When you release the mouse, a task sequence (association) is established.

Creating a Delivery Plan using Graphical Designer

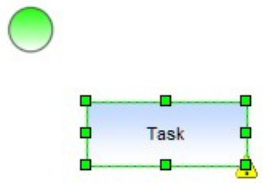
To create a delivery plan using the Graphical Designer, edit the service in Service Designer, click the **Plan** tab, then click the **Graphical Workflow** subtab. The work area appears empty except for the "Start Workflow" icon.

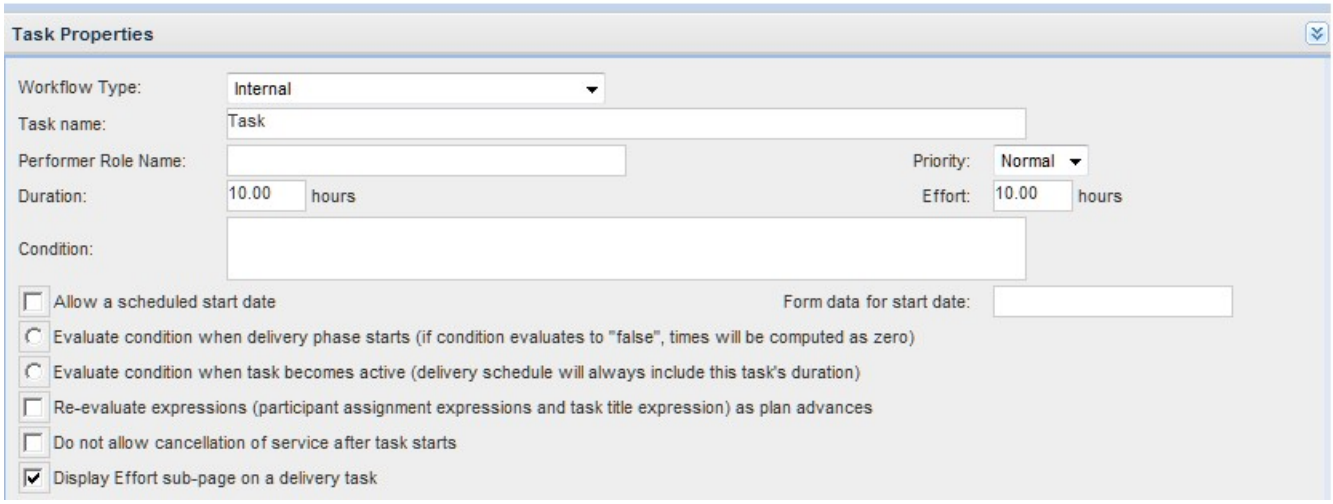
Figure 15: Start Workflow

Step 1 Define a task

a) The first step would typically be to place a task on the workflow. Click the **Task** tool (from the Tools panel) and drag it onto the work area. A task appears. Double-click the task to display the Task Properties dialog box at the bottom of the work area.

Figure 16: Task Properties – Yellow Alert Icon





The Task Properties window includes all data shown in the General subtab for the task as well as the Performer Role Name, which is especially useful in design sessions. Detailed explanations for the fields here are given in the Defining Service Item Task in a Delivery Plan, on page 188.

b) Start defining the workflow for the task here, but will need to use the Plan tab subtabs for each task, to complete the task definition.
Critical aspects of defining the task are to assign a Task Name and choosing the Workflow Type. You can create a new “Internal” task (a task to be performed wholly within Service Manager); designate that a previously defined Service task (external task specified via a Service Link agent); or configure a Service Item Task (task to create, update, or delete a service item specified via Service Item Manager) to be included in the workflow.

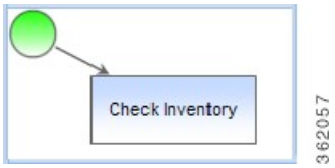
c) As soon as you move from the Task Properties dialog box back to the diagram, the task name is displayed and the Task icon in the diagram may change, to reflect the workflow type. You may return to the Task Properties dialog box at any time by double-clicking the task.

Step 2 Specify workflow

a) The yellow alert icon warns you that the diagram in its current state is not valid (Figure 16: Task Properties – Yellow Alert Icon). You can hover over the alert icon to get a description of the problem, but in Figure 16: Task Properties – Yellow Alert Icon it is pretty obvious—the task needs to be associated with the "Start Workflow" icon, and assigned to its appropriate sequence within the workflow.

b) To place the task within the workflow, click the **Associate** tool. Then click the initial item in the workflow (in this case, the "Start Workflow" icon) and drag the mouse to the subsequent item (in this case, the first and only task on the diagram). When you release the mouse, an arrow is added to the diagram, showing the flow from "Start Workflow" to the first task in the workflow.

Figure 17: Start Workflow

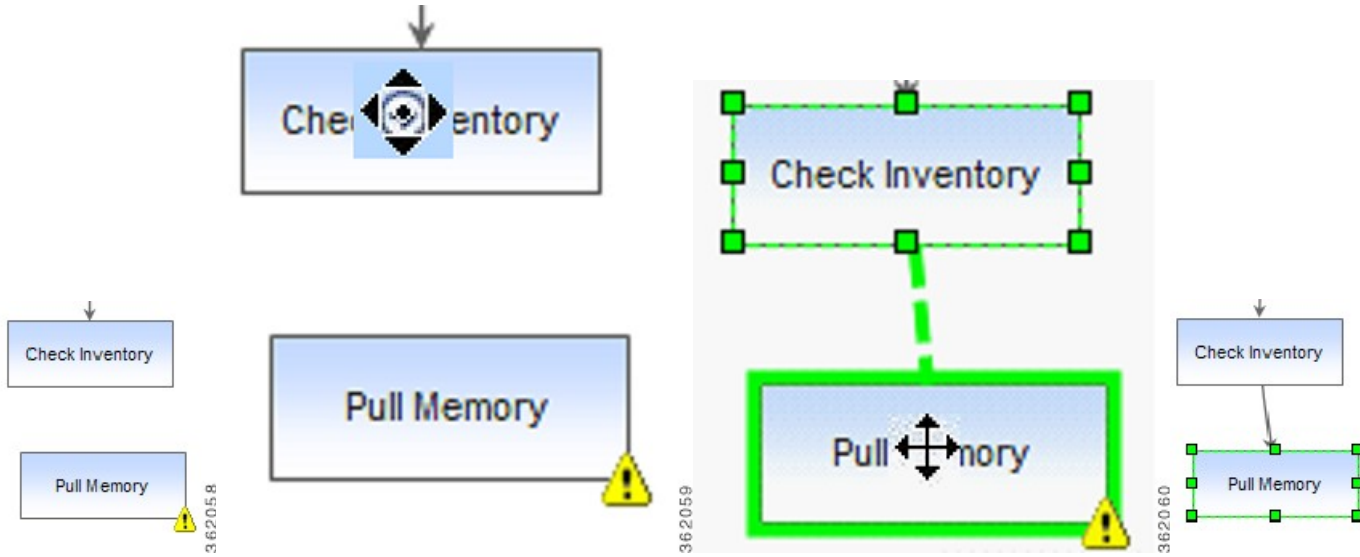


c) A more convenient method for connecting tasks is to use the Connect Cursor:

- Add another task to the diagram and, if desired, give it a name, for example, "Pull Memory". This task is next in the workflow, after "Check Inventory".
- To associate the tasks in this sequence, move the cursor to the middle of the first task (Check Inventory) until the connect cursor appears.
- Drag the mouse to the second task. A thick dotted line appears between the tasks and the target task is highlighted in a thick solid line. The select cursor appears in the second task.

◦ Release the mouse. An association is drawn and the diagram is valid.

Figure 18: Flow Diagram



Step 3 Create Parent Tasks and Subtasks
Tasks may be grouped for several reasons. For example:

Some of the tasks are conditional, and you want to send an email notification when the first or last task in the series completes, regardless of which specific task is executed.

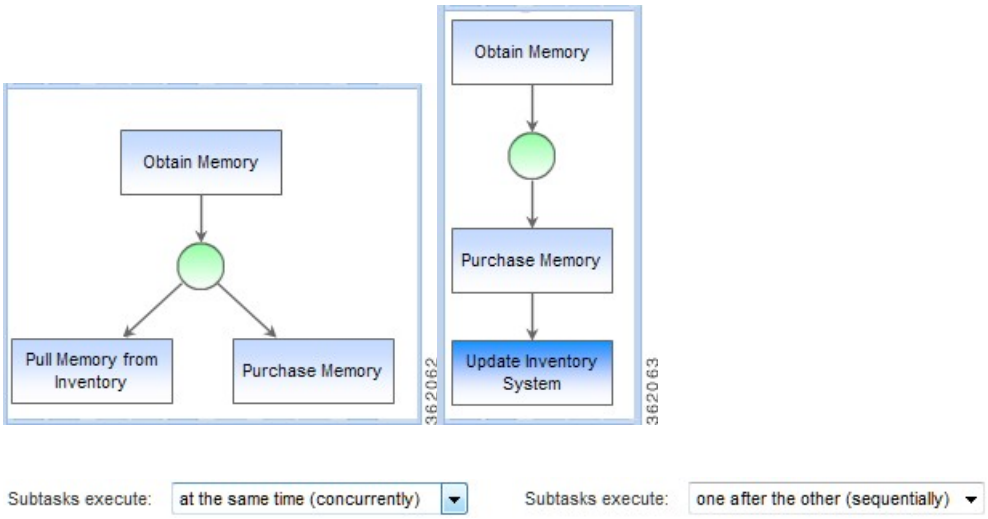
A task should be performed concurrently with a second task. In computing the Service Level Agreement (SLA) for the service, you do not want to count both tasks, just the one that would take the longest.

To account for these and other scenarios, Service Catalog supports grouping tasks. A parent task may have one or more children or subtasks. The subtasks may execute either consecutively or concurrently.

a) Drag the Start Subtasks icon onto the work area in the appropriate sequence.
b) Drag task icons onto the work area underneath the Start Subtask icon. Define the tasks as appropriate.
c) To indicate that the tasks execute concurrently, move the cursor to the center of the Start Subtasks icon, until the connect cursor (with the cursor handles) appears, then drag the mouse to each subtask. Each are connected to the Start Subtasks, as shown in the left diagram, below.

d) To indicate that the tasks execute consecutively, use the connect cursor to connect the Start Subtasks icon to the first task. Then connect the first subtask to the next. The resultant diagram will look like the example on the right, below.

Figure 19: Task Diagram



Step 4 Save a Workflow

You can only save valid workflows. If any alert icons appear on the diagram, it indicates the workflow is not valid. You can hover the mouse over the alert icons to get a detailed description of the problem; it will usually be that the elements of the diagram are not properly connected to each other. Once the error has been corrected, click **Save** again to save the workflow.

Whether you use the Graphical Designer or the Plan tab dialog boxes, Service Catalog saves the workflow exactly the same way. This approach is good, in that you can use whichever tool is most convenient for you. But Service Catalog does not save the diagram itself—it reconstructs it from the delivery plan previously saved, and then adds to that delivery plan as you continue to work on the diagram. You can customize the appearance of the workflow for printing, for example, by moving tasks or increasing the size of a task icon, to show a longer description. However, any such changes are removed when the diagram is saved, and the workflow reverts to its default layout (horizontal or vertical).

Step 5 Print the Diagram

Click the **Print** option from the toolbar to print the diagram. The printer-friendly version will reflect the current view of the diagram, including its scale and orientation. If required, the diagram can span multiple pages. Use the standard browser **File > Print** command to print the diagram.

It is possible to make manual changes to the diagram, for example, to change the size of an individual task icon. If such changes have been made, they are reflected in the printer-friendly version. However, all such changes are temporary, and are not saved when the workflow is saved.

Graphical Designer vs. Plan Tab

There are certain things the Graphical Designer does more efficiently than using the dialog boxes in the Plan tab. For example, it is much easier to change the sequence of tasks, simply by reconnecting one task to another, in the Designer than it is by moving rows of data up and down.

The Graphical Designer includes an option to print the diagram. You will need to compare that printed version to the service preview available in Catalog Deployer, which includes the detailed workflow in textual format, to see which better suits your needs. Perhaps a user presentation will call for the printed diagram, while developer documentation is better served with the Catalog Deployer preview.

To specify the general task properties, designers can use either the Plan tab or the Graphical Designer. Most of the same properties appear on both the Task Properties sheet in the Graphical Designer and the General subtab for the task. However, the General subtab dialog boxes have the following advantages:

- Although you can enter a conditional expression and form data for scheduled start date in both places, only the General subtab dialog boxes include the option to Validate the expression or namespace entered. Validating at design time is generally preferable to seeing the error for the first time when you test the form.
- The way you connect the subtasks to the Start Subtask icon determines whether the parent task is defined so that “subtasks execute sequentially” or “subtasks execute concurrently”. It may be easier to simply change the “Top Level Tasks Execute” attribute of the parent task definition in the General subtab than having to reconnect all the subtasks in the diagram.

You will always use the Plan tab dialog boxes to complete the definition of the delivery plan, to specify the participants, task instructions, email notifications, and checklists for tasks as required.

Tasks are typically not assigned to a particular person, but rather to a queue or functional position. In this way, multiple people are available to work on the task, and it will not be delayed if one person is unavailable.

Configuring Authorizations

The Authorizations tab is used to configure authorization, review, and escalation tasks for the service and to specify the order in which these occur. You can also customize the escalation tiers, recipients, and email messages for late authorization/review tasks for a service.

For each service you can choose from three options:

- **Use service group authorization structure only**: The service inherits the authorization structure defined for the service group (specified in **Administration > Authorizations**); no additional configuration is expected. See the Site Administration chapter of the Cisco Prime Service Catalog Administration and Operations Guide for more information.
- **Use service-level authorization structure only (will not use service group-level)**: Any authorizations defined for the service group are ignored. Only authorizations configured for the service are applied.
- **Use both service group-level and service -level authorization structures**: Accepts the authorization structure defined for the service group, and allows you to supplement it with a customized scheme for each service.

This section covers the following:

- Authorization Types, on page 219

• Using the Site Authorization Scheme, on page 222
• Configuring Authorizations for use with Service Link, on page 222

Authorization Types

The Authorizations and Reviews subtabs are where you determine who should authorize or review the service. You can set up unique roles and define the order in which these roles review and approve request.

- **Authorizations** give the approver the opportunity to determine if the person requesting the service is eligible to receive it. If an authorization is rejected, the process stops and the service is not delivered. If multiple authorizations are defined, they are performed sequentially in the order in which they are specified—one authorization cannot be started until the previous is approved.
- **Reviews** are for information only. A reviewer cannot reject a service or cancel the delivery process, only indicate that he/she has reviewed the service. If multiple reviews are specified, they are performed concurrently. However, the delivery process will not proceed until all reviews have been completed.

Note Based on the authorization type you choose, either the Authorizations – Sequential Process or Reviews – Concurrent Process subtab appears.

- The Up and Down Arrow buttons to the right of each role allow you to move the role up or down in the approval process.

Configuring authorization, reviews, escalations tasks

Step 1 Choose **Service Designer** > **Services** > **Authorization**.

Step 2 Choose an **Authorization Structure and Authorization Type**.

Step 3 Choose **Review-Concurrent Process** to define the review process. Select the check box to the left of the Name field to the authorization role. To add a review or escalation, choose appropriate review- Concurrent Process or escalation-Concurrent process appropriately and click **Add**.
To Update a review or escalation, choose a previously defined authorization/review role by checking the check box to the left of the Name field.

Step 4 Add or update the details using Fields in Reviews Table.

Step 5 Click **Save** or **Update** as appropriate.

Table 60: Fields in Reviews Table

Field	Definition
Name	Name given to the authorization role. This role name does not appear anywhere else in the application.
Subject	The title for the authorization/review task that this role performs. The entry here appears in the My Authorizations portlet that authorizers and reviewers see in My Services.
Duration	The published amount of time (in hours) to perform the review or authorization. This field exists for the purpose of letting you add some time to the amount of time that the authorization really takes. For example, you estimate that an authorization in the service group that you are configuring takes 0.5 hours. However, you want to add some time to the "official" estimate displayed in My Services. Therefore, you enter 1.0 hours in this field, and 0.5 hours in the Effort field.
Effort	The actual amount of time expected to be expended on this authorization.
Assign	Options to assign the default reviewer/approver from: • From a position – A person in a functional position. • A person/queue – A specific person or queue. • From an expression – A conditional expression shown in the Assign to field.
Assign to	Specific person, queue, title of functional position, or conditional expression, depending on what you chose in the "Assign" field. Use the "..." button to choose the specific person or functional position, or enter an expression directly into this field.

Field	Definition
Workflow Type	The default setting for the Workflow Type drop-down is “internal,” which means that the task is performed manually by a user. To set the task so that it is performed as an external task in a third party system, use the drop-down list to choose the appropriate action from this list (see the Configuring Authorizations for use with Service Link, on page 222.)
Escalation Tiers (option button)	Choice for tiers on the Escalations subtab (see the Configuring Escalation Email for Delayed Tasks, on page 207): • Use all Indicates that all escalations set up on the Escalations subtab are to be used. • Use only (where a number (X) is entered from 0 to 99) Indicates that X tiers of escalations set up on the Escalations subtab are to be used. For example, if you have set up the Escalations tab with three tiers of escalations and choose the first option, all three tiers are implemented if the activity is later. If you choose the second option, specifying the number 1, only the first tier of three is implemented if the activity is late.
Condition	Enter any expressions containing conditions that need to be met to fulfill the review. For example, if you enter the following: ActualCost<=1000, then anything that has an actual cost of less than or equal to $1000 is automatically approved by the chosen authorization role. See Syntax for Conditional Statements, on page 335 for more details on creating conditional expressions. Click **Validate** to make sure that your expression is correct. Service Designer indicates if the syntax of the expression is correct or has errors. Note that this is a syntactical check only; the validate function does not check to see if the data you have referenced is actually in the system database.
Evaluate condition when authorization phase starts (radio button)	The condition set up in the Condition field becomes active as soon as the review phase starts.

Field	Definition
Evaluate condition when task becomes active (radio button)	The condition set up in the Condition field becomes active after the review phase completes and the activity phase begins.
Re-evaluate expressions as authorizations/reviews proceed (check box)	Configure the system to re-evaluate expressions for the assignment of participants and title of task following each authorization/review task. This is useful if the participant or task title expressions change after initiating the task.
Notify when authorization/review starts	The email template that is automatically sent out when the review process begins; or chooses “none”.
Notify when authorization/review completes	The email template that is automatically sent out when review is completed; or chooses “none”.
Notify when activity is cancelled	The email template that is automatically sent out when the requester cancels the activity; or chooses “none”.
Notify when activity is rejected (authorization only)	The email template that is automatically sent out when the authorizer rejects the activity; or chooses “none”.
Notify when task is rescheduled	The email template that is automatically sent out when the task is rescheduled; or chooses “none”.
Notify when task is reassigned	The email template that is automatically sent out when a task is reassigned.
Notify when external tasks fail	The email template that is automatically sent out when external tasks fail.

Using the Site Authorization Scheme

You can use the site authorization scheme as set up in the Authorizations tab of the Administration module. This is the default setting, and often is the easiest to implement. See the Site Administration chapter of the Cisco Prime Service Catalog Administration and Operations Guide for more information.

If you see the message “not currently enabled via the Administration module,” then the particular service authorization/review is not enabled for your site. This can be changed in the Administration module.

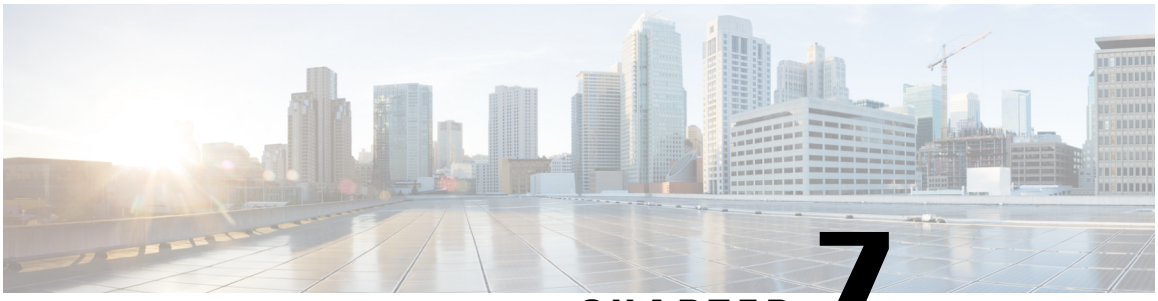
Configuring Authorizations for use with Service Link

If your business process requires authorizations to be processed through a system other than this product, you can use Service Link to have an authorization task performed by that external system. The settings required to define an external task and how it should be handled appear on the Authorization tab—in the Details screen of a role.

1. The default setting for the Workflow Type drop-down menu is “internal,” which means that the task is performed manually by a user. To set the task so that it is performed as an external task in a third party system, use the drop-down list to choose the appropriate action from this list.
2. If for any reason the third party application does not successfully complete the authorization task, you might want to notify an administrator that a problem occurred in the Service Link integration for this task. In the “Notify when external tasks fail” menu, you can choose a template that will send such a notification. For information about email notifications, see Defining Email Notifications for Delivery Tasks, on page 205.
3. If you have appropriate permissions to the Service Link module, you will see an ellipsis (...) button to the right of the Workflow Type drop-down menu. When you click this button, the Task Data Mapping dialog box appears.

This dialog box includes settings that show how the data to be sent to the external system is mapped to data on the service order form, or elsewhere in the system. If you have permission to change these settings, the dialog box is editable. Otherwise, it is read-only.

Now you are ready to set up escalation emails. See the Configuring Escalation Email for Delayed Tasks, on page 207 for details.



CHAPTER 7

Configuring Rates and Accounts For Billing

This chapter contains the following topics:

Configuring Rates and Accounts For Billing

Introduction

You can design billing rates and also configure accounts and agreements to charge services based on usage. Using Demand Management module in Prime Service Catalog you can manage the consumption and billing/charge back of service items for your customers.

You must be granted the read permissions to Accounts, Agreements, and Billing Rates portlets in Portal Designer module to be able to access these portlets. You can also grant permissions in the Organization Designer module using the portlet, portal page, and portal page group. For more information see, Assigning Permissions, section in Cisco Prime Service Catalog Administration and Operations Guide

The key concepts involved in demand management are:

- Billing Rates - These are the rates or rate plans used for pricing service item consumption, characterized by the type of operation involved. Billing rates are equally applicable to the scenario of internal IT charge back. They are used synonymously as charge back rates in the context of demand management.
- Quotas - Quotas govern the maximum amount of service item resources allowed to be consumed by the customers. They are specified in the form of total item count, or the sum of the values for a specific service item attribute. Quotas are enforced by the corresponding policies defined in service items.
- Accounts - Each account represents a logical or physical customer whom the service provider transacts with. Each account may cover a business unit or multiple business units that are grouped together for the purpose of billing/charge back and quota management.
- Agreements - An agreement covers the types and quotas of service items an account may consume.

Configuring a Billable Rates

Billing data on service item consumption are recorded every time a billable event occurs. The rates for billing the service item consumption normally depend on the operation type and/or the quantity consumed.

Before You Begin:

During the billing rate design, the following factors need to be considered:

- What state changes or actions on a service item constitute a billing event?
- What operations need to be defined to model these billing events?
- How are the billing rates determined? Do they change based on the amount of service item resource consumed or are these flat rates? Are there different pricing scheme for different types of customers?
- How is consumption measured? Which attributes of the service item capture the amount consumed?
- What other information needs to be captured along with the billing events?

Step 1 Create billable events as operations in the Define Service Item page in Service Item Manager. Only operations that have the “Billable” check box enabled are eligible for billing rate configuration.

Step 2 Define multiple billing rate groups to segregate the different pricing schemes for the same billable operation over different fiscal periods and/or different accounts.

Step 3 Create one or more billing rate tables for the billable operations once billing rate groups are in place. Create the corresponding rate tables using the Billing Rates portlet.

Step 4 On the Billing Rate Definition tab, specify the service item attributes to be used to look up the billing rates, as well as the ones to be captured along with the rates as memo information.

Step 5 On the Billing Rate Table tab, enter the rates and/or rate codes that correspond to the unique combinations of lookup attribute values.

Note A billing rate group is associated with one or multiple accounts when the accounts are set up. The billing rate group associations may need to be updated at the turn of fiscal periods if there are billing rate changes between the periods.

Step 6 When a billable operation of a service item is invoked, the billing rate for the service item is derived based on the values of the matching billing attributes as defined in the billing rate table.

Step 7 The billing information will be collated and recorded in a billing history table. The billing history can be accessed in the Billing History Portlet. It can also be retrieved through REST web services for passing to a third-party billing engine for further processing.

Creating a Billing Group For Similar Rates

The operations performed for each service item may vary and therefore the rate at which each operation can be billed also varies. Create a billing rate group to associate a rate table logically and also group similar rate tables. You can create multiple rate groups.

Procedure

Step 1 Choose **Demand Management** > **Billing Rates**.

Step 2 Click the + **Add** in the list panel to the left and select **Rate Group**.

Step 3 Enter a **Name** and a **Description** of the rate group in the Add New Billing Rate Group window.

Step 4 Click **Add**.

Step 5 Click **Save** in the Billing Rate Group Definition page to save the rate group that you created.
The rate group appears in the list panel to the left.

Configuring Rates for a Service Item

You can create a billing rate table to define billing attributes and attributes that could be used for reference. You can also use the billing rate table to configure rates for a service item.

Note Ensure that you have additional permissions to manage billing rates in the Organization Designer module.

The content panel for billing rate table contains two tabs.

Billing Rate Definition- You can define the billing attributes and operations of a service item task that can be used for billing.

Billing Rate Table-You can configure the rates for the service item.

You can create multiple rate tables for a service item operation. When the service item operation is initiated, the system refers to the billing rate table and inserts an entry into the billing history table with the rate code, rate and the memo attributes.

To create a billing rate table:

Step 1 Choose **Demand Management** > **Billing Rates**.

Step 2 Click the + **Add** in the list panel to the left and select Rate Table.

Step 3 Enter the required information in the Add New Billing Rate Table window. Ensure that:

- **Name-** The name has alphanumeric characters and begins with an alphabetic character.
- **Service Item Type-**The service item type that you want to associate with the rate table. The attributes that were already defined for the service item, are used to configure rates in the Billing Rate Table tab.
- **Rate Group-** Choose the rate group from the drop down list. The rate group will be associated with the rate table.
- **Unit Rate-** Select the check box if you want the price of the service to be calculated using unit rate. The price of a service will be calculated by multiplying the unit rate with the corresponding attribute value in the service form. The calculated rate will be saved in the billing history table if the unit rate flag is set. Unit rates are also considered during export and import operations in catalog deployer. For more information about exporting/importing billing rates and options considered while export/import, see 'Managing Content Deployment' chapter in Cisco Prime Service Catalog Administration and Operations Guide

For example, if you are ordering a hard disk of unit rate 10 dollars and the quantity ordered is 2, the billable amount is 10*2 that is 20 dollars. The quantity entered in service form, ie 20 dollars is saved in the billing rate table and billing history.

The per- unit rate billing does not support more complex billing rate calculations such as step rates and volume discounts. Therefore when you select Unit Rate ensure that:

- ◦ Only one billing rate attribute is billable.
  ◦ Only one data record exist in the Rate Table data.
  ◦ The single billable attribute must be of 'Number' data type only (Integer/Long/Double/Money).
  ◦ If there are multiple records in data tab, do not select Unit Rate flag.

Step 4 Click **Add**. The content panel for billing rates contains 2 tabs:

- **Billing Rate Definition -** displays the attributes that have already been defined in Step 3. This tab also contains the following tables:

  1 In Billing Rate Attributes table, select the attributes that you want to consider for billing in the **Billing** check box and the other attributes that you want to be saved as a reference in the Lookup check box.

The information in the Billing Rate Attributes table is saved in the BiBillingHistory table. The BiBillingHistory table can be used by the billing engine to compute a rate for the service item task.

1 In Billing Rate Operations table, check the **Apply?** check box for the service item operations that you want the billing rate table to be associated with.

Billing Rate Definition tab displays the billable operations that you defined when you created a service item operation. See Configuring Service Item Definition.

- **Billing Rate Table -** Click **Add**, to configure the rates for the service item operation. An empty line opens at the bottom of the grid, where you can enter the appropriate rate information for the attributes. You can configure multiple billing rate codes.

For example, a service item like sandisk could be applicable for multiple companies such as IBM and Cisco. You can create rate code based on the company name that is using the sandisk.

Rates could also vary depending on the unit of measure. You can define multiple rate codes depending on billing rates for the usage of sandisk. You could create a rate code for the usage like the number of months, days, or hours the sandisk is used.

The attributes displayed in the Billing Rate Table is a one-one mapping of the billing attributes defined for the service item in the Billing Rate Definition tab

Step 5 Click **Save**.

Configuring Accounts for Billing Transactions

An account covers one or more Organization Units (OU)s that can consume services and get billed for what is consumed. Therefore when you order a service, the application verifies if you belong to the OU that is mapped to the account. You can assign and track quotas using accounts.

There are two types of accounts:

- Tenant Account - A tenant account has a one-to-one relationship with the OU.

Create a tenant account when the resource consumption is tracked based on a person's Home OU membership. A tenant account may cover a single OU or a hierarchical organization, i.e., an OU with one or multiple levels of sub-units. A tenant account should be associated with the top OU in the hierarchy. When a member of the organization hierarchy orders a service that has billing or quota impact, the person's tenant account membership is determined in the system automatically by looking up the account associated with his/her Home OU. If there is no associated tenant account, the system repeats the search with the parent OU, and traverses up the organization hierarchy until one is identified.

- Project Account - This is a grouping of OUs. Project account can be associated with multiple OUs.

You create a project account when quota needs to be tracked collectively for a group of OUs that may or may not have a relationship with each other. When a member of the project account orders a service that has billing or quota impact to the account, the project account information should be included in the service item operation to override the tenant account lookup. For example, in a service item task, the associated service item-based dictionary should have the AccountID field included and set to the corresponding value for the project account that is being tracked for consumption.

In most cases, tenant accounts are used to model business units and project accounts are used to model groups established for projects.

Defining Attributes for an Account

The attributes are common to all accounts. If you are a service designer you can use attributes to define any additional attributes that should be captured for every account (e.g. Taxpayer ID). You can define custom attributes as this could vary based on the your domain. The attribute values can be referenced/exposed in policy email notifications. They can also be passed as parameters to policy triggered service requests.

Step 1 Choose **Demand Management** > **Account** > **Define Account**.

Step 2 In Define Account tab, click **Add**. An empty line opens at the bottom of the grid, where you can type your entries.

Step 3 Click **Save**.

To delete an attribute, choose **Demand Management > Account > Define Account**, select the attribute and click **Remove Selected**.

Creating an Account

Step 1 Choose **Demand Management** > **Account** > **Manage Account**.

Step 2 In Manage Account tab, click **Add**

Step 3 Enter the following details:

- **Name**
- **Account Type**: Click on the radio button depending on the account type you want to create.
- **Description**

Step 4 Click **Add**.
The details you entered are displayed in the Account Information panel.

Step 5 In the Account Information panel, do the following:

Table 61: Account Information

Field	Action	Description
Billing Rate Group	Click the lookup icon. Select a Rate Group and click **Add**.	The rate group window displays the rate group that you had already defined. See Creating a Billing Group For Similar Rates, on page 226.
Organizational Unit (OU)	Click the lookup icon. Select an OU and click **Add**.	The organizational window displays the organizations that you had created in the Organization.
Functional Position	Select a functional position and click **Add**.	The functional position window displays the functional position that you created in **Organization Designer > Functional Positions**.
Custom Attributes		The custom attributes table displays the service item attributes you defined in Configuring Accounts for Billing Transactions, on page 229.

Step 6 Click **Save**.

An account may be deleted only if it has no agreement and service item subscription associated with it. To delete an account, choose **Demand Management > Manage Account**, select an account and click **Delete**.

Configuring Billing Rates Based on Usage (Agreements)

Using agreements accounts can be billed based on the services used. You can configure agreements to specify the maximum quota of service item. Therefore only attributes that are managed by quota can be used to create an agreement.

You can also create multiple agreements and sub-agreements and associate it with OUs or accounts. For example you can create a sub-agreement for an agreement such that a customer (OU or tenant account) can distribute his quota for a service item, to his sub OUs.

Agreements are created to enable tenant or project accounts to consume service items that are quota-managed. Service items that have any attributes marked as “Managed by Quota” are resources that need to be regulated for consumption to prevent them from over-subscription by individual accounts. As such, an agreement is a set of service item quotas defined for the associated account, specified in the form of “Count” (maximum number of service item instances) or “Sum” of certain attributes (aggregated total of the attribute values for all instances). The used amounts are also tracked in the agreement.

For example, if there is a need to restrict the maximum number of virtual machines provisioned for each account and the total vCPUs consumed, the service designer would first mark the vCPU attribute in the service item definition as managed by quota, and then create an agreement template which includes a count quota for the number of virtual machines, as well as the sum of vCPU allowed. When a new account is set up, the finance manager creates an agreement from the template to specify the quotas allowed for the account, and define a quota policy to enforce the quota check at the appropriate thresholds (e.g., notify the account manager at 80% consumption, and stop ordering at 100% consumption).

Once the account, agreement and quota policies are in place, when a service item operation is processed to create a new virtual machine for an account, the system performs the following steps:

Step 1 Look up the agreement for the account involved in the operation.

Step 2 Locate the total and used quotas for virtual machine within the agreement.

Step 3 Return an error if no agreement is found, or there is no corresponding quota for virtual machines.

Step 4 Look for any quota policy in place for the service item and account involved. Based on the quota policy condition and actions, determine if the operation should be prohibited.

Step 5 If the quota policy allows the operation to proceed, increment the used quotas of virtual machine count by 1 and the sum of vCPU consumed by this virtual machine in the agreement.

Adding Agreement Templates

Agreement templates are created to enable you to create an agreement effectively. You can define service offering and attributes in the agreement template and then associate the template while you create an agreement.

However you can also add additional attributes in the agreement which might not have been defined in the agreement template. The agreement will then be disassociated from the agreement template.

Step 1 Choose **Demand Management** > **Agreement** > **Agreement Templates**.

Step 2 Click **Add** in the Agreement templates list panel.

Step 3 In the Add New Agreement Template window, enter the **Name** and **Description**, and click **Add**.
The details you entered are displayed in Agreement Template information panel.

Step 4 To define the attributes for the agreement template, click **Add** in the attributes table of the Agreement Template information panel.

Step 5 Enter the following information in the **Add New Agreement Attribute** window:

- Service Item Type
- Aggregate Method
- Service Item Attribute - The drop-down list of the service Item Attribute field is enabled only if you have defined managed by Quota attributes when you designed the service item. See Managing Service Item Attributes, on page 34.
- Quota - Defines the limitation of using the service item. If you are an administrator, you can increase or decrease the quota assignment. This is the only attribute you can change in the template.

Step 6 Click **OK**, and then click **Save**.

Note Quota is the only attribute you can update once an agreement template is created. To change any other attribute you must select the row that contains the attribute that requires to be updated, click Remove Selected, save the agreement template, and add the attributes again with new values.

Adding an Agreement

To add an agreement:

Step 1 Choose **Demand Management** > **Agreement** > **Agreements**.

Step 2 Click **Add**, in the Agreements list panel.

Step 3 Enter the following information in the New Agreement window.

- Name
- Account Type - Choose the account type you want to associate the account with.
- Account - The accounts are listed in the drop-down list based on the account type you have chosen. The organizational unit that is associated with the account is displayed in the Associate Organizational Unit tab.

- Agreement Template - Agreement templates are listed only if you have created agreement templates as described in Adding Agreement Templates, on page 231.
- Description

Step 4 Click **Add**.

Step 5 The content panel of the Agreements tab displays the following tabs:

- Agreement Information - The information provided when you added an agreement is displayed in this panel. It also imports information from the Agreement Template that you associated the Agreement with. Click **Add** to add more attributes in the attributes table.
- Associate Organizational Unit - This tab displays the organizational unit that is associated account. This is blank for a master agreement.

For a master agreement (the top agreement for a tenant account), the tab shows the OU already associated with the tenant account. For sub-agreements (agreements that are created from a master agreement), the tab is editable and one or more OUs can be added here. The OUs are however restricted to only sub-OUs of the tenant account OU. Also each sub-OU can be associated with only one sub-agreement.

Step 6 Click **Save**.

Creating Sub-Agreements

You can create sub-agreements of a parent agreement such that the quota limited for a master account can be shared appropriately with the sub accounts or fiscal periods. You can also ensure that you apply the sub-agreement to a particular OU of all the OUs that the account is associated with.

To create a sub-agreement:

Step 1 Choose **Demand Management** > **Agreement** > **Agreements**.

Step 2 Click **Add**, in the Agreements list panel.

Step 3 Choose an account that is already created from the drop-down list in the New Agreement window.
The **Parent Agreement** field is displayed that lists the agreements associated with the account that you selected.

Step 4 Choose a Parent Agreement from the drop-down list.

Step 5 Click **Add**.
The **Add** and **Remove Selected** buttons are disabled indicating that you can only update the quota limit for a sub-agreement.

Step 6 Update the quota as required and click **Save**.

Step 7 Click **Add** in the Associate Organizational Unit tab, to select the Organizational Units that can be associated with the sub-agreement. Click **Save**.

Pricing a Service

Use the Pricing subtab to configure summary pricing and detailed cost information for the service. Service costs and prices can be used as the basis of chargebacks for delivering the service.

If you need to price service item based on consumption or by the type of operation involved, you can configure pricing in the Demand Management module. For more information, see Configuring a Billable Rates, on page 226 section.

To define the price for a service

Step 1 Choose **Service Designer** > **Services** > **Offer** > **Pricing**.

Step 2 Enter the Cost Details in the **Cost Details** table. See Pricing Options and Dynamic Pricing.

Step 3 Enter the Pricing in the **Pricing Summary** table. See Pricing Options and Dynamic Pricing.

Step 4 Click Save **Pricing Summary**.

Pricing Options and Dynamic Pricing

The options specified on the Pricing Summary determine how price information is displayed in My Services > Service Overview. Only the “Pricing Required” option from Service Designer > Offer > Estimated/Fixed influences the workflow of the service by inserting a pricing moment after the request has been submitted. Because this behavior is inflexible, it may not be suitable for most scenarios where a price must be determined dynamically.

To provide more flexibility, service designers can write active form rules that dynamically set the price of the service. Like any active form rules, rules that set the price for a particular service request can be executed conditionally at any time during the authorization, review, and delivery cycle. The updated “transactional price” of the service request is shown in the Service Catalog > My Stuff > Open Orders or Completed Orders.

Dynamic pricing rules can be used in any services, not just those that are “Priced Dynamically”. The only benefit of using this option vs. “Fixed” is that it better sets the customer’s expectations.

For more information on using rules to dynamically price a service, please see Configuring Dynamic Form Behaviors Using Form Rules, on page 72.

Table 62: Cost Details Information

Field	Definition
Accounting Code	Category of expense, such as Capital expense or labor expense. Click **Add** to add a code. Click **Update** to update information.
Description	Description of the chosen code.
Quantity	Number of the chosen expense to be included.

Field	Definition
Rate	The cost per service. The quantity and the rate are calculated and reflected in the total if you choose the Include option. You can also copy the total into the Pricing Summary if you choose **Copy**.
Cost Driver	Units appropriate for the chosen service such as BTUs or CPUs. The list of available Cost Drivers is configured in the Administration module in Lists.
Time Period	The time period in which the service expense is applied, such as Daily or Once.
Subtotal	Price based on the quantity and the rate for the given expense.
Include	Choose to include the expense in the pricing total.
Copy (button)	Click to include the Total from the Cost Details into the Pricing Summary information.

Table 63: Pricing Summary Information

Field	Definition
Price	The summary price for this service. If you want the summary price to be the same as the total of the Cost Details, click **Copy**.

Field	Definition
Display Options	Customers can click on the service name to get more details about the service before ordering it. If an option to display pricing is chosen, the pricing summary and cost details appear on the Overview page, as shown in the sample below. • **Display both cost and price:** The individual expense lines you specify in the Cost Details area and the Pricing Summary appear to the customer. • **Display only price**: Only the Pricing Summary appears to the customer. • **Do not display cost and price**: Both the Cost Details and Pricing Summary do not appear to the customer. **Note** If you have chosen the Service Catalog module in the main menu, you will see the price for a service. The costs and price description are not applicable to the Service Catalog module.

Field	Definition
Estimated/Fixed	Type of pricing for the service: • **Fixed price**: This option indicates that the price is fixed. • **Pricing Required**: This is both a descriptive field and one that affects system behavior. It activates the Pricing moment for the service and My Services displays the price entered in the Price field as "subject to pricing." The Pricing moment occurs before the Authorization moment, if applicable, and the Delivery moment. If a "Pricing Required" task is created, it is sent to the Default Service Delivery Queue. A Service Manager user with Access Queue permission for that queue must enter a final price during the Pricing moment and click **Set Price** to advance the service request. The Pricing moment is optional. • **Time and Materials**: This option indicates that the price is based on time and materials. • **Leased**: This option indicates that the price is based on lease. • **Estimated**: This option indicates that the price is estimated. • **Priced Dynamically**: This option indicates that the service price is determined dynamically. The type of pricing appears to the customer in My Services unless you choose the "Do not display price" option. The display of cost or price to approvers and task performers is not affected by this setting.
Description	A description of the chosen pricing.

Computing the Price for a Service

While designing services, you can enable the end user to calculate the price for a service before ordering. This will enable the end user to make conscious decisions while ordering a service.

An estimated price defines the billable amount for the dictionary. However, when you enable compute price, it calculates the sum of all the dictionaries for a service form and/ or calculates the price for each dictionary.

You must ensure that a rate table is associated with the billable attributes of a service.

Step 1 Create a Service Item, and do the following:

a) Define attributes for the server, like, Name, Type, or Storage

b) Define billable and non billable operations. Billable operations could be Start Server or Restart Server for a server. Non billable operations could be Stop Server.
For more information, see Defining Service Items for a Service, on page 20.

Step 2 Create a service item-based dictionary for the service item.

a) Enable estimated price as a dictionary attribute. For more information, see Service Item-Based Dictionary, on page 43.

Note The other attributes defined when you created the service item are already enabled.

Step 3 Create a form in the Active Form Components, and add the service item-based dictionary to the form. For more information, see Service Item-Based Dictionary, on page 43.

Step 4 Create a service (You can define the service as Order Server), and do the following:

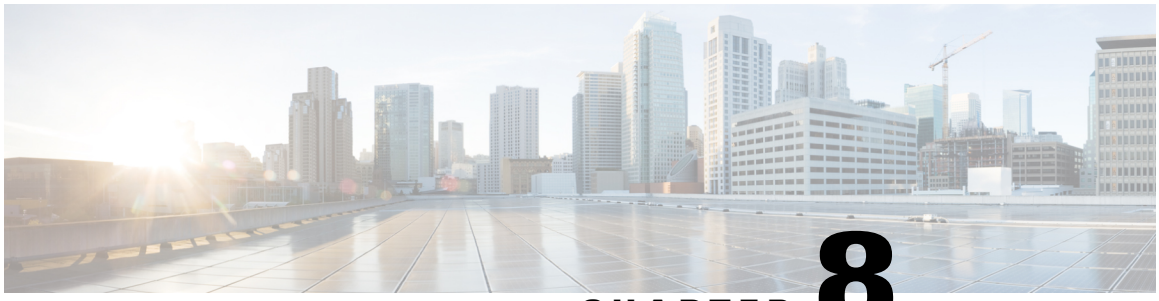
a) Add the form that you created, to the service. For more information, see Configuring an Active Form, on page 64.

b) Create a task. In the **Service Designer** > **Task**tab, define service item workflows and choose the billable operations that you created earlier. The workflow type could be **Service Item Task** and task name could be **Start Server**. For more information, see Defining Service Item Task in a Delivery Plan, on page 188.

c) **Enable Compute Price** option in **Service Designer > General**. For more information, see Creating a Service, on page 158.

Step 5 Create a billing rate table that is applicable to the service item. For more information, see Configuring a Billable Rates, on page 226.

Step 6 Create an account and configure it such that it is mapped with the billing rate group applicable to the service item. Also add the organizational unit. For more information, see Configuring Accounts for Billing Transactions, on page 229. Order the service from **My Services**. Click on **Compute Price** to evaluate the estimated price of the service that you ordered.



CHAPTER 8

AdditionalConfigurationsForDesigningServices

This chapter contains the following topics:

Additional Configurations For Designing Services

This section will cover the additional attributes required for Services, like, setting keywords, defining service objectives.

Defining Service Objectives

Objectives can be defined for a service, but these are not used by Service Catalog. Rather, they are useful for portfolio designers who specify component services for their portfolios. Define and manage the measurable service delivery objectives defined in Service Definition Offer tab.

Note No Service Catalog behavior is associated with the Objectives subtab.

To configure service objectives:

Step 1 Choose **Service Designer** > **Offer** > **Objectives**. Select the service you are configuring in Service Designer module.

Step 2 Choose the service you are configuring in Service Designer module and select **Offer** > **Objectives** >

Step 3 Click **Add Objectives**.

Step 4 Search the defined objectives in the Search window and click **Add**.

Formatting Service Presentation

The appearance of a service within the service catalog is configured on the service's Presentation tab. This includes:

- Associating an image with the service
- Entering text, which could include HTML formatting, to further describe the service
- Associating one or more external links (URLs) with the service
- Deciding which sections of the form to display and the manner in which they display

Adding an Image to a Service Presentation

To associate an image with the service:

Step 1 Choose **Service Designer** > **Presentation**. Select the service you are configuring in Service Designer module.

Step 2 Click **Select an image to insert**. The "Select an image to insert" dialog box appears.

Step 3 If the image has already been uploaded to Service Catalog, it is listed and can be previewed by clicking the View icon. To choose a previously uploaded image, choose the image by clicking its radio button, and then click **Add Selected Files**.

Step 4 If the image has not yet been uploaded, click **Browse** to locate and choose the image, click **Attach** to upload the image to the system, choose the image from the list by clicking its radio button, and then click **Add Selected Files**.

Images that represent services can be in JPG,TIF, PNG, or GIF format. SVG image type is also supported when the category is displayed in the Service Catalog module. By default, images are resized to 64 (width) x 57 (height) pixels in **My Services**, or 50 x 50 pixels in the Service Catalog module. You can change these pixel sizes by using a custom style sheet.

See the Custom Style Sheets chapter in the Cisco Prime Service Catalog Administration and Operations Guide for more information. Service Catalog images should be created to the default (or custom) size and aspect ratio. You may use any image editing software to ensure that the size and aspect ratio of the image matches those dimensions. If the image is not sized correctly, it is resized by the browser and may appear stretched, pixilated, or distorted. Depending upon how you have customized Service Catalog using the custom CSS capabilities, you might also want to consider developing custom icons with a transparent background, or a border.

Previewing the Service Presentation

It is a good idea to preview the service presentation in **Service Catalog** or **My Services** before proceeding.

To do so:

Step 1 Choose the **Service Catalog** module.

Step 2 Navigate to the service and check your work.

Step 3 Return to **Service Designer** and choose the service.

Step 4 Make any necessary modifications, and be sure to click **Save**.

Step 5 Return to **Service Catalog** to verify the service presentation is correct.

Tip Rather than switching back and forth between **Service Catalog** and **Service Designer**, it might be easier to open two browser windows running Service Catalog application; keep the **Service Designer** service's **Presentation** subtab displayed in one, and the **Service Catalog** order page for the service in the other. Make changes as required in the service's presentation, remembering to save the changes by clicking **Save**. As soon as you have clicked Save, you can refresh the **Service Catalog** page to review your changes.

Adding Descriptive Information For a Service

Use Service Details section to associate additional descriptive information with the service. Any of the description entries can include HTML formatting. Additional HTML elements or even JavaScript functions could be referenced by editing HTML source via the Source button provided by the editor.

Step 1 Choose **Service Designer** > **Presentation** > **Service Details**

Step 2 Select which of the following sections you want to Show (and format) content for by clicking the Show radio button for the section:

- **Overview**: Information entered here is visible in **My Services** when the user clicks on the service name or service icon to display an Overview of the service. This page looks different for the Service Catalog menu option.
- **More Details**: If information is entered here, the **My Services Overview** page includes a More Details link directly below the Overview link. This information appears to the right of service information.

  Note The **More Details** section is not used when services are displayed in the Service Catalog module.

- **Service Form**: Information entered here is displayed on the right side of the service form, immediately below the dictionary monitor.

Step 3 If you choose to show multiple sections, click the name of the section you want to format.

Step 4 If desired, enter a URL in the **Overview URL** field to display within the section. The URL must be fully qualified, beginning with "http://."

Step 5 Use the included HTML editor to enter and format text and graphics (standard editor is chosen by default).
Or, to insert HTML code, click **Source** to disable the HTML editing tools and enter your coding directly. Click **Source** again to return to the HTML editor to display the rendered HTML code.

Step 6 Click **Save**.
Repeat for additional sections of the presentation window.

The details you enter are displayed in the Account Information panel.

Step 7 In the Account Information panel, enter the following details:

Field	Action	Description
Billing Rate Group	Click the lookup icon. Select a Rate Group and click **Add**.	The rate group window displays the rate group that you had already defined. See Creating a Billing Group For Similar Rates, on page 226.
Organizational Unit (OU)	Click the lookup icon. Select an OU and click **Add**.	The organizational window displays the organizations that you had created in the Organization Unit.
Functional Position	Select a functional position and click **Add**.	The functional position window displays the functional position that you created in **Organization Designer > Functional Positions**.
Custom Attributes		The custom attributes table displays the service item attributes you defined in Defining Attributes for an Account, on page 229.

Step 8 Click **Save**.

Setting Keywords for Search

Keywords are words associated with a service that are used to support searching for a service within the service catalog. Keywords supplement words in the service name and description, and in categories (based on system settings), already used in a keyword search. You can configure additional keywords, and associate them with the services you configure. All keywords in the system are available to all services in the system.

When adding a keyword, put yourself in the place of the customer and try to anticipate the words that make the most sense.

For example, if you have a “Computer Memory Upgrade” service, possible search keywords might be RAM, memory, expansion, or upgrade.

You can create each of these keywords in Keyword Manager and then associate them to the applicable services.

- Adding a New Keyword, on page 243
- Associating Keywords with Services , on page 243
- Removing Keywords from Services , on page 243

• Deleting Objectives, on page 245

Adding a New Keyword

To add a new keyword:

Step 1 Choose **Service Designer** > **Keywords**.
Step 2 Choose **New** > **New Keyword**.
Step 3 Enter a name for the new keyword in the New Keyword window.
Step 4 Click **Add This Keyword**.
Service Designer adds the new keyword to the list of available keywords, and allows you to associate it with services.

Associating Keywords with Services

When you associate a keyword with a service, a user can employ it in My Services and My Services Executive to search for a service.

To associate a keyword association:

Step 1 Choose **Service Designer** > **Keywords**.
Step 2 Click the name of the keyword you want to associate with a service.
The details for that keyword display to the right.
Step 3 Click **Use in more services**.
Step 4 In the Select services window, select the check boxes next to the services that you want to associate with the keyword.
Step 5 Click **Add**.
Step 6 If you modify the keyword name, or edit the spelling, click **Save** to save changes.

Removing Keywords from Services

You may find that you want to remove a keyword from its association with a service. By removing an association, you are *not* deleting the keyword from Service Designer.

To remove a keyword association:

Step 1 Choose **Service Designer** > **Keywords**.
Step 2 Click the name of the keyword you want to remove.
Step 3 Check the check box next to each service you want removed from association with the keyword.
Step 4 Click **Remove**.

You can also completely remove a keyword by deleting it in the Keywords component of Service Designer. Choose **Delete** and Service Designer deletes the keyword and removes its associations to services.

Adding a Service Objective

To add service-level delivery objectives:

Step 1 Click **Service Designer** > **Objectives**.
Step 2 Choose **New** > **Add New Objective**.
Step 3 Create the new objective. See the table below for field descriptions.
Step 4 Click **Add This Objective** to save it.

This table summarizes the fields on the Objective screen.

Field	Definition
Name	Enter a name for the objective.
Objective Owner	Click the ... button to choose a person to own the objective. **Note** Assigning an Objective owner is an optional action that can be taken for the purpose of documentation. There is currently no system behavior or functionality associated with Objective ownership.
Metric	Choose a metric from the drop-down menu. The metric is the item you are tracking, whether it's budget spent or response time by your service team. The list of available Metrics is managed on the Lists tab in the Administration module where you can add, modify, and delete user-defined Metrics.

Field	Definition
Unit of Measure	Choose the unit of measure from the drop-down menu. For example, if your metric is "Number of Incidents", your UOM would be "Incidents". Or, if your metric is "Budget Spent", you could measure the budget by weekly, monthly, or annual spending. The list of available Units of Measure is managed on the Lists tab in the Administration module where you can add, modify, and delete user-defined Units of Measure.
Description	Enter a description of the objective.
Target	This is the specific measurable target that is committed to in the Objective.
Benchmark	This is an optional reference point, provided by IT, which references the industry standard for this Objective. Measured against the target, the benchmark indicates whether IT is performing on par with other IT service organizations, or better or less than the industry average, for a particular Objective.
Consequence	A consequence is the action that is taken if the service organization does not meet the proposed target for delivery. For example, the customer might be offered a 5% reduction in charge-back fees for periods in which the target is not met.

Deleting Objectives

To remove an objective entirely from the system, delete the objective in the Objectives component of Service Designer.

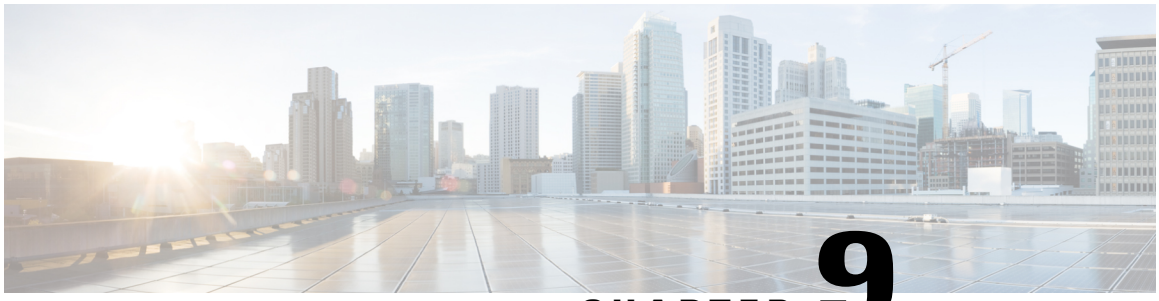
To delete an objective:

Step 1 Click **Service Designer** > **Objectives**.

Step 2 Click the name of the objective you want to remove from the tree menu.

Step 3 Click **Delete**.

Step 4 Click **OK** to confirm the deletion.



CHAPTER 9

Managing the Services and Attributes

This chapter contains the following topics:

Managing the Services and Attributes

Manage Service Items tab allows service item administrators to:

- Review the service items currently tracked by Service Catalog, no matter how they were added to the system.
- Add new service item instances.
- Update information related to a service item such as service item details, services associated with the service item, permissions provided for each service item.

Existing service items are listed in alphabetical order by name.

Step 1 Choose **Service Item Manager** > **Manage Service Items** >

Step 2 Select appropriate Actions, see Table 64: Actions in Manage Service Itemstable.

Actions available are summarized in the table below. Additional information is provided in the following sections.

Table 64: Actions in Manage Service Items

Action	Description
New	Add a service item of the specified type.
Delete	Delete the chosen service item.

Action	Description
Assign	Assign the service item to the chosen person (customer).
Un-Assign	Remove the customer to whom the service item is currently assigned.
Export to Excel	Export the list of service items to a text file in CSV (comma-separated values) format and display that file as an Excel spreadsheet. To export service item data to spreadsheet format, click **Export to Excel**. A text file in CSV format is produced. You can open the file in Excel or save it to your local disk for future use. **Note** The exported CSV file must be opened and edited only using the open source spreadsheet editor-LibreOffice, to ensure that the Japanese characters are displayed properly.
Save View	Save as your default service item view to the current view, with filters and field selection criteria in effect. To save the filter and search criteria currently in effect, click **Save View**. If you exit from Manage Service Items and return, the saved filter criteria will be in effect. To revise the saved view, click **Filter and Search** and edit the criteria as appropriate. Click **Save View** after you have applied those changes to replace the view with the edited or default view (if you remove all criteria).
Filter and Search	Display the popup window to configure search parameters. The Filter and Search option allows you to restrict the service items that are displayed. The filter criteria include all attributes that comprise the service item, as well as the customer and organizational unit to which the service item is assigned; the date the service item was assigned to that person; and the requisition ID through which the service item was created. By default, filter criteria for alphanumeric fields use a “Contains” filter. For example, entering ‘386’ (without the quotation marks) in the Name field will find all service item instances whose name includes the string ‘386’.

Action	Description
Permissions	You can define permissions to view or edit service items based on the role assigned to a user. For more information on permissions that can be added, see Assigning Permissions section in Cisco Prime Service Catalog Administration and Operations Guide .

Managing Service Request Initiated by Other Authorized Person (Order-On-Behalf)

The Service Catalog order-on-behalf process allows a service request initiator to order a service on behalf of another person, the intended customer. A drawback of this approach is that the intended customer must have ordering permissions for the service. This contradicts some corporate policies where, for example, IT or administrative personnel should be able to order services for other people, but people not in the IT or administrative roles should not be able to order these services for themselves.

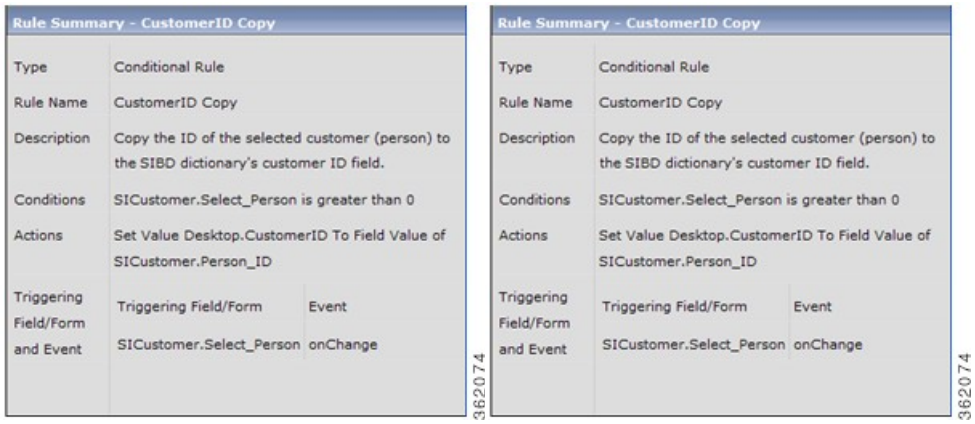
Without service items and service item-based dictionaries, a workaround for this situation is possible but hardly ideal. An initiator could choose the person for whom the service was intended (using a person-based dictionary), but there was no way for that person to be able to track “his” requisitions. This situation is remedied by the use service items and service item-based dictionaries. An initiator can order a service (and provision a service item) and designate a chosen person as the customer. The customer would then be able to monitor the request in the form of a service item that would appear in his Service Items page.

The procedure for implementing this functionality is fairly straightforward:

1. Design the service item.
2. Create the service item-based dictionary, being sure to include in the dictionary the CustomerID and OrganizationalUnitID. Optionally include a user-defined Status field, which can be updated as the delivery plan progresses.
3. Include the SIBD in an active form component.
4. Create a person-based dictionary to allow the initiator to choose the customer for the service item. Include this dictionary either in the same active form component as the SIBD or another form component—as long as both dictionaries are included in the service.

5 In the active form component containing the person-based dictionary write two rules as shown below, to copy information on the chosen person to the service item-based dictionary.

Figure 20: Rule Dictionary



An administrator is now able to order the service for various customers, and the customers can see the service item in the Service Items page or My Items portlet (provided they have been granted the My Services 360-Degree Consumer or Professional role.) If the service item has associated services for which the owners of the SIs have ordering permission, they are able to order those services directly from the Service Items page.

A drawback of this approach is that the requisition would not be searchable by "Customer", within Service Catalog's online modules, including Reporting and Advanced Reporting. In Advanced Reporting, the Service Item's Customer replacement dictionary (in the example above, SICustomer) would need to be reportable.

Retrieving Service Items, History, and Subscriptions

You should generally not have to write a SQL query using the service items, history and subscriptions table. Most of this information is provided to users via the My Items portlet and page. However, it is possible to write a data retrieval rule retrieving information from these tables. For example, perhaps you do not want a user to request a service which provisions a service item if he already has one.

In order to write SQL-based data retrieval rules, you need to know a little bit about the table structure.

- As specified above, the name of the table corresponding to a service item is the item's name prefixed by "Si". So, an SI with the display name **State of the Art Laptop**, with a table name of **StateOfTheArtLaptop** has a corresponding table in the database named **SiStateOfTheArtLaptop**.
- How is that table related to ServiceItemSubscription? The easiest way to join it is on ServiceItemTypeName. So, if you want to get the subscription information for a State of the Art Laptop named **joe-nov13-2013**, this query works:

```
select * from SiServiceItemSubscription
 where ServiceItemTypeName = 'State of the Art Laptop'
   and DisplayName = 'joe-nov13-2013'
```

Or, you could get a list of all State of the Art Laptops by executing this query:

```
select*from siserviceitemsubscription sis
join siStateoftheArtLaptop al
```

```
on al.PrimaryID = sis.ServiceItemIDpredefined
where sis.serviceitemtypename ='State of the Art Laptop'
```

- And then how do I get from there to the ServiceItemHistory records? The key is the **PrimaryID** in the Service Item Subscription equals the **OwnerID** in the Service Item History. So:

```
select sih.*, sis.*from SiServiceItemHistory sih
join siServiceItemSubscription sis
on sih.OwnerID = sis.PrimaryID
where sis.ServiceItemTypeName ='State of the Art Laptop'
and sis.DisplayName ='joe-nov13-2013'
```

Duplicating Services

Export/import is provided primarily for backward compatibility with previous versions of Service Catalog. It provides a way to copy or "clone" an existing service by downloading the service definition to the local workstation; modifying the service name; and importing the revised service definition. Cloning a service is best accomplished by Copying a Service, on page 253.

Exporting and Importing a Service

The procedure for using service export/import is as follows.

1. Configure the service.
2. Export the service.
3. Modify the service name in the exported XML file.
4. Import the modified XML file.

Although service export/import may work in transferring a service definition from one site to another, this is not recommended. The export includes associations to people, OUs, queues, roles and other entities that are not part of the service definition. Importing the service does not create these entities in the target environment if they do not already exist; it simply drops the association without issuing any error messages.

Tip Catalog Deployer is the recommended tool for transferring Service Catalog content between sites.

Note Passwords for 'Persons' and 'Agents' data imported from a different database (using Catalog Deployer or REX), must be reset in the target database. This is because the Key Encryption Key is different for different instances of Prime Service Catalog databases. If the passwords for 'Persons' and 'Agents' are not reset, they cannot be decrypted correctly in the target database.

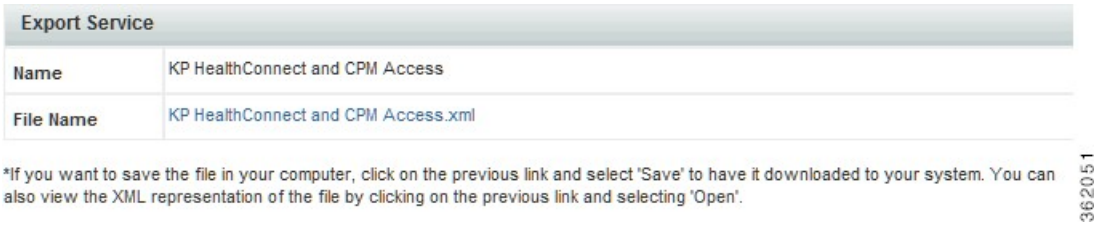
To export a service:

Step 1 Choose **Service Designer** > **Services**. Select the service you want to export.

Step 2 From the General tab, click **Export**.

The Export Service window appears.

Figure 21: Export Service



Step 3 Right-click the File Name link and choose **Save Target As** to save the file to your desired location.
Service Designer exports the service in XML format to the location you designated.

Caution If you are using Internet Explorer, you may experience difficulty saving the XML file. If this occurs, go to **Tools > Internet Options > Advanced Tab > Securit**y and check the **Do not save encrypted pages to disk** option. This option is unchecked by default.

Editing the Exported XML File

To edit the exported XML file that is saved in the user specified location.

Step 1 Open the XML file in a plain text editor such as Windows Notepad.

Step 2 Change the service name in the file where it reads:

Example:

```
<ServiceDefinition name="service name here" ....>.
```

If you do not change the service name within the XML file, the import process will overwrite your original service. (It is possible to change other elements in the XML file, but this practice is not recommended. If invalid XML is produced, the import will fail.)

Step 3 Click **Save** to save the edited XML file.
The edited XML file is now ready for import.

To import a service:

To import a service:

Step 1 Choose **Service Designer** > **Services** > **New** > **Import**.

Step 2 In **Import Data** window, browse to or enter the filename (including the path) in the Import from file field.
You can only import an XML file that has been previously exported and saved to its default location ora location on the local machine.

You can also use the Browse function to choose a file on your workstation.

Step 3 Click **Import**.
Service Designer displays an Imported Data screen to confirm the import.

Step 4 Click **OK** to refresh the Service Catalog display. The imported service appears in the service group specified in the XML file.

Copying a Service

Service Designer includes the ability to copy a service definition. Copying a service copies the complete service definition, including any associations with keywords and categories as well as the associations to included active form components.

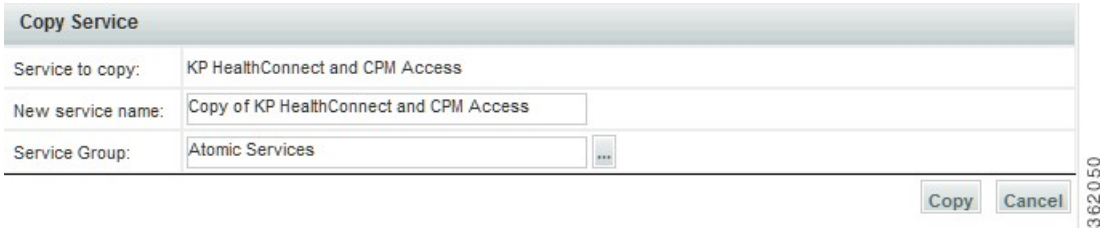
Copy a Service

To copy a service definition by reusing the existing Active Form Components (AFC) and Dictionaries:

Step 1 Choose **Service Designer** > **Services** > **General**.

Step 2 Click **Copy** on the General tab of the service definition. The Copy Service popup window appears.

Figure 22: Copy Service



Step 3 Enter the name of the new service and choose the service group into which the new service definition should be placed. Only service groups in which the current user has permission to “Design forms” are available for selection.

Note Copying a service to a new group, does not affect the service group authorizations of the target group—these remain the same.

Clone a Service

Clone a service option allows you to copy a service definition along with the Active Form Components, dictionaries, email templates, and scripts. You could, however, choose to clone all the entities or only some of them. For all the entities you choose to clone, you must provide a new name and group to which it must belong. In case you choose not to clone some of the entities, then they are referenced to the entities in the original service. Unchecking the main check box for dictionaries only means that the reference will be maintained with the original service and not that the forms are not associated with the service at all.

To clone a service:

Step 1 Choose **Service Designer** > **Services**.

Step 2 Select the service to be cloned from the left hand side.

Step 3 Click **Clone** on the General tab of the service definition.
The Clone Service window appears.

Step 4 Enter a name for the new service and choose the service group into which the new service definition should be placed. Only those service groups in which the current user has permission are available for selection.

Step 5 Choose the AFCs, Dictionaries, Email Templates, and Scripts to be cloned.

Note The unchecked AFCs, dictionaries, email Templates, and scripts are referenced to the corresponding entities of the original service.

Step 6 Enter a new name for the selected entities.

Step 7 Select the group into which the new AFC and dictionary must be placed from the drop-down list.

Note
- Only those form groups in which the current user has permission are available for selection.
-
- If the *Form Groups* and *Dictionary Groups* are not specified they are saved to the old form/dictionary groups.

Step 8 Click **Save** to save the new service.

Tracking Service Design Changes

You can track the changes done to the Service Designer entities via History option. To track the changes performed by various service designers and to store the data in the database, you must first enable the service design change history option. For more information on how to enable service design change history, see Cisco Prime Service Catalog Administration and Operations Guide .

For now, service design change history is supported only for Services, Dictionaries, and Active Form Components entities. There are various other attributes for each of these entities, which are also tracked but not in detail. The track changes are limited to support create, update, and delete operations only to the following attributes and can be viewed from Change Details table:

- Services

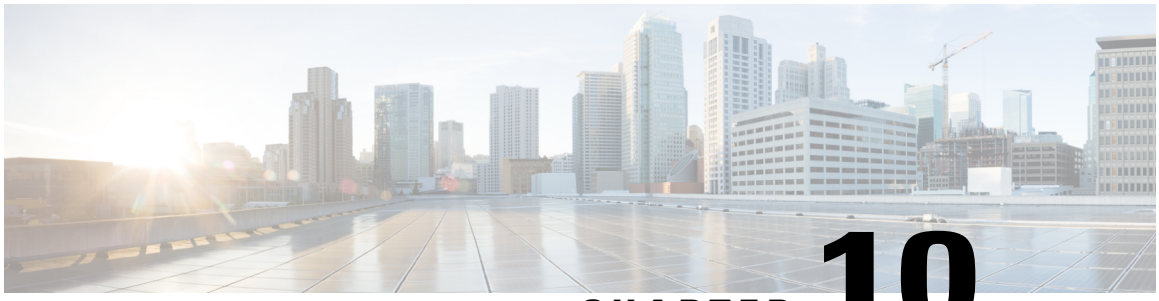
- Name and Description fields
- Other fields, like Additional URL, Hide in Service Catalog, Standard Duration, Service Group, Image URL, and so on.

- Active Form Components
  - Name and Description fields
  - Addition and Deletion of dictionary
  - Display Properties of the Form. For example: Change in the Input Type field
  - Active Form Rules (Conditional rule)

- Dictionaries
  - Name and Description fields
  - Dictionary Attributes

To view service design change history:

Step 1 Select **Service Designer** > **History**.
Step 2 Enter the **Start Date** and **End Date**. The dates should not exceed 60 days.
Step 3 (Optional) Select the type of **Entity** from the drop-down list.
Step 4 (Optional) Select the user name or the user group from the **Users** drop-down list.
Step 5 Click **Filter**. The **Change Details** table is displayed.
Step 6 (Optional) Select a row from the table to see the details of a specific entity, which is displayed in a new pop-up window.



CHAPTER 10

Designing Portlets and Portals Using Portal Designer

This chapter contains the following topics:

Designing Portlets and Portals Using Portal Designer

Portal Designer is the Service Catalog module that allows designers and administrators to design and manage pages and portal content and to specify which users or groups or users are able to access particular content.

Portal Designer addresses many user interface customization requirements by providing administrators and designers with finer control over the appearance of their Service Catalog implementation. At the same time, the portal platform allows multiple external data sources to coexist within Service Catalog screens, providing a holistic view into Data Center or general IT services and resources.

The portal front-end provides a way to interact with services, service items, standards, offerings, and other core entities in the application, by integrating portlets exposing this content into the portal. Portal Designer provides an interface to build a variety of portlets using application data, JavaScript/HTML, ad hoc lists, or third-party JSR-compliant portlets.

Portal Designer allows interface designers to:

- Create portlets from external or third-party sources
- Create portlets to highlight common services
- Create portlets to show users what they already own, with links to services related to those items
- Show announcements, video, or other types of media
- Leverage RBAC to create a flexible user interface that is at once simple for casual users, and advanced for power users

Note Portal Designer is a separately licensed module of Cisco Prime Service Catalog. You must be licensed to run Portal Designer in order to use the content management functionality. Author Notes: Check whether we have explained “content management functionality” in earlier chapters.

Portal Designer Roles and Capabilities

Site administrators can use the Organization Designer module to grant users access to the Portal management modules.

Access to the capabilities provided by Portal Designer is controlled via standard Role-Based Access Control (RBAC). Design personnel can be granted access to all or selected portions of the Portal Designer functionality. Capabilities to customize the portal's appearance or manage portlet content can be assigned to selected users, roles or groups through the use of Role-Based Access Control (RBAC). These capabilities include:

- Drag and drop user interface, fashioned after MyYahoo and iGoogle portals
- User-selected skins
- User-selected content
- Ability for users to create their own portal pages

Details on the portal-related capabilities and how to assign these to project personnel are given in the Cisco Prime Service Catalog Administration and Operations Guide , and in the Organization Designer Online Help regarding roles.

Similarly, end users' ability to access the portal front-end can be controlled via RBAC. Only users with a role that includes the “Access Service Portal” capability are able to see the “Service Portal” module menu and navigate to the portal pages and portlets.

Table 65: Portal End User Roles

Role	Description
Portal Basic User	Enables users to access the portal front-end and view portal pages defined by the portal administrators. End users of Cisco Prime Service Catalog who need access to the Service Portal module should be assigned the Portal Basic User role.
Portal Advanced User	Enables users to access the portal front-end and manage the content and presentation of their portal pages.
Portal Professional User	Enables users to manage portal pages and make them available to other users. This user can initiate and track service requests, authorizations, and service items on behalf of others in their business units and access those transactions in portlets.

Users of these roles require read permissions to particular portal pages and portlets in order to put the pages on their portal and view the portlets.

Users may need to be granted additional capabilities and permissions if they need to access other modules through hyperlinks on the portlets and to see the content in the portlets.

Configuring Portlets

A portlet is a software module that can be plugged into a portal page and arranged as non overlapping portions of the page. A portal page can include one or more portlets. The following are the two types of portlets available for users in the application:

- **Reserved Portlets**, These are preconfigured portlets that are installed with every application instance. See Customizing Reserved Portlets , on page 273.
- **User-defined Portlets**. These are JSR portlets or portlets developed using the Portal Designer. Portal Designer allows the designer to define the content and presentation of the portlets with predefined filters and lookup, HTML, and JavaScripts. A JSR portlet may be developed in any Java development environment that is compliant with JSR 168 or 286, and optionally using the Service Catalog Java Client to leverage the application public APIs. A third-party JSR portlet can also be easily integrated into the portal management solution.

Note The preconfigured portlets available in the My Services module are not true portlets. They are not available in Portal Designer and cannot be added to a portal page. Some examples are My Authorizations and My Requisitions.

Portlets leverage the Service Catalog REST API which support the RBAC-enabled access to the application data. The API framework, along with functionality for defining the appearance and behavior of portlets, allow portal designers to easily include predefined content in a portlet and to configure that portlet for inclusion in a portal page. The portlet content may consist of many types of data available within Service Catalog such as:

- Definitional data (agents and service definitions)
- Directory data (people and organizations)
- Transactional data such as requisitions
- Service items and service standards

In addition, designers can define their own Custom Content, maintainable through Portal Designer, for inclusion in portlets.

Creating and Configuring User-Defined Portlets Using Portal Designer

- Creating a New Portlet, on page 260
- Configuring Portlet View, on page 262
- Defining Portlet Filter Criteria, on page 263
- Configuring Portlet Permissions, on page 265

Creating a New Portlet

Step 1 Click **Portal Designers** > **Portlets** > **Actions** > **New Portlet**

Step 2 Enter portlet details as described in the table below.

Step 3 Click **Add**.

Step 4 Depending on your requirements, do the following:

- Configure the portlet view. See Configuring Portlet View, on page 262.
- Define the portlet filter criteria. See Defining Portlet Filter Criteria, on page 263
- Configure portal permissions. See Configuring Portlet Permissions, on page 265

Table 66: Configuration Table for Adding Portlets

Field	Description
Display Name	The name to be displayed as the default portlet title; free-format.
Name	The internal name for the portal; can contain only letters, numbers, and the underscore character (no spaces).
Author	A text string which defaults to the first and last name of the current user; may be edited.

Field	Description
Content Type	The type of content of the portal. Options are: **Core Entities** Entities used by Service Catalog; all such objects are listed under the Reference tab of Portal Designer. **Service Items** Any system- or user-defined service item, defined via Service Item Manager. **Standards** Any system- or user-defined standard, defined via Service Designer. **HTML/JavaScript** The portal designer will design the portlet according to rules specified in the Defining HTML and JavaScript Portlets, on page 265. **Custom Content** User-defined tables, defined and maintained using the Custom Content tab of Portal Designer. See Configuring Custom Content Portlets, on page 269.
Source	Once the Content Type is chosen, a drop-down list of data sources available for that type appears. One is chosen as the basis of portlet content.
Description	Optional documentation on the portlet.
Automatic Login	You can associate a portlet that makes use of automatic authentication with the external site by choosing the site name and checking the **Automatic Login** check box.

Once the portlet has been saved (added), the rest of its definition can be provided using the tabs available in the content portlet information pane on the **General** tab. The portlet is assigned to a folder corresponding to its content data source, and selectable from the list pane on the left of the Portal Designer window.

- All portlets are created in an “Active” status. Only “Active” portlets can be included on a portal page.
- To disable portlets from use in portal pages, set the status of the portlets to “Inactive” and remove them from the pages in which they are included. Inactive portlets that are still present in portal pages are hidden from the users.
- Keywords can optionally be associated with a portlet to allow users to search for portlets when adding content to portal pages. Such keywords are defined using the Portal Settings tab.

Configuring Portlet View

Content portlets are implemented with a Grid view. For such portlets, which reference a Service Portal entity, the View tab allows designers to specify which columns from the chosen data source to display (via the "Select Columns ..." grids), and (optionally) which rows to display (via the Filter subtab). The view properties specify the default appearance of the portlet when it is included in a portal page. These can be overridden by individual users and on individual pages.

Step 1 Click **Portal Designer** > **Portlets**

Step 2 Select a portlet from the portal tree and click **View**.

Step 3 Enter the view properties as described in the table below.

Step 4 Click **Save**.

Table 67: Field Configuration Table for Portlet View

Field	Description
Height (px)	The initial height (in pixels) of the portlet; not applicable when the Auto Height setting is enabled.
Auto Height	Check box that indicates whether the height of portlet should be sized to the content automatically; not applicable to HTML portlets.
Auto Scroll	Check box that indicates whether a scroll bar should be displayed in the portlet; not applicable to HTML portlets.
Portlet State	Whether the portlet should initially be displayed in its Normal or Minimized view.
Show Portlet Title	Check box that indicates whether the portlet title (Display Name) should be displayed at the top of the portlet.
Show Controls in Portlet Title	Check box that indicates whether the controls on the title bar such as Minimize and Maximize buttons should be displayed.
Select Columns for Normal View	Provides a summary/overview of portlet content. Up to three columns can be included and data sorted by the contents of one of those columns.

Field	Description
Select Columns for Maximized View	Provides more detailed portlet content and can include any number of columns from the portlet's data source. • To move one or more columns between the Available Columns and Visible Columns panes, click the columns (**Ctrl-Click** to choose multiple columns) and click the upper left- and right-Arrows buttons  Click the lower left- and right-Arrow buttons to move all columns  Click the up- and down-Arrows  to rearrange the order in which the Visible Columns appear.
Sort By and Sort Direction	Sorting, in ascending or descending order, can be applied to any of the columns chosen in a view. The specified data is displayed in a user-configurable grid which may be sized to fit on a portal page. The columns that comprise Service Catalog core entities and those supported for sorting are described in the Integrating Service Catalog Entities in Portlets, on page 275. Other data sources for content portlets (Custom Content, Service Items, Standards) are site-specific—see your service catalog designers for detailed information.

For HTML or JavaScript portlets, the View subtab presents an editor for the designer to enter the source code that defines the appearance and content of the portlets. See Defining HTML and JavaScript Portlets, on page 265.

Defining Portlet Filter Criteria

Defining a portlet filter criteria allows designers to filter the data retrieved from the designated portlet source.

Note The Filter subtab is available only for portlets that are defined by directly referencing a Service Catalog entity.

Step 1 Click **Portal Designer** > **Portlets**.

Step 2 Select a portlet from the portal tree and click **Filter**.

Step 3 Click **Add** and choose filter criteria. For more information, see the table below.
Query criteria are formulated by specifying one or more filters. Each filter consists of a relational statement comparing the value of a column in the entity on which the portlet is based to a specified. Multiple filters can be combined by ANDing or ORing individual filter specifications; rules of precedence apply.

The filtering criteria available for selection are dependent on the entity referenced in the portlet. The drop-down lists for column and operation are context-sensitive and show only the supported combinations in the underlying REST API of the entity.

Step 4 Click **Save Filter**.

Table 68: Filter Criterion Table for Portlet Filters

Filter Criteria	Descriptions
Categories, services, service offerings, and agreements	Permissions to Order the Service/Initiate an Offering are applied on top of the filtering criteria. Portal end users are able to view only data to which they normally have access through the My Services module
Directory entities and agents	RBAC read permissions are enforced along with the filtering criteria. Portal end users are able to view only data which they would see in Organization Designer and Service Link.
Service items, standards and custom content	All attributes are available as filters. In addition, subscription-based filters are available to display service items based on the user's access permissions: • ◦ My Service Items: Instances of the chosen service item owned by the portal user, or instances owned by the portal user's business units; the latter is applicable to users who have the My Services capability "View Service Items for My Business Units". ◦ All Service Items: All instances of the chosen service item if the user has the Service Item Manager "Manage Service Items" capability.

Filter Criteria	Descriptions
Requisitions, authorizations and tasks	Context-specific filters are available to restrict what the portal user can view in the portlets. The filters mimic the predefined views that are available in My Services and Service Manager (for example, Ordered for Myself, My Assigned and Unassigned Authorizations, Available Work).

Configuring Portlet Permissions

Configuring portlet permissions allows designers to designate which users should be able to access the portlet and the type of access to grant. Any users so designated should also be granted the Portal module capability to "Access Service Portal".

Step 1 Choose **Portal Designer** > **Portlets**.

Step 2 Select a portlet from the portal tree and click **Permissions**.

Step 3 Click **Add Permission**.

Step 4 From the **Object Type** drop-down list, choose one of the following: **Organizational Units, Group, Person,** or **Role**.

Step 5 Fill in all or part of the name of the object in the search box and click **Search**.
To display all objects of the specified type, leave the search box blank. To display objects whose name matches a particular pattern, you may include the wildcard character (*) in the search string.

Step 6 Click the rows to which the permission is to be granted. Use **Shift-Click** and **Ctrl-Click** to choose multiple rows.

Step 7 From the **Permission To** drop-down list, select **Read or Read / Write**.

Step 8 Click **Add**.

Portlet permissions also control which portlets users can access in Portal Designer if they have the "Manage Portlets" capability. The user who creates the portlet is automatically granted all access permissions to it.

Note RBAC filtering is applied to the objects available for assigning permissions. In other words, the portal designer needs to have read or read/write permissions to organizational units, person, groups and roles in order to search for them in the Add Permission pane and to view them in the permission summary grid once they have been added. To enable all users to view the portlet, portal designers can assign Read permission to an "umbrella" organizational unit which is the parent of all business units. Alternatively, they can work with the organization designer to grant the portlet permission to the "Anyone" role in the Organization Designer module.

Defining HTML and JavaScript Portlets

HTML and JavaScript portlets provide the ability to define free-format portlets and use those portlets within portal pages. Such portlets can be defined and maintained completely within Portal Designer. They must conform to the coding rules described in this section.

For HTML and JavaScript portlets, the View and Filter subtabs are disabled as all the data displayed in the portlet is provided using HTML or JavaScript code.

HTML Portlets

An HTML portlet consists of an HTML snippet or a URL.

After you define an HTML portlet, the View subtab adjusts its contents so that:

- The View Type is by default set to Web Page.
- The designer can designate the portlet subtype (HTML or URL).
- An edit window for data entry of the appropriate type of text appears.

URL

The URL provides a hyperlink to the specified web page. It can be an absolute reference to an external website or a relative reference to a Service Catalog page, for example: /RequestCenter/myservices/navigate.do?query=orderform&sid=14 .

Authentication settings can be optionally associated with a URL-based portlet to allow automatic login to the external site. The common settings for the external site authentication are defined in the Portal Settings tab in Portal Designer. Credentials for individual users are maintained through the Edit Password tab in the portal front-end. See Authentication Setting fields in Table 80: Portal Setting Options Table table for more details regarding the different options for configuring external site authentication.

HTML Snippet

The HTML snippet can include:

- tags, for applying styles to portions of the portlet
- <script> definitions or invocations of JavaScript functions defined in local script, defined within the HTML snippet.

Make sure that the snippet does not include <head> or <body> tags because the portal is rendered as part of the page body.

JavaScript Portlets

JavaScript portlets can display dynamic content and use the full range of JavaScript functionality. The user interface of JavaScript portlets should be written using ExtJS functions, since ExtJS is the UI framework used for the portal front-end. The complete reference can be found at the Sencha website.

Like the content portlets, JavaScript portlets may consist of data available within Service Catalog which is accessible through the use of the REST API. See the *Cisco Prime Service Catalog Integration Guide* for details regarding the APIs available.

After you define a JavaScript portlet, the View subtab is set to JavaScript and a text area is provided for entering the code.

Example: Using REST API and EXTJS to Create a Grid Portlet

The key concepts for using REST API and EXTJS to create a grid portlet are illustrated in the following sections with sample code snippets.

1. Retrieve the data you want to show in the portlet from the REST API. See Retrieving Data from REST API.
2. Create a grid for displaying the data in the portlet. See Rendering Data in EXTJS Grids.

Retrieving Data from REST API

Step 1 Identify the appropriate REST API to use based on the content type and filtering method required.
For example, if the portlet is used to display all the orderable services for a particular category, the REST API to be used is:

Step 2 Define an array that contains all the attributes of the content type as defined in the REST API.

Example:
```
fieldList = [ "serviceId",
              "serviceName",
              "description",
              "topDescription",
              "middleDescription",
              "bottomDescription",
              "pricingScheme",
              "revisionNumber",
              "status",
              "statusId",
              "expectedDuration",
              "expectedDurationUnits",
              "price",
              "priceDisplaySchemaId",
              "priceDescription",
              "canStartLater",
              "isBundle",
              "dateQualityId",
              "serviceLevelDescription",
              "isOrderable",
              "isReportable",
              "serviceURL"];
```

Step 3 Create the proxy for REST HTTP GET calls.

Example:
```
var proxy = new Ext.data.HttpProxy({
    url: '/RequestCenter/nsapi/definition/servicedefs',
    method: 'GET'
});
```

Step 4 Create an XML data store for the result set, including an XML reader that defines the parameters.

Example:

```
var store = new Ext.data.XmlStore({
    autoDestroy: true,
    storeId: 'myStore',
    proxy: proxy,
    root : "services",
    record: 'service',
    idPath: 'rowId',
    totalProperty: '@totalCount',
    autoLoad: true,
    paramNames: {
      start: 1,
      limit: 10,
      catName : 'Sample Category'
    },
    fields: fieldList
```

Note In addition to REST APIs, AJAX calls can be invoked to retrieve data from other sources to provide the content for the portlets.

Rendering Data in EXTJS Grids

Rendering Data in EXTJS Grids

Step 1 Define an array for the columns and appearance of the grid to be used to display the content.

Example:

```
displayList = [
                {id: 'id', header: 'Id', width: 50, sortable: true, dataIndex: 'serviceId'},
                {header:'Service Id',dataIndex: 'serviceId',hidden:true},
                {header:'Service Name',dataIndex: 'serviceName'},
                {header:'Description',dataIndex: 'description'},
                {header:'Top Description',dataIndex: 'topDescription',hidden:true},
                {header:'Middle Description',dataIndex: 'middleDescription',hidden:true},
                {header:'Bottom Description',dataIndex: 'bottomDescription',hidden:true},
                {header:'Pricing Scheme',dataIndex: 'pricingScheme',hidden:true},
                {header:'Revision Number',dataIndex: 'revisionNumber',hidden:true},
                {header:'Status',dataIndex: 'status'},
                {header:'Status Id',dataIndex: 'statusId',hidden:true},
                {header:'Expected Duration',dataIndex: 'expectedDuration',hidden:true},
                {header:'Expected Duration Units',dataIndex: 'expectedDurationUnits',hidden:true},
                {header:'Price',dataIndex: 'price',hidden:true},
                {header:'Price Display Schema Id',dataIndex: 'priceDisplaySchemaId',hidden:true},
                {header:'Price Description',dataIndex: 'priceDescription',hidden:true},
                {header:'Can Start Later',dataIndex: 'canStartLater',hidden:true},
                {header:'Is Bundle',dataIndex: 'isBundle',hidden:true},
                {header:'Date Quality Id',dataIndex: 'dateQualityId',hidden:true},
              {header:'Service Level Description',dataIndex: 'serviceLevelDescription',hidden:true},

                {header:'Is Orderable',dataIndex: 'isOrderable',hidden:true},
                {header:'Is Reportable',dataIndex: 'isReportable',hidden:true},
                {header:'Service URL',dataIndex: 'serviceURL'}
];
```

Step 2 Create an EXTJS grid, using the column array and data store defined in the earlier steps.

Example:

```
var grid = new Ext.grid.GridPanel({
         store : store,
         columns : displayList,
         renderTo : '#divName#',
         width : "100%",
         autoHeight : true,
         layout : 'fit',
         viewConfig : {
            forceFit : true
         },
         tbar : [combo,filterButton],
         bbar : [new Ext.PagingToolbar({
                  store : store,
                  displayInfo : true,
                  pageSize : 5,
                  params:{
                           startRow: 1,
                           recordSize: 5
                         }
                  emptyMsg : "No record found"
```

```
                    })]
});
```

Configuring Custom Content Portlets

Custom content comprises user-defined tables that serve as a source of content for the portal. Such tables can be referenced as the data source for portlets just like Standards. Such tables are defined and maintained in the Custom Content tab in Portal Designer and are organized into content groups.

Content groups allow custom content tables to be grouped in a logical manner for easier navigation and control of access permissions in Portal Designer. Read/write permissions can be granted at the group level for managing all content tables within the group or at a more granular level for individual tables.

The “System” content group is available by default. It contains two commonly used custom content definitions—Announcements and Links—to provide convenience to portal designers.

Creating and Configuring a Custom Content Table

Step 1 Choose **Portal Designer** > **Custom Content** > **Actions** > **New Content Definition** to define a new custom content table.

Step 2 Enter the field details as provided in the table below and click **Add**.

Step 3 Update the rest of its definition using the tabs available in **Content Definition** tab.
Content Definition comprises of the name, description and table columns that are characterized by the following four attributes:

- Display Name – Label for the table column as displayed in the portlets
- Name – Internal name for the table column; should be unique within the same table
- Data Type – The type and maximum value/length allowed for data stored in the table column
- Unique Key – Indicates whether the table column is used alone or along with other columns to uniquely identify a row of data in the table; used for validating rows entered into the table

Step 4 Click **Save**.add

Step 5 In the Content Data tab, click **Add** and enter required values. The values entered should conform to the data type and unique key restrictions specified in the Content Definition.

Step 6 In the Permissions tab, click **Add Permission** and update permission details.
Content access permissions can be controlled in a way similar to the portlets on the Permissions tab (see the Configuring Portlet Permissions, on page 265).

For users to view the content in their portal pages, read permissions to both the content definition and data are required.

Table 69: Configuration table for Custom Content

Field	Description
Display Name	The name to be displayed as the name of the table; free-format
Name	The internal name for the table; can contain only letters, numbers, and the underscore character (no spaces)
Content Group	The group in which the custom content table is located
Description	Optional documentation on the custom content table

Creating and Configuring JSR Portlets

Portlets developed using APIs which meet the Java Portlet Specification (JSR168, JSR286) standards may be integrated into Portal management solution. These will appear in Service Catalog as “Third-Party Portlets”. Vendor-specific implementations may include extensions to the approved APIs; these may not be supported.

JSR portlets can also be developed using Service Catalog REST APIs for processing and displaying Service Catalog entities. The information about these REST APIs, as well as the guidelines for developing JSR portlets to be used within the portal management solution can be found in the Cisco Prime Service Catalog Adapter Integration Guide.

Process for Configuring JSR Portlets

1. Deploy JSR Portlets. See Deploying JSR Portlets, on page 270.
2. Add JSR Portlets to the Portal. See Adding JSR Portlets to the Portal, on page 272.

Deploying JSR Portlets

Before You Begin

- Assemble the portlets with Pluto-specific information for deployment. For instructions, see Deployment Descriptor.
- On the JBoss 7 application server, the tag for renderSingleLine in **pluto.tld** must be commented out. For instructions, see Dependencies.
- The portlet WAR file should include the file: **jboss-deployment-structure.xml**, located under the WEB-INF folder in the portlet WAR file. For instructions, see Dependencies.
- (Optional) If the nsAPI java client is used in the portlets, the related Service Catalog and third-party libraries need to be included in the application package. For more information, see nsAPI Java Client.

Deployment Descriptor

The portal front-end uses Apache Pluto 1.1 libraries for the portal framework. To deploy a JSR portlet into Service Catalog, the portlets must be assembled with Pluto-specific information for deployment. Specifically, a servlet and servlet mapping are added to the deployment descriptor (web.xml). This servlet (org.apache.pluto.container.driver.PortletServlet) is used to dispatch portlet requests to the portlet application. For more detailed information, see the deployment instructions in the Apache web site.

The following is a sample web.xml file for a portlet called “CategoryPortlet”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
                         "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
  <display-name>CategoryPortlet</display-name>
  <description>Category Portlet</description>
<servlet>
  <servlet-name>CategoryPortlet</servlet-name>
  <servlet-class>org.apache.pluto.container.driver.PortletServlet</servlet-class>
  <init-param>
  <param-name>portlet-name</param-name>
  <param-value>CategoryPortlet</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>CategoryPortlet</servlet-name>
  <url-pattern>/PlutoInvoker/CategoryPortlet</url-pattern>
</servlet-mapping>
</web-app>
```

Dependencies

On the JBoss 7 application server, the tag for “renderSingleLine” in pluto.tld should be commented out.

```
<!--
<tag>
    <name>renderSingleLine</name>
    <tagclass>org.apache.pluto.driver.tags.PortletRenderSingleLineTag</tagclass>
    <bodycontent>empty</bodycontent>
</tag>
-->
```

In addition, the portlet WAR file should include a file named “jboss-deployment-structure.xml”, located under the WEB-INF folder in the portlet WAR file, to describe the dependencies on the JBoss modules.

Here is the sample content for the XML file:

```
<jboss-deployment-structure>
  <deployment>
     <dependencies>
          <module name="javax.portlet" slot="main" export="true"/>
          <module name="org.apache.pluto.container.om" export="true"/>
          <module name="org.apache.pluto.container.driver" export="true"/>
          <module name="org.apache.pluto.tags" export="true"/>
      </dependencies>
  </deployment>
</jboss-deployment-structure>
```

nsAPI Java Client

If the nsAPI java client is used in the portlets, the related Service Catalog and third-party libraries need to be included in the application package. A complete list of those dependent libraries and their locations can be found in the Cisco Prime Service Catalog Adapter Integration Guide.

Procedure

- JBoss

JBoss

Step 1 Create a subdirectory with the portlet name under the " <JBOSS_HOME>\requestcenterserver\deployments " folder, for example: <JBOSS_HOME>\standalone\deployments\<portlet_name> .

Step 2 Extract the portlet WAR file into the <portlet_name> directory that you just created.

Step 3 If the deployment descriptor has not been configured for the Apache Pluto portal server, modify the **web.xml** file accordingly. See Deployment Descriptor.

Step 4 If the server is already running, create a text file named <portlet_name>.dodeploy.

Adding JSR Portlets to the Portal

All successfully deployed JSR portlets will show up automatically on the JSR Portlets tab in the Portal Designer module. The portlets are placed into the "Third-Party Portlets" folder and their statuses are initially set to Inactive.

As with content portlets, access permissions are applied to the JSR portlets.

Step 1 Choose **Portal Designer** > **JSR Portlets**.

Step 2 Set the portlet Status to **Active**.

Step 3 Modify the author and description as necessary for better documentation.

Step 4 Add appropriate keywords to the portlet to facilitate portlet search.

Step 5 Save the settings.

Step 6 On the Permissions subtab, grant the read permission to the appropriate entities.

Step 7 Edit the desired portal page and add the JSR portlet to the page, just as you would for content portlets.

Removing JSR Portlets from the Portal

Before a JSR portlet is made obsolete and permanently removed from the application server, all dependencies and associations to the portlet should be removed. To do this, you should remove the portlet from all portal pages that contain the portlet, and delete the portlet from the JSR Portlets tab. This would allow permissions and subscriptions to be dropped for the portlet. Finally the portlet can be undeployed from the application server.

Migrating JSR Portlets between Portals

The import/export of JSR portlets configurations is not supported at this time. The steps for entering general information and permissions for JSR portlets need to be repeated manually when the portlets are deployed to a Service Catalog environment for the first time.

Customizing Reserved Portlets

Service Catalog includes number of Reserved Portlets—the Search, Order, Approvals, Account, Agreement, Billing Rates, Charge History, Search, and Policy Alert portlets are listed on the Reserved Portlets folder on Portal Designer > Portlets page.

You can modify the filtering parameters of the following reserved portlets:

- Search Portlet
- Order Status Portlet
- Approvals Portlet

Search Portlet

The Search portlet functions the same as the “Search for Services” function (“Search for services containing:” field) in My Services. See the My Services Online Help for more information.

To perform a Search, enter search criteria in the text box and click on the Search icon

. Clicking on the

icon clears the search box. Wildcards (*) are supported and perform as a case-insensitive search. A list of services matching the search criteria appears below where you can click on a service name to pop up the Service Overview/Summary page, or click **Order** to pop up the Service Order page.

Order Status Portlet

The Order Status portlet, used to track and view orders, is similar to the Requisitions tab in My Services. The Order Status portlet displays a list of requisitions filtered by requisition type and requisition status.

Step 1 From the left drop-down menu, choose the type of requisition to view**—Ordered for Myself, Ordered for Others,** or **Ordered for my unit**.

Step 2 In the right drop-down menu, choose a requisition status**—Preparation, Ordered, Ongoing, Cancelled, Closed, Rejected,** or **All**.

Step 3 Click the button to filter the requisition list based on your selections in the Steps above.

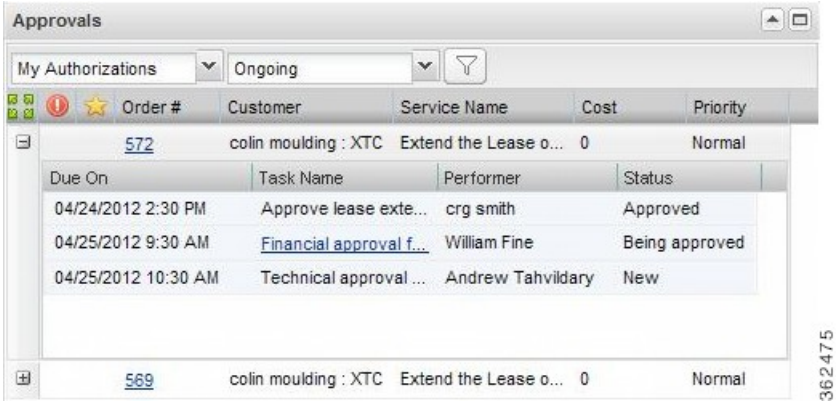
Note Filter selections are remembered for the Order Status portlet for each user on every page the portlet is added, even when filter selections are changed in the Requisitions tab of My Services.

Note The "Ordered" requisition status only appears in the requisition list if the "Submit, Approve and Review Asynchronously" setting is turned on in the Common section of Administration > Settings > Customizations. See the Site Administration chapter of the Cisco Prime Service Catalog Administration and Operations Guide for more information.

Approvals Portlet

The Approvals portlet, used to track and view authorizations, is similar to the Authorizations tab in My Services. See the *My Services Online Help* for more information.

Figure 23: Approvals Portlet



The Approvals portlet displays a list of authorizations filtered by authorization type and authorization status.

Step 1 From the left drop-down menu, choose the type of authorizations to view—**My Authorizations, My Assigned and Unassigned,** or **Authorizations for Others**.

Step 2 In the right drop-down menu, choose an authorization status—**Ongoing, Approved, Rejected, Reviewed, Cancelled,** or **All**.

Step 3 Click the button to filter the authorization list based on your selections in the Steps above.

Note Filter selections are remembered for the Approvals portlet for each user on every page the portlet is added, even when filter selections are changed in the Authorizations tab of My Services.

Note The “Ongoing” status appears as “Being Approved” or “Under Review” in the authorizations list.

For more information see Understanding Service Items Policies, on page 27.

Integrating Service Catalog Entities in Portlets

The portal management solution allows you to integrate views of Service Catalog entities (objects) into portlets. Such reference data cover the following types of objects:

- Core Entities represent the definitions of Service Catalog entities, such as categories, and services, as well as directory data (people and organizations) and actual transactional data on tasks and requisitions.
- HTML/JavaScript reference data refers to portlets defined in Portal Designer of the corresponding type.
- Service Items defined in the Service Item Manager module, used to track corporate assets that have been ordered or updated via service requests.
- Standards defined in the Service Item Manager module, used to enforce data entry rules or otherwise standardize the configuration of service items or other orderable assets.

Reference data serves as a quick reference to the definitions of Service Catalog entities. A list of the attributes in each object that comprises the reference data is given in the “Content Definition” tab displayed when that object is chosen from the Reference Data list panel, as shown in the sample below.

All attributes listed in the “Definition” section are available for inclusion in the portlet views. Detailed descriptions of these columns are given in the sections below.

Content Definition

The Portal Designer > Reference Data > Content Definition tab lists details on each of the entities that can be included in a portlet. This subtab is read-only, allowing portal designers to review entity definitions before including them in a portlet.

Field	Description
Display Name	The name of the object displayed in Portal Designer.
Name	The system name of the entity as stored in the portlet content metadata tables.
Content Group	The categorization of content type; for core entities, the entity type (definitional, directory, or transactional); for service items and standards, the service item group or standards group, respectively.
Description	A description of the entity.

Field	Description
Definition	
Display Name	The name of the attributes that comprise the entity. This is the column name displayed by Portal Designer and within portlets.
Name	The name of the database column containing the attribute data.
Data Type	The data type of the attribute/column.

In addition to the attributes which comprise the entity, the content definition includes one or more “URL” as the last attribute.

In a portlet, the URL attribute generates a clickable link which takes the user to the corresponding application module to bring up the entity details or the actionable view of the entity on a popup page. The user interface is the same as the one that user would see by navigating through the search views in those modules. The only difference is that the user stays in the portal and does not lose the context once the review/action is completed for the entity and the popup page is closed.

Users must have the required RBAC capabilities to navigate to other modules via the URL links; otherwise, an insufficient permission error appears.

Core Entities

The Portal Designer > Reference Data > Core Entities tab allow portal designers to expose information on application transactional, definitional, and directory data to portal users. In general, the attributes available correspond to the fields displayed on the corresponding user interfaces in the application modules.

Categories

Categories are used to group services in the service catalog for presentation to end users who may wish to browse or order those services.

Table 70: Categories

Column (Display Name)	Description
Category ID	Internal ID of the category
Category Name	Name of the category
Description	Description of the category
isRoot	Whether the category is a root category, that is, “Consumer Services” or “Service Offerings”
TopDescription Enabled	Whether the top section of category details is enabled in the category presentation

Column (Display Name)	Description
Top Description	HTML defined in the top section of category details in the category presentation
TopDescription URL	URL defined in the top section of category details in the category presentation
MiddleDescription Enabled	Whether the middle section of category details is enabled in the category presentation
Middle Description	HTML defined in the middle section of category details in the category presentation
MiddleDescription URL	URL defined in the middle section of category details in the category presentation
BottomDescription Enabled	Whether the bottom section of category details is enabled in the category presentation
Bottom Description	HTML defined in the bottom section of category details in the category presentation
BottomDescription URL	URL defined in the bottom section of category details in the category presentation
Category Image	Relative URL for the category image within Request Center.war
CatalogType ID	Internal ID of the category type: 1-Consumer Service category, 2-Service Offering category
Description URL	(Not Used)
Category URL (My Services)	Relative URL link for accessing the category in the My Services – Category Overview tab
Service ID	Internal ID of the service
Service Name	Name of the service
Description	Description of the service
Top Description	HTML defined in the Overview section of service details in the service presentation, shown only when the display is set to Show
Middle Description	HTML defined in the More Details section of service details in the service presentation, shown only when the display is set to Show

Column (Display Name)	Description
Bottom Description	HTML defined in the Service Form section of service details in the service presentation, shown only when the display is set to Show
RevisionNumber	Internal version number of the service
Price Description	Description of how the service is priced, shown only when the Pricing Summary display is set to “Display both cost and price” or “Display only price”
Pricing Scheme	Pricing scheme of the service, shown only when the Pricing Summary display is set to “Display both cost and price” or “Display only price”
IsBundle	Whether the service is a bundle
IsOrderable	Whether the service is orderable
IsReportable	Whether the service is reportable in the Advanced Reporting module
Service Level Description	Description of the service level as defined in the service general information
Status	Status of the service; possible values are Active and Inactive
Status ID	Internal ID for the status of the service; possible values are: 1 (Active),2 (Inactive)
Expected Duration	Expected duration of the service in hours
Expected Duration Units	Units of measure to be used when displaying the service; possible values are “Business Days” and “Hours”
Price	Price of the service, shown only when the Pricing Summary display is set to “Display both cost and price” or “Display only price”
Can Start Later	Whether future delivery is allowed for the service
Date Quality ID	Forecasting method defined for the service; possible values are: 2 (Estimate Due Date from task durations) 3 (Approximate Due Date using Standard Duration) 4 (Do not forecast Due Date)

Column (Display Name)	Description
Service Image	Relative URL link for the service image in the form of servlet reference within RequestCenter.war
Service URL	Relative URL link for accessing the service in the My Services Service Overview page
Service Order URL	Relative URL link for accessing the service in the My Services Service Order page
Ordering Mode	Ordering mode defined for the service. Possible values are • Add review enabled • Add review disabled • 1-Click • For more information, see Creating a Service
Compute Price	The value defined for computing the price of a service. The possible values are True or False.

Services

Table 71: Services

Column (Display Name)	Description
Service ID	Internal ID of the service
Service Name	Name of the service
Description	Description of the service
Top Description	HTML defined in the Overview section of service details in the service presentation, shown only when the display is set to Show
Middle Description	HTML defined in the More Details section of service details in the service presentation, shown only when the display is set to Show
Bottom Description	HTML defined in the Service Form section of service details in the service presentation, shown only when the display is set to Show
RevisionNumber	Internal version number of the service

Column (Display Name)	Description
Price Description	Description of how the service is priced, shown only when the Pricing Summary display is set to “Display both cost and price” or “Display only price”
Pricing Scheme	Pricing scheme of the service, shown only when the Pricing Summary display is set to “Display both cost and price” or “Display only price”
IsBundle	Whether the service is a bundle
IsOrderable	Whether the service is orderable
IsReportable	Whether the service is reportable in the Advanced Reporting module
Service Level Description	Description of the service level as defined in the service general information
Status	Status of the service; possible values are Active and Inactive
Status ID	Internal ID for the status of the service; possible values are: 1 (Active), 2 (Inactive)
Expected Duration	Expected duration of the service in hours
Expected Duration Units	Units of measure to be used when displaying the service; possible values are “Business Days” and “Hours”
Price	Price of the service, shown only when the Pricing Summary display is set to “Display both cost and price” or “Display only price”
Can Start Later	Whether future delivery is allowed for the service
Date Quality ID	Forecasting method defined for the service; possible values are: 2 (Estimate Due Date from task durations) 3 (Approximate Due Date using Standard Duration) 4 (Do not forecast Due Date)
Service Image	Relative URL link for the service image in the form of servlet reference within RequestCenter.war
Service URL	Relative URL link for accessing the service in the My Services Service Overview page

Column (Display Name)	Description
Service Order URL	Relative URL link for accessing the service in the My Services Service Order page
Ordering Mode	Ordering mode defined for the service. Possible values are • Add review enabled • Add review disabled • 1-Click
Compute Price	The value defined for computing the price of a service. The possible values are True or False.

Agents

Agents are used by Service Link, the integration hub of Service Catalog, to provide an interface between Service Catalog service requests and third-party systems such as help desks, inventory control systems, purchasing systems, or other external applications.

Column (Display Name)	Description
Agent ID	Internal ID of the agent
Agent Name	Name of the agent
Description	Description of the agent
Status ID	Internal ID of the agent status; possible values are: 1 (Active), 2 (Inactive)
Action	Action to be performed by the agent, as seen in the task general information in Service Designer
Context Type ID	Internal ID for the context of external task; possible values are: 1 (Service Task), 2 (Service Item Task)
Inbound Adapter Name	Name of the adapter used for inbound action
Outbound Adapter Name	Name of the adapter used for outbound action
InboundTransformation	Name of the transformation used for inbound document
OutboundTransformation	Name of the transformation used for outbound document
Failed email	Name of the email notification used for failure

Column (Display Name)	Description
Status	Status of the agent; possible values are: Active, Inactive
Agent URL	Relative URL link for accessing the agent in the Service Link Manager Integration tab

Requisitions

Requisitions are the service requests that have been submitted by Service Catalog users.

Table 72: Requisitions

Column (Display Name)	Description
Requisition	Internal ID of the requisition
Name	Display name of the requisition; normally the first service included in the requisition
Initiator Id	PersonID of the initiator of the requisition
Customer Id	PersonID of the customer of the requisition
Expected Duration	(Not used)
Actual Duration	Actual duration, measured in hours, to complete the requisition
Due Date	Date on which the requisition fulfillment is expected to complete
Closed Date	Date on which the requisition was actually set to Closed status
Expected Cost	Price of the requisition
Status	Status of the requisition; possible values are: Ongoing, Closed, Rejected, Canceled, Delivery Canceled
Initiator	First and last name of the initiator of the requisition
Customer	First and last name of the customer of the requisition
Bill To	Name of the home organizational unit of the customer of the requisition
Submit Date	Date on which the requisition was submitted

Column (Display Name)	Description
Requisition URL	Relative URL link for accessing the requisition details page in My Services

Authorizations

Authorizations are any approvals or reviews required in conjunction with completing fulfillment of a service request. The columns cover those that are presented in the Authorization tab in My Services.

Table 73: Authorizations

Column (Display Name)	Description
Requisition	Requisition ID associated with the task
Total Price	Total price of the requisition associated with the task
Due On	Due date of the task
Task Name	Name of the task
Service Name	Name of the service associated with the authorization. When there are multiple services for the authorization, only the first service name is shown.
Customer	Name and home organizational unit of the customer for the associated requisition, presented in the format {FirstName LastName} : {OU Name}
Performer	First and last name of the performer of the task
Status	Status of the authorization task; possible values are: Under review, Being approved, Reviewed, Approved, Rejected
Priority	Priority of the authorization task; possible values are: High, Normal, Low
Authorization ID	Internal ID (Task ID or Activity ID) of the task
Authorization URL	Relative URL link for accessing the task data page in Service Manager

Tasks

Tasks are activities associated with a request, including reviews, authorizations and fulfillment tasks. The columns cover those that are presented in the Home tab of Service Manager.

Table 74: Tasks

Column (Display Name)	Description
Task Id	Internal ID of the task (aka Activity ID)
Task Name	Name of the task
Requisition	Requisition ID associated with the task
Due Date	Due date of the authorization task
Service Name	Name of the service associated with the task
Initiator	First and last names of the initiator of the requisition associated with the task
Customer OU	Name of the home organizational unit of the customer for the associated requisition
Customer Name	First and last names of the initiator of the requisition associated with the task
Performer	First and last names of the performer of the task, if the task is assigned to a person
Queue	Name of the queue assigned to perform the task, if the task is assigned to a queue
Status	Status of the task; possible values are: New, Ongoing, Under review, Being approved, Completed, Reviewed, Approved, Rejected, Skipped, Cancelled, Scheduled, Review Submitted, Approval Submitted
Scheduled Start Date	Scheduled start date of the task
Effort	Effort estimated for the task
Task URL	Relative URL link for accessing the task data page in Service Manager

Organizational Units

Organizations are the business units and service teams into which users are organized.

Table 75: Organization Unit

Column (Display Name)	Description
OrganizationUnit Name	Name of the organizational unit

Column (Display Name)	Description
Description	Description of the organizational unit
Organizational Unit ID	Internal ID of the organizational unit
Parent ID	Internal ID of the parent organizational unit
Parent Name	Name of the parent organizational unit
Organizational Unit Type ID	Internal ID of the organizational unit type; possible values are: 1 (Business Unit), 2 (Service Team)
Status ID	Internal ID of the status of the organizational unit; possible values are: 1 (Active), 2 (Inactive)
Status	Status of the organizational unit; possible values are: Active, Inactive
Manager ID	Person ID of the person assigned to the organizational unit manager functional position
Manager Name	First and last names of the person assigned to the organizational unit manager functional position
isBillable	Whether the organizational unit is marked as billable
Organizational Unit URL	Relative URL link for accessing the organizational unit general information page in Organization Designer

Persons

Persons are individual users as defined in Organization Designer.

Table 76: Personas

Column (Display Name)	Description
Person ID	Internal ID of the person
First Name	First name of the person
Last Name	Last name of the person
Email	Email address of the person
HomeOrganizationalUnit ID	Internal ID of the home organizational unit of the person
HomeOrganizationalUnit Name	Name of the home organizational unit of the person

Column (Display Name)	Description
TimeZone ID	Internal ID of the time zone of the person
TimeZone Name	Name of the time zone of the person
Login Name	Login name of the person
Birth Date	Birth date of the person
Hire Date	Hire date of the person
Title	Title of the person
Employee Code	Employee code of the person
Locale ID	Internal ID of the locale of the person
Language Code	Internal ID of the preferred language for the person
Language Name	Preferred language for the person
Supervisor ID	Person ID of the supervisor of the person
Supervisor Name	Name of the supervisor of the person
Status	Status of the person; possible values are: Active, Inactive
Person URL	Relative URL link for accessing the person general information page in Organization Designer

Groups

Groups are a user-defined grouping of OUs or people that can be used in the assignment of work, roles and permissions.

Table 77: Groups

Column (Display Name)	Description
Group ID	Internal ID of the group
Group Name	Name of the group
Description	Description of the group
Status ID	Internal ID of the status of the group; possible values are: 1 (Active), 2 (Inactive)

Column (Display Name)	Description
Status	Status of the group; possible values are: Active, Inactive
Parent ID	Internal ID of the parent of the group
Parent Name	Name of the parent of the group
Group URL	Relative URL link for accessing the group general information page in Organization Designer

HTML/JavaScripts

The HTML/JavaScripts objects are those HTML/Java script portlets that have been designed in Portal Designer.

Service Items

Both system- and user-defined service items are available for display in portlets. For user-defined service items, the attribute names and data types correspond to those defined at your site—see your service catalog design team for more information.

Standards

Both system- and user-defined standards are available for display in portlets. For user-defined standards, the attribute names and data types correspond to those defined at your site—see your service catalog design team for more information.

Configuring Portal Pages

Once a portlet has been defined, made active, and made available to users via the appropriate permissions, it can be incorporated into portal pages. Two approaches are available for configuring portal pages:

- The portal designer can preconfigure the page by specifying its layout characteristics and including portlets as appropriate.
- Portal end users can dynamically incorporate portlets for which they have permission into their own pages and optionally save the portal pages.

Creating a Portal Page

Step 1 Choose **Portal Designer** > **Portal Pages**.

Step 2 Click **Actions** drop-down arrow and choose the appropriate option to create both portal pages and page groups. All portal pages must be in a Portal Page Group.

Page groups serve as containers of portal pages for easier navigation and control of access permissions in Portal Designer. Only users who have read permissions to those page groups can see them in the Service Portal module.

Service Catalog includes two preconfigured page groups, both of which are displayed in the Service Portal module:

- System – This portal page group is reserved for site-wide information. The **Site Homepage** is located in this page group.
- My Workspace – This portal page group is accessible by all users and is available for users to place their portal pages.

Step 3 Click **Add**.

Step 4 Depending on your requirements, configure the portal page by performing the following actions:

- Modifying Page Configuration, on page 288
- Adding Portlets to the Portal Page, on page 291
- Granting Portal Page Permissions, on page 291
- Configuring Subscribed Users, on page 292

Modifying Page Configuration

After you create a Portal Page, use the General tab to view or modify the page configuration. The page group for a portal page cannot be modified once it has been specified because of the permissions already associated with the group.

The General information determines the overall look-and-feel of the page, including its color scheme (“Theme”) and layout.

Step 1 Choose **Portal Designer** > **Portal Pages** > **General**.

Step 2 Configure fields to specify general information about a portal page as summarized in the table below.

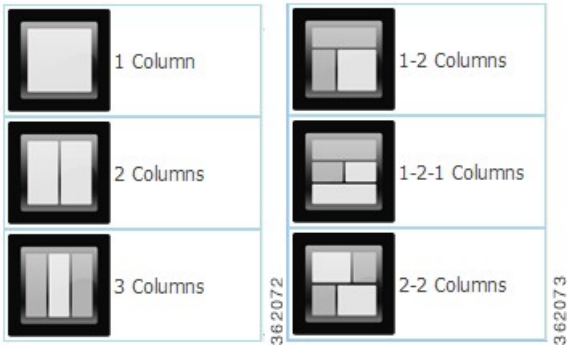
Table 78: Portal Page General Tab Field Descriptions

Field	Description
Status	The current status of the portal page; values are Active and Inactive. All portal pages are created with an “Active” status. When the page status is set to “Inactive”, the portal page is hidden from Service Catalog. Users who currently subscribe to the page will remain in the subscription record until the page is deleted. If an inactive page is still marked as the landing page for a person or an organizational unit, the setting is ignored, and users will land on their organizational unit homepage (if one is defined) or the Site Homepage instead when they first navigate to Service Portal.Catalog
Name	The name assigned to the page.
Theme	The default theme with which the portal page is displayed. The Portal Manager solution is distributed with a set of “Themes”, color schemes, and styles. Portal pages are set to use the “Gray” theme by default. They can be configured to use other preconfigured themes both by portal designers and by portal users who have the “Manage Portal Page Theme” capability. For more information on themes and the styles used by the Portal, see the Cisco Prime Service Catalog Administration and Operations Guide .
Page Group	The page group to which the portal page was assigned.
Layout	The layout for the portal page. Details on available layouts are given below.
Author	A text field containing the name of the author or other appropriate comments about the author; default value is the name of the user who created the portal page.
Make this page public	Making a portal page public makes it visible to other users. Users having appropriate permissions are able to subscribe to the page.

Step 3 Modify layout configurations.

The portal page can be divided into sections and columns either vertically or a combination of vertically and horizontally. Available page layout formats are summarized below:

Figure 24: Page Layout



Additional properties of the Layout Configuration are inherited from the portlet's definition, and can be overridden on a page-by-page basis:

Table 79: Configuration Table for Layout Configurations

Field	Description
Section 0 Column <n> Width	For each column in the Layout, the user can specify the percentage of the browser width that the column should take up. The percentages should not exceed 100 percent. The number of sections available varies with the layout chosen.
Portlet Borders	True if each portlet should have a border around it; false otherwise.
Portlet Headers	True if the grid column headers should be displayed; false otherwise.

Adding Portlets to the Portal Page

Use the Portlets subtab to include portlets on a portal page, configure their appearance, change the portlet configuration on a page, or remove a portlet from a page.

Step 1 Choose **Portal Designer** > **Portal Pages** > **Portlets**.

Step 2 Click **Add Portlets to Page.**

Step 3 If desired, you can filter the portlets displayed by entering a keyword and clicking **Search**.

Step 4 Expand the portlet groups displayed, until you find the portlets of interest. Highlight the name of a portlet to display a summary/description in the right-hand pane. Select the check box to the left of the portlet name to include the portlet on the page.

Step 5 Click **Add** to add the chosen portlets to the page.

When a portlet is added to a page, it is placed into the first section by default and set to display last on the page. The Section, Row and Column information in the grid indicates the location of the portlet on the page. The values in these fields cannot be modified directly but designers can change the position of a portlet by highlighting the portlet and clicking the “Move Up” or “Move Down” buttons.

If the portlet position needs to be substantially rearranged, designers may choose to do so in the portal front-end by adding the page to their portals, and using the mouse controls to drag the portlets to the desired location on the page.

To place a portlet into the second section or third section of a page with multi section layout (for example, 1-2, 1-2-1 or 2-2 columns), you need to edit the page in the portal front-end, use the mouse to drag the portlet to the bottom of the page and drop it when the second section outline appears. The third section, if available for the page layout, shows up only when one or more portlets have been placed into the second section.

Many of the properties shown are inherited from the portlet definition. Some of these (Name, Label, Type, Group) can be changed only by changing the portlet’s configuration. Click on the Name to go to the Portlets tab to modify these properties. The remaining inherited properties can be overridden on a page-by-page basis.

Granting Portal Page Permissions

The user who creates a portal page group or a portal page is automatically granted all access permissions to the object. For portal page groups, apart from the read and write permissions, the following permissions are also granted to the user:

- Read all pages in the group – Allows the user to view all the pages in the page group in Portal Designer. Also allows the user to subscribe to all the public pages in the page group in the portal front-end.
- Write all pages in the group – Allows the user to edit the settings and definition of all the pages in the page group in Portal Designer. Also allows the user to enter edit mode of the pages in the portal front-end.

Users without these permissions can only access individual pages for which they have read/write permissions. As with portlets, the Permissions subtabs for portal page group and page allow designers to designate which users should be able to access the portal object and the type of access to grant.

Step 1 Choose **Portal Designer** > **Portal Pages** > **Permissions**

Step 2 Click **Add Permission**.

Step 3 To configure permissions, see the Configuring Portlet Permissions, on page 265 for more information.

If you have upgraded from a previous release to the current release, the “Anyone” role is assigned the following permissions automatically:

- Read permission on System Page Group
- Read/Write permission on "My Workspace" Page Group
- "Read" permission on "Site Home Page" Portal Page

You must further remove the permissions from “Anyone” role and assign it to other roles as needed.

Configuring Site Homepage

The Site Homepage is automatically provided within the System portal page group. Users with the Site Administrator role are granted read and write permissions to this portal page in Portal Designer. They can edit the page or grant access to the page to other users so that the page can be configured to include site-wide information that is of interest to the portal users.

The read permission to the page is granted to “Anyone” roles. The portal page also serves as the landing page for a portal user when the user’s home organizational unit does not have a default landing page defined and the user has not set his/her own landing page preference.

Configuring Subscribed Users

The Subscribed Users subtab provides a read-only view of portal users who have included the current page in their portal. Designers cannot remove user subscriptions but can prohibit users from accessing the page by setting the status of the page to “Inactive”, or by removing the read access permissions.

Configuring Global Settings for Portlets and Portals

The **Portal Designer** > **Portal Settings** tab allows designers to specify global data and settings for use in all portlets and portal pages.

The following table describes the available options:

Table 80: Portal Setting Options Table

Option	Description
Common Settings	

Option	Description
	Site-wide settings for portal operations Common settings establish parameters for portal operations. The default settings are the recommended configurations. Changes to higher values may affect the application performance. • Maximum Number of Pages Created in Service Portal – The maximum number of portal pages that you can create a pagegroup from Service Portal. The highest value allowed for this setting is 10. • Maximum Number of Portlets on a Page – The maximum number of portlets that can be included on a portal page. The default setting is 6 and applies to all portal users. The highest value allowed for this setting is 10. • Maximum Number of Grid Portlets on a Page – The maximum number of grid portlets that can be included on a portal page. The default setting is 4 and applies to all portal users. The highest value allowed for this setting is 6. • nsAPI Page Size for Transactional Data – The default number of records returned by portlets and Service Catalog API clients for Requisitions, Requisition Entries, Authorizations, and Tasks (when the page limit is not specified in the API call). The default setting is 10. The highest value allowed for this setting is 50. • nsAPI Page Size for Directory Data – The default number of records returned by portlets and Service Catalog API clients for People, OUs, Groups, and Accounts (when the page limit is not specified in the API call). The default setting is 10. The highest value allowed for this setting is 50. • nsAPI Page Size for Service Item and Standard Data – The default number of records returned by portlets and Service Catalog API clients for Standards and Service Items (when the page limit is not specified in the API call). The default setting is 10. The highest value allowed for this setting is 50. • nsAPI Page Size for Definitional and Custom Data – The default number of records returned by portlets and Service Catalog API clients for

Option	Description
	Categories, Services, Offerings, Agents, Agreements, and User-defined Contents (when the page limit is not specified in the API call). The default setting is 10. The highest value allowed for this setting is 50.
Organizational Unit Settings	Default settings for individual organizational units. The Organizational Unit Settings page allows a home page to be configured for individual organizational units (OUs). A portal page so designated is always displayed in the portal for users who have the organizational unit as their Home OU. Users who have a home page defined for their Home OU will see two pages in My Workspace page group—the OU Homepage, and the Site Homepage. They can add/create more pages and choose to land on a different portal page by setting the desired page as the landing page for themselves. More information regarding the end user view of Service Portal can be found in the XREF.
Keywords	Keywords that can be associated with a portlet to help users find portlets when designing a portal page Keywords provide flexibility in portlet search when designers or portal users are looking for content to include in a portal page. Once defined in the Common Settings, keywords can be associated with a portlet on the Keyword page itself, or on the General Information subtab for portlets. Keywords used for portlets are distinct/disjoint from the Keywords used/associated with services (service definitions).

Option	Description
Authentication Settings	Authentication details for use with external sites to allow Single Sign-On (SSO) from enterprise login accounts to individual portlets, or to use group account credentials to automatically login to those sites. To connect a site that requires a login: **1** Choose **Portal Designer** > **Portal Settings** > **Authentication Settings**. **2** Click **Add External Site**. **3** Create an External Site profile specifying the site name and the login URL, identifying the fields on the login page where the user is expected to enter a user name and password; and specifying the authentication type. **4** If the same connection criteria can be used for all users accessing the site through the portal, specify global authentication criteria. **5** Design a portlet that has automatic login enabled and uses the external site. **6** The portlet can be included on a portal page. If global authentication arguments are specified, they will automatically be passed to the external site for authentication. If global authentication is not used, the user can enter site connection credential via the portal's Edit Password tab accessible in the portal's Edit Mode. **7** Add Authentication details using Table 81: Configuration Table for External Authentication

The following table describes the fields used to define external site authentication settings:

The following table describes the fields used to define external site authentication settings:

Table 81: Configuration Table for External Authentication

Field	Description
Site Name	The name of the site.
Login URL	The URL that identifies the site's login page.
Home Page URL	The URL that identifies the landing page of the application to be displayed.

Field	Description
UserID Field Name	The field on the page specified by the Login URL which contains the user's ID/user name.
Password Field Name	The field on the page specified by the Login URL which contains the user's password.
Authentication Type	URL, Form.GET, or Form.POST.
Other Arguments	Additional arguments that must be passed to the site. These are passed in a format depending on the authentication type. The typical format is arg1=value1&arg2=value2.
Use Global Authentication	A check box that indicates that the User ID and Password values specified below are used for all users to connect to the site.
UserID Value	The value of the User ID to be passed to the site for global authentication.
Password Value	The value of the Password to be passed to the site for global authentication.
Description	Free-format text which describes this site; documentation only.

Importing and Exporting Portal Content

Portal content and portlet definitions are developed in a Service Catalog instance, and the metadata underlying these objects is stored in the Content Management repository. You may want to back up an object definition to a source code control system or other file-based storage. You may want to develop content in a Development system, then transfer the content to a Test system for testing or validation, and then to a Production system for everyday use by end users.

To do this, you need to use the Import/Export facilities provided by Portal Designer.

Contents of an Exported File

The export file is an XML file in an industry-standard CIM (Common Information Model) compatible format, version 2.3.1. CIM is based on an object-oriented model and uses terminology adapted from Unified Modeling Language (UML).

Portlet Contents

What is included:

- Portlet general information, view and filter definitions, including HTML and JavaScript code
- Associated custom content definition and data
- Associated keywords
- Associated authentication settings

What is not included:

- All object permissions
- Definition and data for other content types (core entities, service items, standards)

Portal Contents

What is included:

- Portal page general information, settings and content definitions.
- Associated portal page groups
- Associated portlet definitions
- Associated custom content definition and data
- Associated keywords
- Associated authentication settings

What is not included:

- All object permissions
- Definition and data for other content types (core entities, service items, standards)

Exporting Portal Content and Portal Pages

You can export portal and non-JSR portlet definitions to the local file system.

Step 1 Open the Portal Designer.
Step 2 Depending on your requirement, open a portlet or a portal page.
Step 3 Click **Actions** > **Export**.
Step 4 Select the portal pages or portlets that want to export.
Step 5 Click **Export**.

Importing Portal Content

The ability to create portal objects through the import utility is still controlled by the corresponding capabilities and permissions for such actions. The user who performs the file import should be granted appropriate roles in order to successfully execute the import.

Before You Begin

Make sure that you have a valid XML file created using the Export Portlets and Export Portal Pages features.

Step 1 Open the Portal Designer.
Step 2 Depending on your requirement, open a portlet or a portal page.
Step 3 Click **Actions** > **Import** to import portal objects into the same environment in which they are exported, or another environment which does not contain those objects.
Step 4 Select the conflict resolution you want to choose:

- **Overwrite:** The import replaces the existing definitions of the objects with the same names in the environment with the definition contained in the XML file.
- **New Only:** The import fails if the portal object already exists.

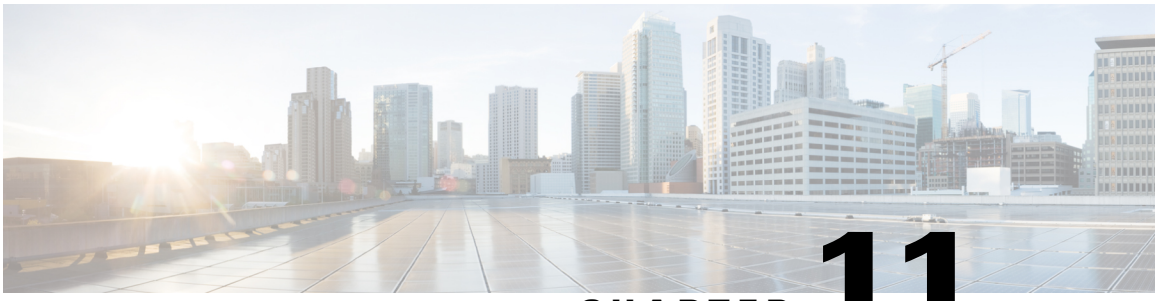
When the import is complete, a summary page appears. A detailed log is also available to show the names of the portal objects created/updated.

Troubleshooting Import Failures

An import may fail due to insufficient permissions.

- The user does not have “Manage Portlets” capability when importing a portal page that contains new portlets. In this case, the portlets will not be created, and the import of the portal page containing the new portlets will also fail.

- The user does not have “Manage Custom Content” capability when importing a portlet that is based on new custom content. In this case, the custom content table will not be created, and import of the portlet making use of the custom content will also fail.
- The user does not have write permission to the portal page group that is specified for a new portal page. In this case, the portal page will not be created.



CHAPTER 11

Localizing Service Catalog

This chapter contains the following topics:

Localizing Service Catalog

Overview

Cisco Prime Service Catalog is often deployed in a multi national corporation where you want the services to be available to users who have different preferred languages. A user may prefer to use the application in his preferred language, and hence the application will need to be localized. Localization is used for different purposes:

- A service designer could design, manage, and publish services; a translator can use the localization module to translate the services in a preferred language. After the translations are complete the localized content can be migrated from development to test to production environments using export/ import functionality in the localization module.
- The end user can choose a preferred language to view the application in the selected language.
- Prime Service Catalog localization module provides a method to localize its strings and enable them such that any user accessing the application will be able to do so in his preferred language.
- If you have not translated a particular string, then that string is displayed in English by default.

Note You can access the localization module if you have the roles of an administrator or a distributed service designer. For more information about roles to access localization module, see Assigning Role Capabilities section in Cisco Prime Service Catalog Administration and Operations Guide.

Localization Workflow

The localization workflow helps you understand the various steps required to translate the strings in Cisco Prime Service Catalog:

Step 1 The site administrator prepares the application for translation.

a) Provide access privileges for a service designer/translator to access localization module. For more information about adding localization management capability, see Accessing Localization Module, on page 303.

b) Add the requested language in the localization module. The translator will further use this language to translate the strings. For more information about adding a language to the application, see Adding a Language, on page 304.

c) Enable the language to end users such that any user accessing the application will be able to do so in the new language. For more information about enabling the language to end users, see Enabling the Language for End Users, on page 305.

Step 2 A service designer designs service forms. For more information about designing a service, see Setting Up Services, on page 6.

Step 3 A translator or localization expert translates the strings using the exported csv file to translate multiple strings uses the localization module to translate strings inline. For more information about translating various strings, see Translating Strings, on page 305. During translation, if the translator needs to know the resource ID of the string to be translated, enable a setting in the administrator module to find the corresponding product string. For more information about finding a resource ID, see Finding a Resource ID for a String, on page 304.

Note The exported CSV file must be edited only using open source spreadsheet editor - LibreOffice. Microsoft Excel must not be used for editing this CSV file as it corrupts the file. For more information on settings while editing these csv file, see Editing Exported CSV File with LibreOffice, on page 308.

Step 4 After translating the strings in the application, the translated content can be migrated from development to test to production environments. For more information about migrating translated content across various environments, see Migrating Translated Content, on page 309.

Step 5 An end user can now access the application in the preferred language. For more information about accessing the application in a preferred language, see Using the Application with your Preferred Language, on page 310.

The other tasks that you might want to perform on an ongoing basis are:

- As the strings are defined in the system, there is a possibility that some translations may become out of sync if the English string is changed. For more information about finding out of sync strings and updating them, see Verifying and Updating Out of Sync Strings , on page 309.
- You can activate or deactivate a language. For more information about prohibiting a user from accessing the application in a particular language or re enabling the language access, see Activating and Deactivating a Language, on page 310.
- If you want to stop support for a particular language you can delete that language. Ensure that you have deactivated a language before you delete it. For more information about deleting a language, see Deleting a Language, on page 310.
- You can configure the number of records you wish to view per page in the grid. For more information, see Configuring the Page View for Product Strings, on page 310.

Accessing Localization Module

You can access the localization module if you have the roles of an administrator or a distributed service designer. For more information about roles to access localization module, see Assigning Role Capabilities section in Cisco Prime Service Catalog 11.0 Administration and Operations Guide.

Translating Application Strings

This section describes the workflow to translate application strings:

- Supported Strings for Localization, on page 303
- Finding a Resource ID for a String, on page 304
- Adding a Language, on page 304
- Enabling the Language for End Users, on page 305
- Translating Strings, on page 305
- Verifying and Updating Out of Sync Strings , on page 309

Supported Strings for Localization

The following strings are supported for localization. Each string appears as a tab in the localization module:

- Product Strings: System-defined strings. These strings are created outside of the application, used in the application, and localized.
- Content Strings: Catalog and portal content defined by end user in the application. Content Strings include:
  - Keyword instances
  - Dictionary attributes for Active Form components
  - Page Group names for Portal
  - Category attributes and category extensions
  - Service attributes and Service extensions for a selected service
  - Service Item group name attribute
  - Standards, Standard group names and its attributes
  - nsAPI includes:

Services (/nsapi/definition/servicedefs)

Categories (/nsapi/definition/categories)

Service Items (/nsapi/serviceitems)

Standards (/nsapi/standards)

- Email Templates: Localized email templates allow users to view the notifications in their own languages. The namespaces that are referred in the email templates are not localized. Also once the namespace expressions are resolved and the values are available, they will not be localized, the namespace expressions are used as is.
- JavaScript Strings: User-defined text strings (labels or alerts) that are referenced in Service Designer scripts or custom JavaScript libraries or portlets. The custom script developers can make use of the JQuery library to refer to these JavaScript strings in the prescribed format to have the JavaScript strings localized, e.g., jQuery.i18n.prop('js4') ; where js4 is the ID of a JavaScript resource Standards (/nsapi/standards). During customization, if there are service forms that refer to the JavaScript resource IDs, the application calls the localized strings as per the end users configured language.

Finding a Resource ID for a String

Each string has a unique resource ID. The resource ID for a string makes it easier for a translator to search for the string during translation.To find the resource ID of a string to be translated, enable the option Show Resource String ID in Administration > Settings > Show Resource String ID. Content strings do not have a Resource ID. To translate content string, see Translating Content Strings, on page 304.

Translating Content Strings

Step 1 Identify the content string that needs translation.

Step 2 Ensure that the exported file contains the content string. To do so, during export navigate to the string as mentioned in Translating Content Strings, on page 306.

Step 3 In the exported file, find the content strings for the particular service and the dictionary or active form component that it belongs to.

Step 4 Update the file with translated entries for corresponding content strings and import.

Note Alternatively, if there are few content strings to be translated you can use the inline edit option. While translating certain strings like String Id 11915, the characters : \', ", >, <, or the new-line character, cannot be used in an expression. When translating via localization module (UI) use \' in place of single quote ('). When translating from a CSV use \\' in place of Single (').

Adding a Language

Cisco Prime Service Catalog ships with a few out-of-box languages to the user. To add a particular language that is not delivered out-of-box you must add the language in the Localization module.

When you add a language:

- A translator can access a language for translation. The added language appears in the **Select Language** list for localization. For more information, see Translating Strings, on page 305.
- An **Enable the Language for End Users** link is enabled. When you enable the language:
  - an existing user can choose the language he wants from his profile, to access the application

- the administrator can enable the language to all new users using **Administration > Settings > Application Locale**.

For more information, see Enabling the Language for End Users, on page 305.

Step 1 Choose**Localization** > **Settings** > **Language Settings**.

Step 2 Select a language from the Language drop down list.

Step 3 Enter a **variant** of the language if required. The variant of the language is displayed within brackets next to the language.

Note The code for any language is displayed next to the language in **Administration > Settings > Application Locale**.

For example, the parent language is French and this language has two variants namely French (Canada) and French (Switzerland). During translation if a user is using French (Canada) and the tool does not have a translation for the string in French (Canada) then the tool finds the corresponding translation in its parent language i.e., French. If the translation is not available in French, the tool replaces the string in English.

Step 4 Click **Add**.

Enabling the Language for End Users

After an administrator adds a language, the translator translates the strings in the application and the translated content is finalized and migrated to the production environment. The administrator can then enable the new language for all end users such that the end user accesses his/her application in his/her preferred language.

Procedure

Step 1 Choose **Localization** > **Settings** > **Language Setting**.

Step 2 Select the check box preceding to the language and click **Enable the Language for End Users**. This will enable the language in the languages list in **Administration > Lists > Language**.

Note Ensure that you have the **Manage List** capability to add a language to the Language List in the Administration module. For more information, see Capabilities for Administration section in Cisco Prime Service Catalog Administration and Operations Guide

Step 3 An administrator can now enable the updated language to all new users by enabling it in the Site Settings. For more information, see Application Locale Site Settings in Cisco Prime Service Catalog Administration and Operations Guide and the language is available in the language list of User's Profile.

Translating Strings

If you are a translator and need to localize the application in a particular language ensure that the language that you want the application to be localized is already added in the Localization module. For more information, see Adding a Language, on page 304. After you add a language you can localize strings inline or at bulk using the export/import option. During localization if you have localized some strings partially and the others are not localized, the unlocalized strings will be defaulted to the variant language or to English.

The strings that are supported for localization are listed in Supported Strings for Localization, on page 303. Each string appears as a tab in the Localization module.

Note To edit a translated string, double-click on the string to be updated, update the string, and click Save.

Translating Product Strings

Step 1 Choose**Localization** > **Product Strings**.

Step 2 In the **Select Language** drop down list, select the check box preceding to the language you choose to update and click **Apply**. The language you choose appears in the grid.

Note The grid supports a maximum of three languages.

Step 3 Highlight the row for the string to be updated and double click on the content string field in the <language> column.

Step 4 Update field and click **Save**.

Note After saving the changes, you must log out and log in again for the changes to take effect.

Translating Content Strings

Step 1 Choose **Localization** > **Content Strings**.

Step 2 In the **Select Language** drop down list, select the check box preceding to the language you choose to update and click **Apply**. The language you choose appears in the grid.

Step 3 In the **Entity** drop down list, select the pre catalog or portal content. Based on the entity you choose the corresponding fields are displayed. For example, if you choose **Active Form** component, the **Dictionaries** and **Attributes** fields are displayed and if you choose **Portal, Portal Groups** and **Portal Pages** fields are displayed.

Step 4 Select the required field attributes that needs to be translated.

Note The grid supports a maximum of three languages.

Step 5 Highlight the row for the string to be updated and double click on the content string field in the <language> column.

Step 6 Update field and click Save.

Note To edit a translated string, double-click on the string to be updated, update the string, and click **Save**.

Translating Email Templates

Step 1 Choose **Localization** > **Email Templates**.

Step 2 In the **Select Language** drop down list, select the check box preceding to the language you choose to update and click **Apply**. The language you choose appears in the grid.

Step 3 Highlight the row for the string to be updated and double click on the content string field in the <language> column, for name and subject attributes.

Step 4 Click **Done**.

Step 5 Return to the main page and click **Save**.

Translating JavaScript Strings

The product and content strings are updated in the application as soon as they are localized. JavaScript strings must be published to be made available in the application:

Step 1 Choose **Localization** > **JavaScript Strings**.

Step 2 Click **Add** in the JavaScript strings tab to add new strings dynamically.

Step 3 In the **Select Language** drop down list, select the check box preceding to the language you choose to update and click **Apply**.

Step 4 Highlight the row for the string to be updated and double click on the content string field in the <language> column, for name and subject attributes.

Step 5 In the **Publish** tab, select the check box preceding to the language you choose to publish and click Apply.

Step 6 Go back to JavaScript program code in service designer or portal designer and reference the resource ID defined or translated here. For more information, see Designing Portlets and Portals Using Portal Designer, on page 257

Translating Multiple Strings

If you are performing wide scale translation that requires you to translate large number of services, you can select a language, export files, translate strings, and import it back to the application. In the procedure below replace <String> with Product Strings, Content Strings, Email Templates, or JavaScript Strings accordingly.

Step 1 Choose **Localization**.

Step 2 Select the **<String>** tab that you want to update.

Step 3 Select the check box preceding to the language you choose to update in the **Select Language** tab and click **Apply**.

Step 4 Choose **Select Action > Export**.

Step 5 Select the check box against the language files you want to export.

Step 6 The files are displayed as a CSV file with its corresponding resource strings. Update the strings and save.

Step 7 Choose **Localization > < String> > Select Action > Import**.

Step 8 Browse the CSV file and click **Import**. The maximum size of the csv file to import into Prime Service Catalog should not exceed 2 MB.

Note You can also use the export or import functionality to move translated files from production environment to development environment and vice versa.

Note The exported CSV file must be edited only using open source spreadsheet editor - LibreOffice. Microsoft Excel must not be used for editing this CSV file as it corrupts the file. While editing the CSV file using LibreOffice Calc, use the settings as described in Editing Exported CSV File with LibreOffice, on page 308.

Editing Exported CSV File with LibreOffice

While editing the CSV file using LibreOffice Calc, use the following settings:

- When opening the file, make sure that you have set:
  - Character set as **Unicode(UTF-8)**
  - Language as **Default-English(USA)**
  - Separator Option as **Comma**
- While saving the file, make sure that you have:
  - Set the File extension to .csv.
  - Used **Edit Filter Settings** and **Use Text CSV Format** options
- While exporting, make sure that you have:
  - Set Character set as **Unicode(UTF-8)**

Set Field delimiter is set to comma

- ◦ Set Text delimiter is set to double quotes
  ◦ Selected **Quote all text cells**

Migrating Translated Content

Localization module allows you to translate strings in a production environment, test it in a test environment, and then deploy it in production environment. The migration of translated content is done using the export/import option of the localization module. Unlike in other cases such as service design and portal design, migration of content is handled by the Catalog Deployer module. For more information about exporting or importing of translated content, see Translating Multiple Strings, on page 308.

Verifying and Updating Out of Sync Strings

As the strings are defined in the system, there is a possibility that some translations may become out of sync if the English string is changed. The application displays the user the entity/instance that is out of sync that requires user action. Also when the Export feature is exercised, the translator can get a detail report of how many strings are out of sync.

For example, a content string such as the name of a service is changed from Order a Desktop to Order a Laptop and there is a corresponding French translation that was done for the content string “Order a Desktop”. As soon as the name of the service changes the application highlights the language column header in red. The translator will quickly realize that there has been a change in the original language and the string needs to be updated in French language too.

The below figure displays how the strings are displayed.

Step 1 Choose **Localization** > **Content Strings** and select **Service** from the entity drop-down list.

Step 2 Select **Export** from Select Action drop-down list and select the entities and instances to export and click **Export**.

Step 3 Save the exported file as CSV format and open with Microsoft Excel.
The Exported file has a column with heading **Requires Review (Out of Sync)**. The translator can update the out of sync strings and import the file.

Managing Localization

This section describes the tasks that an administrator or a designer could perform:

- Activating and Deactivating a Language, on page 310
- Deleting a Language, on page 310
- Configuring the Page View for Product Strings, on page 310

Activating and Deactivating a Language

Any language that is added is activated by default. If you deactivate a language all users who have the language in their profile settings will be defaulted to English. You can select a deactivated language and activate it. When you reactivate a language, the users who had the deactivated language in their profile settings will be able to access the application in the language. You can reactivate/deactivate a language by selecting the language in Localization > Settings > Language Settings.

Deleting a Language

You must deactivate the language before you delete it. You cannot reactivate a deleted language. To delete a language, select a deactivated language in Localization > Settings > Language Settings and click Remove.

Note In Prime Service Catalog, you can only delete the custom languages and not the predefined languages available out of the box.

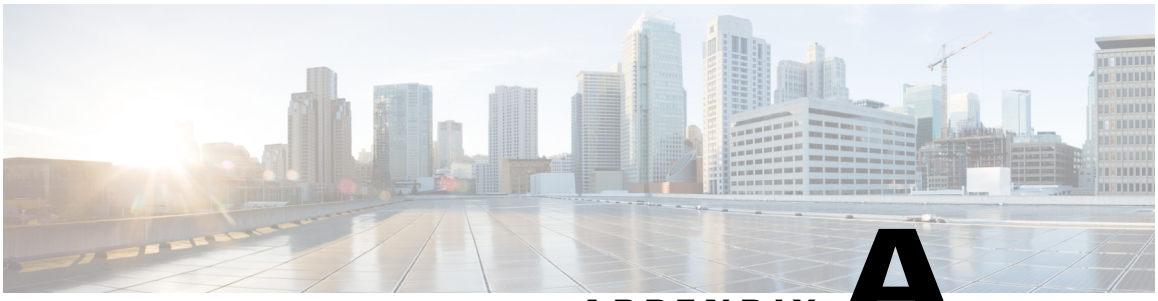
Configuring the Page View for Product Strings

You can configure the number of records you wish to view per page in the grid only for the product strings. The valid values for this setting are between 20 and 100 and are applicable for each page. To customize the number of records per page, choose Localization > Settings > Customization > Product Strings, and enter a value in the Setting Value field, and click Update.

Using the Application with your Preferred Language

All existing users can access localization feature by enabling the language in the Preferred Language list in the user's profile. Ensure that you have the languages in the Preferred Language list. For more information, see Enabling the Language for End Users, on page 305.

If you are an end-user and you want to choose a language to access the applications, choose Profile > Home > Preferred Language, select a language, and click Save.



APPENDIX A

Use Case of Designing a Service to Select Laptop

This appendix contains the following topics:

Use Case of Designing a Service to Select Laptop

The Laptop Selection example uses a grid service item dictionary and a data retrieval rule to demonstrate automatic retrieval of service item data when a field is chosen from a drop-down menu in a grid. The data retrieval rule also includes a Lookup Condition where a data instance is excluded from selection.

This example walks you through the steps.

Step 1 Create New Service Item Group and Service Item, in **Service Item Manager** > **Design Service Items**. For detailed procedure or creating group and service item, see Creating a Service Item Group, on page 22 and Creating Service Items, on page 23.

a) After the new service item is created and as it appears in the Service Item panel on the left under the Group, click **Add** on the bottom of the page to add the attributes to the Laptops Service Item, as shown below. Each attribute, or row, is a service item instance. The Display Name for each instance will become a column header for the grid.

b) Click **Save** after adding all attributes

Step 2 Create New Dictionary Group and Grid Dictionary in the **Dictionaries** component of the Service Designer module. For detailed procedure, see Creating a Dictionary Group, on page 49 and Creating a Dictionary, on page 49.

a) While creating new dictionary, in the Service Item field of the Add New Internal Dictionary section, enter Laptops to search for the Laptops service item you created, then click Laptops from the popup window that appears, as shown below.

b) In the resultant Dictionary window, enter **Laptops** for the Dictionary Name and Default Caption fields. The Default Caption will display as the title for the grid. Choose the dictionary group you just created from the Group Name drop-down menu. In the Use and “Show in Grid” columns, check only the five instances of the Laptops service item you created. By checking Use and “Show in Grid”, the instances will appear as columns in the grid. The Name field is already checked by default and cannot be changed.

c) Click **Save Dictionary**.

Step 3 Create New Form Group and Form.

a) Click the **Active Forms Components** component of the Service Designer module, and choose **New** > **Form Group**. The new form group appears in the Active Form Components tree to the left.
b) Choose **New** > **Active Form Component** > and enter the required details for the new form and save form.

Step 4 Add Grid Dictionary to Form and Edit Display Properties.

a) Click **Add Dictionaries** and in the Search field, enter **Laptops** to search for the Laptops dictionary you created. For the resultant Laptop dictionary, check the Name and "Display as Grid" columns. Checking "Display as Grid" makes the fields of the Laptops dictionary that were checked to "Show in Grid" display as columns in the grid.
b) Click **Add** to add the Laptops dictionary to the form. The Form Content tab of the Laptop Selection form appears with the Laptops dictionary checked for "Display as Grid". Note that it is dimmed here and cannot be changed. The only way to change it would be to remove it, and then add it again without checking "Display as Grid".
c) Click the **Display Properties** tab for the Laptop Selection form.
d) Check **Enable automatic retrieval of service item instance data**, as shown below. This option is disabled by default. This option enables the ability to automatically retrieve and pre fill data about an existing service item instance. For this example, leave the Grid Options to their default selections and values. In particular, you want the user to have Edit access control to be able to add grid rows.
e) Below the Laptops dictionary in the left pane, click the **Name** field to see its HTML Representation window. In the Input Type field, choose "**select (single)**" from the drop-down menu. An Input Type of "Select Single" renders the Name field as a drop-down menu from which the user can choose only a single value. Also, change the Columns value to **32** to make the Name column width smaller.

Step 5 Create New Data Retrieval Rule. Click the **Active Form Rules** tab for the Laptop Selection form.

a) Choose**New Rule** > **New Data Retrieval Rule**
b) In the wizard, do the following:

- ◦ Set the Rule Type set to **Distributing Rule** and leave the Query Type set to **Database Table Lookup**.
  ◦ Check **Form Load** as the triggering event, as shown below.
  ◦ Choose **Service Items** from the Datasource drop-down menu, and **Laptops** from the Table Name drop-down menu.
  ◦ To create a Lookup Condition where data instances of "HP" in the Brand_Name field are excluded from selection, choose **Brand_Name** from the Table Column Name drop-down menu and "**is not equal to**" in the Operator drop-down menu, and enter **HP** in the Literal Value field (enabling its radio button), as shown below. Click **OK**.
  ◦ Map the Name column of the Laptops service item to the Name field of the Laptops dictionary, choose them from the three drop-down menus and then click **OK**.

Note The "*" next to Laptops indicates that Laptops is a grid dictionary.

Step 6 Create New Service Group and New Service. Use the **Services** component of the Service Designer module.

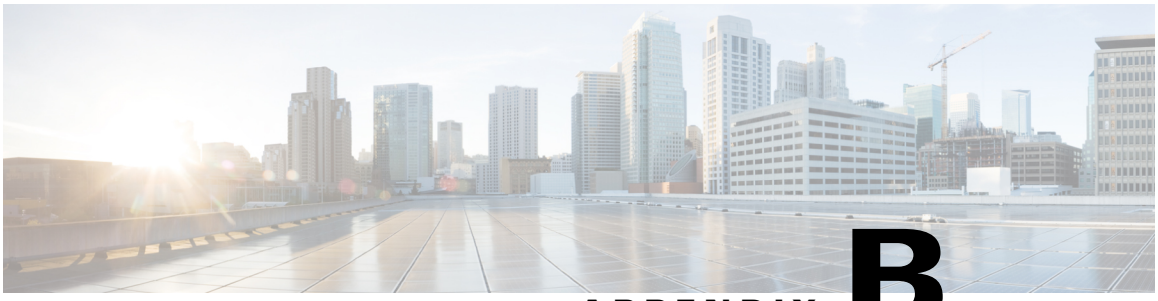
a) Choose **New** > **New Service Group**and enter the group name, choose the service team and enter the description.
b) Click **Add This Service Group**. The new group appears in the Services tree to the left.
c) Choose **New** > **New Service**. In the Name field of the resultant New Service window, enter **Laptop Upgrade** for this example. Enter "**Upgrade one or more laptops.**" for the description. Choose the service group you just created, **Dave Service Group**, from the Service Group drop-down menu.
d) Click **Add This Service**. The General Tab of the Laptop Upgrade service appears.

Step 7 Add Form to Service and View Service in My Services.

a) Click the **Form** tab of the Laptop Upgrade service that appeared after adding service in step 6 d.
b) Click **Add Forms** in the bottom left of the window. The Add Form popup window appears.
c) In the Search field, enter **Laptop Selection** to search for the Laptop Selection form you previously created.
d) Check the Laptop Selection form and then click **Add**.
e) The Laptop Selection form is added to the Laptop Upgrade service.
f) Choose the **My Services** module of Service Catalog.
g) Enter **Laptop Upgrade** in the "Search for Services containing:" field and then click **Search**. The Laptop Upgrade service appears.
h) Click **Order** to see the grid you created.
   The grid appears with the header of "Laptops" taken from the Default Name of the dictionary. The column header names are taken from the HTML Representation Label field of the dictionary fields.

   The Name field appears as a drop-down menu from which you can only choose a single value because you chose "select (single)" as the Input Type for the Name field. Click in the first row of the Name column and choose a name from the drop-down menu—the rest of the fields are automatically filled in. This happens because you checked "Enable automatic retrieval of service item instance data" for the form dictionary and mapped the Name column of the Laptops service item to the Name field of the Laptops dictionary.

Use Case of Designing a Service to Select Laptop



APPENDIX B

Namespaces

This appendix contains the following topics:

Namespaces

The Business Engine provides facilities for service designers and system administrators to customize the contents of emails and to dynamically configure the progress of a requisition through the approval and delivery process by specifying conditional workflows.

In order to implement this capability, Service Catalog provides a unified way to access the values of data elements, such as a request's initiator or the service form data related to a request. This unified way is referred as the Business Engine Namespace.

Namespace functionality increases the flexibility of the service design, by allowing the delivery plan to incorporate approvals or tasks that are executed conditionally, depending on the current value of a namespace variable, or to dynamically adjust the routing of a particular task. It also allows the content, subject, and addresses of emails to be dynamically adjusted.

Note The use of grid dictionary fields for Business Engine namespace is not supported.

Namespace is a term used to describe a set of valid names that address the data objects used within Service Catalog, exposing these objects to service designers. This allows designers to use these elements in these contexts:

- Within an email, to dynamically resolve the recipient, subject, or references within the email body.
- To conditionally execute reviews, authorizations, or tasks in a delivery plan.
- In expressions which determine the person or queue to which an authorization or delivery task is assigned.
- In task names.

Namespaces are hierarchical. Namespace names reflect the structure of the data which defines a service, and the data entered on the service form when that service is requested. The key to manipulating namespace variables is knowledge of the hierarchy which defines Service Catalog data and the contexts in which particular Namespace variables can be used.

Each element in the hierarchy is case-insensitive. The elements are separated by periods (“.”). The last element may also be referred to as a “property”. All elements before the last one are “nodes” in the hierarchy.

This chapter describes Namespace in detail, its standards and implementation.

References

When Namespaces are used in a textual context, as in an email template or an assignment expression, delimiters are required to differentiate the Namespace from surrounding text. This delimiter is the character “#” surrounding the namespace references.

EXAMPLES:

Table 82: References

For emails:	Dear #Customer.Firstname#, We are pleased to inform you that the service you requested has been approved...
For expressions:	CN=#Service.Data.Initiator_Information.First_Name#
For conditions:	ActualCost > 2000.00

Nodes

Every node in the hierarchy has a node type. The type determines the subnodes and properties available for that element.

Node types are summarized in the table below.

Table 83: Nodes

Node Type	Description
Activity	Information about an activity (task), including its priority, status, and scheduled and actual start date, completion date, and duration
Service Form Data	Data in the fields in dictionaries used in a service
OrganizationalUnit	Information about an organizational unit
Person	Information about a person or queue, including name and, for person, company affiliation and contact information
Process	Information about a process (task), including its scheduled and actual dates and duration, and its current status and escalation level

Node Type	Description
Requisition	Information about a service request (shopping cart), including the start date and expected duration
Service	Information about the definition of the service requested, including form data
Message	Information about a failed Service Link task

The OrganizationalUnit (OU) and Person node types have multiple instances within the Namespace. For example, the Person node appears as the requisition's Customer and Initiator, as well as in many other contexts. In this case the “PersonType” is used to reference data for the appropriate person.

The Service node, and its children, relates to a particular service request. Elements in these nodes are available only in contexts when there is a current service request. In particular, emails and design configuration details pertaining to site- and organization-level authorizations, which apply to a complete shopping cart, rather than to an individual request, should not contain Service nodes. Similarly, the Order Confirmation email Template, which pertains to the requisition rather than to individual service requests, should not contain Service nodes.

Details on PersonTypes, as well as the hierarchical structure of the namespace and the properties and subnodes available for each node are summarized in the Person-Based Namespaces, on page 343.

All dates are maintained in Greenwich Mean Time (GMT). Date data includes both the date and the time. All dates are presented in the format:

YYYY-MM-DD mm:HH:ss SSS

That is, a 4-digit year, 2-digit month, 2-digit (zero-filled) day, followed by minutes, hours, seconds, and fractions of seconds using a 24-hour clock. For example, “2007-06-27 23:19:39.197” corresponds to June 27, 2007 at 11:19 PM.

Expressions

Authorizations, reviews, and delivery tasks may be assigned to a person or queue or by using an expression. The expression provides a means to identify the person or queue, stored within Organization Designer, to whom a task should be dynamically assigned.

Configuring an Expression

The person/queue may be identified by using one of four assignment types:

Table 84: Assignment Types

Assignment Type	Meaning
CN	“Common name” – The full name of a user, which is expressed as Firstname Lastname
ID	User ID; the unique numeric identifier of a user

Assignment Type	Meaning
LOGINNAME	The login name of a user
QUEUE	The name of a queue defined in Organization Designer

Expression-based task assignments use the format:

```
AssignmentType=#Namespace_Expression#
```

The assignment statement should follow the rules below:

- The assignment operator (=) must immediately follow the AssignmentType and immediately precede the Namespace_Expression; no spaces are allowed.
- Namespaces used in the Namespace_Expression must be enclosed in hash marks (#).
- Expressions can consist of a combination of namespaces and constant data to create a person's full name or the name of a queue. Separate expression elements by a single space.
- Do not include spaces in variable and functional position names, so that the expressions can evaluate properly.
- All lookups into the database using the result of an expression are case insensitive. For example, LOGINNAME assignment types in which the expression evaluated to "Ann" and "ann" would yield identical results; both would find a person whose login name was "ann"—or "Ann" or "ANN".

CN (Common Name) Assignments

The CN assignment type attempts to find a person by matching the specified full name against the data stored in Organization Designer.

EXAMPLES:

```
CN=#Service.Data.Customer_Information.First_Name#
#Service.Data.Customer_Information.Last_Name#
CN=#Customer.FirstName# #Customer.LastName#
```

The values in the Namespace Expression must exactly match the name of the desired user as stored in Organization Designer. For example, a CN expression whose value is "Carroll Hastings" will not match a user whose name in Organization Designer is "Carol Hastings". For this reason, you should not use dictionary fields into which users have been allowed to type data as free-format text. Instead, use dictionary fields that have been populated by using a Person data type, as this allows the user to choose people from a dialog box, eliminating typing errors. Alternatively, other Namespaces, such as the Customer, Initiator, or Performer, may be used if appropriate.

Use caution when using a CN assignment. It should not be used if two people could have identical first and last names. The assignment will always pick the first person (the one with the lowest PersonID) with the specified name.

When a field in a dictionary is defined as a Person data type, as is the "Select_Person" field in a person-based dictionary, its value is populated by choosing a person from the Select Person search box. The value is set to the unique identifier of the person in Organization Designer. This "ID" value may be referenced in a CN assignment.

EXAMPLE:

```
CN=#Service.Data.Approver.Select_Person#
```

ID (Identifier) Assignments

The PersonID for any of the Person nodes available via Namespace may be referenced in an ID assignment. Alternatively, a text field to which a PersonID has been copied or assigned may be used.

EXAMPLE:

```
ID=#Service.Data.TT_Contact.SupervisorID#
```

LOGINNAME Assignments

Login names uniquely identify a person in the application and can safely be used for assignments.

EXAMPLES:

```
LOGINNAME=#Service.Data.Customer_Information.Login_ID#
LOGINNAME=#Customer.HomeOU.Manager.LoginName#
```

QUEUE Assignments

A queue must sometimes be assigned based not only on a service team, but also on the location of the customer. Such location-based queues can be accommodated by using a QUEUE assignment.

EXAMPLE:

```
QUEUE=#Service.Data.RC_Queue.Location# Desktop Services
```

This assignment would direct the task or approval to a queue named "*Location* Desktop Services," where *Location* is the current value of the Location field in the RC_Queue dictionary on the service form. The resultant value for the QUEUE must *exactly* match the name assigned to a queue defined in Organization Designer. The match is case sensitive. If a match is not found, the task is directed to an Unassigned Work Queue.

Namespace Usage in an Email Template

Namespaces can be used in the following sections of an email template:

- Subject – the subject of the email
- To(s) – the addressees/recipients of the email
- Body – the text of the email

The Namespace data available depends on the context in which the email is sent. For example, task-level data (such as Process or Activity details) will not be available for use in an email that is triggered by a Requisition-level event, such as a Financial Authorization. See the Namespace Reference, on page 338 for details as to which Namespaces are allowable in which emails.

Defining an Email Template

The email Template definition page is accessed from the Administration menu:

Choose **Administration** > **Notifications** > **Email Templates**.

To preview email templates:

Step 1 Start the Service Designer module.

Step 2 From the Services component, choose a service from the left side of the screen.

Step 3 Click the **Plan** tab.

Step 4 Click the **Email** subtab.

Step 5 Assign the email template you wish to preview to one of the moments listed in the drop-down lists in the email subtab area.

Step 6 Click the corresponding **Preview** button to the right of the email template you just assigned.

Whitelist Images in Email Template

You can add external images to the email template for the notification emails generated from Service Catalog. If you find any missing image in the generated email template, then you must whitelist the resources so that authentication is not required to display an image. The ImageServlet generates a url with the documentId for every image that is uploaded in the request center. You can whitelist these images based on their documentId.

To display the images in email template:

Step 1 Open the file '**signon-resources.xml**' from the path ../Cisco Service Portal/deployments/RequestCenter.war/WEB-INF/classes/xml/.

Step 2 Search for 'ExcludedResources'.

Step 3 Add the following ResourceDefinitions to the xml file under ExcludedResources.

Example:

```
 <ResourceDefinition> /* For imageservlet */
      <Name>ImageServlet</Name>
<UrlPattern>/RequestCenter/imageServlet.img?tenantId=1&amp;type=def&amp;documentId=1</UrlPattern>
    </ResourceDefinition>
<ResourceDefinition> /* For other images */
      <Name>Common PNG</Name>
      <UrlPattern>/RequestCenter/images/<file name>.png</UrlPattern>
    </ResourceDefinition>
```

Step 4 Ensure to delete the CnfFile table entries from the database table. Run the query : Delete from CnfFile where LogicName = 'signon-resources.config'.

Step 5 Save and close the xml file.

Step 6 Restart the server for the changes to take effect.

Note Add only individual image files to exclude from authentication and not the entire image folder from ImageServlet, as this will result in a security compromise. To find the documentId of a particular image, click **Source** in Email Template editor, after the image has been uploaded.

Recipients

Namespaces can be used to send emails to people with an interest in the requisition.

Table 85: Recipients

Namespace Example	Resolves To
#Requisition.Customer.Email#	The customer's email, maintained on the To(s): field on the Notifications tab of the Administration module.
#Requisition.Customer.Supervisor.Email#	The email of the customer's supervisor.
#Alternate.Email#	The approver's authorization delegate. The delegate can be assigned via the user's Profile.
#Performer.Email#	The task performer's email, where the performer could be either a person or queue. The person's and queue's email can be found on the To(s): field on the Notifications tab of the Administration module.
#Position.EscalationManager.Email#	The escalation manager's email, where Escalation Manager is a user-defined functional position that has been associated, for example, with a Service Group, and to which a person on the Service Team has been assigned.
#Service.Data.**DictionaryName.FieldName#**	The current value of the specified field in the specified dictionary in the service form. The field should be validated to hold an email address.

Subject

In addition to hard coding data into the subject line of the email template, namespaces can be used to make the template more specific to the requisition (while still maintaining a consistent look throughout services).

Table 86: Subject Namespace

Namespace Example	Resolves To
#Requisition.RequisitionID#	The requisition number

Namespace Example	Resolves To
#Requisition.Customer.FirstName#	The customer's first name
#Service.ServiceDefinition.Name#	The service name

Body

As with the subject field, the body of an email can also be configured to include requisition data, task data, service data, or service form data.

Table 87: Body Namespace

Namespace Example	Resolves To
#Requisition.RequisitionID#	The requisition number.
#Service.Data.**DictionaryName.FieldName#**	The current value of the specified field in the specified dictionary on the current service form.
#Site.URL#	The URL to access Service Catalog; this will resolve to the user's default view, which is set under **Profiles > Preferences**.
#Site.URL#myservices/myservice.do?	A link to the My Services module.
#Site.URL#servicemanager/homepage.do	A link to the Service Manager module.
#URL#	A link to the Task Details (service form) page of the task in Service Manager.

Site URL Namespaces

The namespace #Site.URL# designates the URL through which Service Catalog is accessed. The URL path may also include a specific module to run as well as parameters (following the question mark) to pass to that module. The examples above generate a hyperlink in the email body that will direct the user to the My Services module, the Service Manager module or to capabilities available within those modules.

It is possible to create a namespace reference that will link to just about any page in My Services or Service Manager. The key to determining the appropriate URL is to have a site administrator temporarily turn on site debugging (but please read the caveats in the Cisco Prime Service Catalog Administration and Operations Guide carefully) and note the URL that appears when you navigate to the desired page. When embedding that URL in an email, you must use an encoded representation of characters such as ampersand (&).

Additional examples of such namespace references are given below.

Namespace Example	Resolves To
#Site.URL#myservices/navigate.do?query=requisitionentry status&reqid=#Service.Requisition.RequisitionID#&reqentryid= #Service.RequisitionEntryID#&formAction= displayEntryStatus&serviceid=#Service.ServiceID#&requisitionId= #Service.Requisition.RequisitionID#&waiting=0&**authTaskType=4**&activityId= #ActivityID#&buttonsPresent=true&	A link to the service authorization page within My Services with the current approval/review task selected. The Authorization Task Type of 4 refers to a Service Group authorization.
#Site.URL#myservices/navigate.do?query=authorizationtask&taskId=#ActivityID #&return_to_url=authorizationslist&	A link to the Service Group Authorization identified by ActivityID, which will take the user directly to the requisition containing the service requiring approval.
#Site.URL#myservices/navigate.do?query=authorizationlist&return_to_url=portal	A link to the authorizations list page within My Services where the user will see all the authorizations awaiting approval.
#Site.URL#myservices/navigate.do?query=requisition&requisitionId= #Requisition.RequisitionID#	A link the requisition confirmation page in My Services.

Site URL Configuration

As part of the installation process, administrators specify the Service Catalog URL. This URL is stored in the configuration file newscale.properties. A sample entry is shown below.

```
!-------------------------------------------------------------------
!Define the URL for the application
!This is used in emails for constructing the various urls
```

```
!--------------------------------------------------------------------
ObjectCache.Application.URL=http://appsrv01.celosis.com/RequestCenter/
```

The entry for ObjectCache.Application.URL is the value of the namespace #Site.URL# and #URL#. The URL typically ends with "/RequestCenter/"; however, it is possible to configure the application server to automatically reroute requests to /RequestCenter/, so this portion of the path may be absent from the properties file.

In this case, the word **/RequestCenter/** must be added after #Site.URL#, such as #Site.URL#/RequestCenter/myservices/myservice.do? , and to #URL#, such as #URL#/RequestCenter/myservices/myservice.do? to ensure that the namespace link resolves to the correct URL.

In order to determine if the /RequestCenter reference must be inserted, you may consult the newscale.properties file (or have the application server administrator report on its contents). An alternate method is:

- Enter #Site.URL# or #URL# by itself in an email and send the email when, for example, a task starts.
- When receiving the email, note if the URL resolves to just http://www.mycompany.com or http://www.mycompany.com/RequestCenter/.
- If the resolved URL does not include the wording **/RequestCenter/**, then add **/RequestCenter/** after *#Site.URL#* or *#URL#* respectively.

Service Link Message Namespaces

The namespace node #Message# is available only within the body of email messages generated as a result of a Service Link failure to deliver an outbound message. The template to be used is specified as the "Failed email" in the Service Link agent definition. Elements in this node may be useful in helping a administrator diagnose the cause of the failure.

Service Manager Task Details URL

The #URL# namespace takes the user directly to the form data of a task in Service Manager. This is very nice for Ad-hoc tasks, and for teams that rarely perform tasks in Service Catalog. It can also be used for authorization tasks, but after the approver approves/denies the service, they find themselves in Service Manager, which can be confusing.

Some customers prefer this to the multistep process required to view the form data when approving via the #Site.URL#myservices/navigate.do?query=authorizationtask&taskId=#ActivityID# namespace. In fact, here, an approver can approve without seeing the form data which could be a slight audit issue depending if the data is important to the approval—they have to know to click on the name of the service to see the form data.

Any change to the ObjectCache.Application.URL in the newscale.properties file will also affect the #URL# namespace in the same way as the #Site.URL#.

Namespace Processing and Alternate Values

When a Namespace element is referenced, Service Catalog retrieves its value from the current context and replaces the reference to the Namespace in the email with the resolved value. Some Namespace elements are always guaranteed to exist and have a valid value, such as those relating to the customer or initiator of a request. Other Namespace elements, however, may be undefined at certain times. For example, a functional position may be vacant; no authorization delegate has been designated; or an employee is temporarily without a supervisor.

Any reference to a Namespace that is undefined is blank. Except for the email namespace, it is possible to provide an alternate value, by using a slash (/) immediately following the Namespace element name, and then assigning the alternate value.

EXAMPLE:

```
TO: #Alternate.email# #Performer.email#
```

- Design emails to be sent to both the designated reviewer and a possible alternate (authorization delegate). If no delegate is currently designated, the #Alternate.email# is blank

```
#Supervisor.LastName/Your current supervisor#
```

- If the person referenced currently has a supervisor, the supervisor's last name will appear in the email; otherwise, the string “Your current supervisor” will appear.

Namespace Usage for Policy Configuration

Namespaces are used when you configure an action for a service item policy. The actions that use namespaces are policy alert, send email, and order service. These namespaces are resolved during execution.

For more information about policies, see Understanding Service Items Policies, on page 27.

Table 88: Namespace Example

Namespace Example
#SI.Subscription.ID#
#SI.Subscription.OrganizationalUnit.ID#
#SI.Subscription.OrganizationalUnit.Name#
#SI.Subscription.Customer.Name#
#SI.Subscription.Customer.LoginName#
#SI.Subscription.Customer.Email#
#SI.Subscription.AssignedDate#
#SI.Subscription.SubmittedDate#
#SI.Subscription.RequisitionID#
#SI.Subscription.RequisitionEntryID#
#SI.Subscription.Operation#
Account Namespaces
#SI.Subscription.Account.ID#

Namespace Example
#SI.Subscription.Account.Name#
#SI.Subscription.Account.FunctionalPosition.<FPName>#.ID
#SI.Subscription.Account.FunctionalPosition.<FPName>#.Name
#SI.Subscription.Account.FunctionalPosition.<FPName>#.LoginName
#SI.Subscription.Account.FunctionalPosition.<FPName>#.Email
Agreement Namespaces
#SI.Subscription.Agreement.ID#
#SI.Subscription.Agreement.Name#
Service Item Attributes Namespaces
#SI.Type.ID#
#SI.Type.Name#
#SI.Type.DisplayName#
#SI.Type.Classification.ID#
#SI.Type.Classification.Name#
#SI.Attribute.<AttributeName>#
Policy Attributes
#Policy.ID#
#Policy.Name#
#Policy.Type#
#Policy.Parameter.Attribute#
#Policy.Parameter.Threshold#
#Policy.Parameter.Operator#
#Policy.Parameter.Hours#
#Policy.Parameter.Value#

Namespace Example
Account Namespaces
#Policy.Account.ID#
#Policy.Account.Name#
#Policy.Account.FunctionalPosition<FPName>.ID#
#Policy.Account.FunctionalPosition<FPName>.Name#
#Policy.Account.FunctionalPosition<FPName>.LoginName#
#Policy.Account.FunctionalPosition<FPName>.Email#

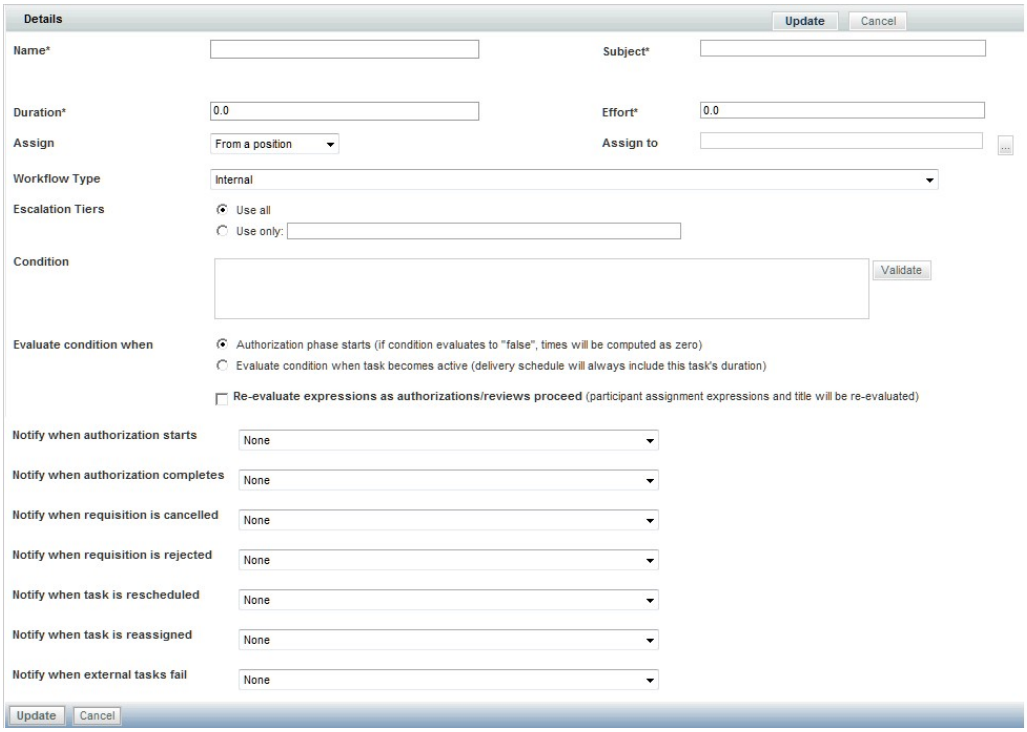
Namespace Usage for Authorizations and Reviews

Namespaces can be used in the following components of an authorization/review definition:

- Subject: Namespaces can be used to include form data in the task name
- Assign to: The authorization/review may be assigned from an expression.
- Condition: The authorization/review task may be conditionally performed

The subtab for specifying the details of an authorization or review looks like the figure below:

Figure 25: Service Definition Subtab for an Authorization Details



Subject

The subject of an authorization or review task can be configured to reflect requisition data through the use of namespaces, thus making the subject specific to the customer's order. Simply include the namespace in the Subject field.

The most frequently used namespace is the name of the service to which the current review/authorization applies. This is typically concatenated with other descriptive text; for example:

```
Group Review for #Name#
```

The namespace #Name# is available as a shortened form for the Service Name.

Only namespaces referring to a specific attribute of the service and **NOT** the content of a service form can be used for Organizational Unit Reviews or Authorizations or Financial Authorizations. For instance, when using a namespace to refer to the customer's first name:

```
#Requisition.Customer.FirstName# -- CORRECT
#Service.Data.Customer_Information.First_Name# -- INCORRECT
```

Service Group Reviews and Authorizations may use #Service.Data# namespaces. Such reviews/authorizations apply to an individual service, so the service (form) data is available. Financial Authorizations and Organizational Unit Reviews/Authorizations apply to an entire requisition, which may include multiple services (requisition entries); therefore, the service data for an individual service is not available.

Assigning from an Expression

Authorization and review tasks can be assigned to:

- the person currently filling a *functional position* defined in Organization Designer.
- a static person or queue.
- the person or queue identified by the value of an *expression* .

In many cases, assigning tasks to a predefined position, individual, or queue is not a viable option. For example, the same service may be handled by different queues, depending on the location of the requestor. In cases like these, you may need to route the tasks based on requisition data entered or otherwise supplied in the ordering moment.

Step 1 Choose **From an expression** in the Assign drop-down list.

Step 2 Enter the desired expression in the “Assign to” field.

Example:

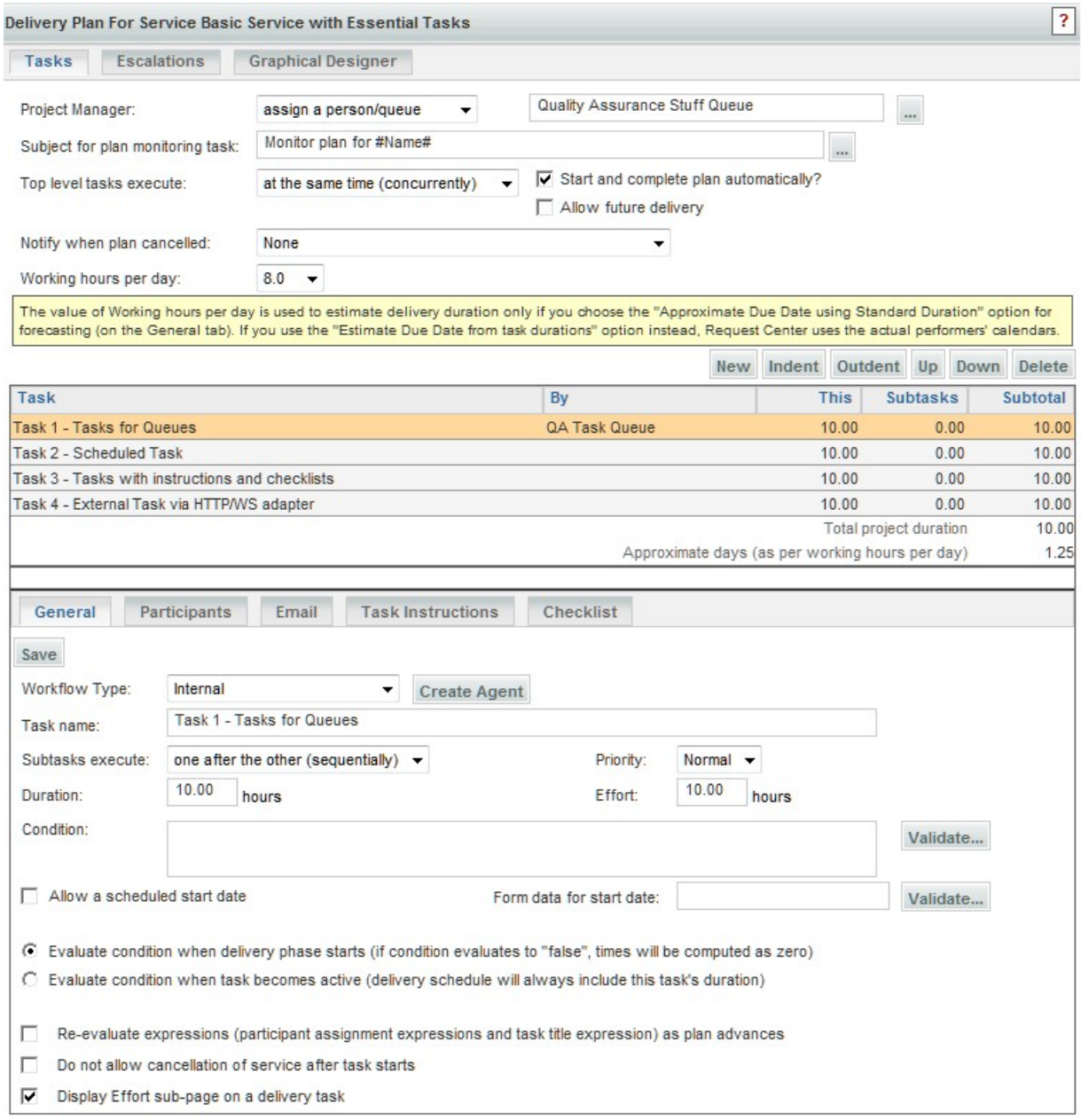
```
ID=#Service.Data.Customer_Information.SupervisorID#
```

Step 3 Click **Update** to save.

Delivery Plans and Tasks

The delivery plan is configured via the Plan tab of **Service Designer > Services**.

Figure 26: Monitor Task Definition



Each task is configured by choosing the task from the list of tasks (or clicking New to create a new task) and filling out the task-related subtabs.

For more information about delivery plan, see Configuring AMQP Tasks for Publishing Service Request to an External System, on page 199.

Namespace Usage for Delivery Tasks

Configuring a service's delivery plan and component tasks provides multiple opportunities to use expressions and namespaces.

Namespaces can be used in the following components of an authorization/review definition:

- Project Manager: May be assigned from a namespace expression.
- Subject for Monitor Plan: May include namespaces as well as literal text.
- Task name: May include namespaces as well as literal text.
- Participants: Performer and supervisor may be assigned from a namespace expression.
- Condition: The task may be performed only if the specified condition, which may use namespaces, is true.

Project Manager for the Delivery Plan

The Project Manager may be assigned from a person, queue, or expression. The expression may use any of the Assignment Types to identify a person in Organization Designer.

EXAMPLES:

```
LOGINNAME=#Service.Data.Project.ManagerLogin#
ID=#Customer.Supervisor.ID#
```

Monitoring Task Subject

The subject for the monitoring task typically includes the #Name# namespace, to refer to the name of the service being monitored.

```
Monitor Plan for #Name#
```

Namespaces referring to performers, either people or queues, cannot be used.

Namespaces referring to the content of a service form can be used, but this may have side effects, as documented for the Delivery Task Name below.

Delivery Task Name

The delivery task name typically includes the service name, through the use of the #Name# Namespace. Service data (#Service.Data....#) can also be used. Since all tasks are created when the form is submitted, the value of the specified field must have been supplied in the ordering moment.

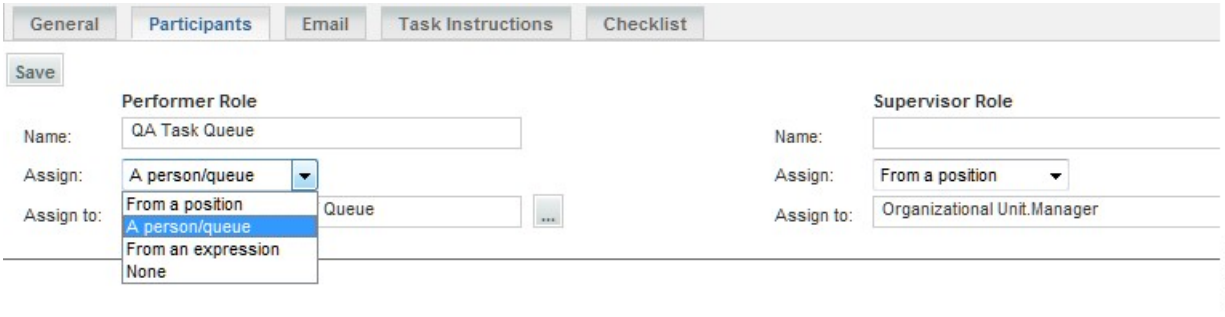
Note Using form data for a task name is not best practice, since the task would no longer be groupable by the task name in a reporting environment. Further, a Service Manager query would need to use the "contains" operator to retrieve the rest of the task name, which may adversely affect performance.

Assigning Roles (Performers and Supervisors) to the Task

As with authorizations, the service team or person assigned to tasks in the delivery plan may depend on data on each requisition, such as a customer's location. Expressions can be used to intelligently route tasks, rather than creating different forms or workflows for each possible scenario.

Plan tasks that can be dynamically routed include the Performer and Supervisor roles found on the Participants tab of each task:

Figure 27: Delivery Plan Task Participants



To assign "From an expression", choose that option from the Assign drop-down list and click on the ellipsis (...) next to the "Assign to" field. The Edit Expression window appears to allow you to enter and validate the expression.

Conditional Statements

A conditional statement allows tasks to be initiated or skipped based on the condition and can be configured to evaluate at various times within the authorization, review, and delivery plan.

Namespaces used in conditional statements are not enclosed in # signs:

```
Service.Data.DictionaryName.FieldName="Yes" -- CORRECT
#Service.Data.DictionaryName.FieldName#="Yes" -- INCORRECT
```

Conditional Authorization and Review Tasks

Authorizations and reviews may be needed only when certain conditions exist. For example, an authorization may be required only when a unit price exceeds a specified threshold. To define a conditional authorization or review task, follow the procedure below.

Step 1 Click on the authorization/review you are configuring. The Review/Authorization Details window appears, as shown in Figure 28: Review Details Window.

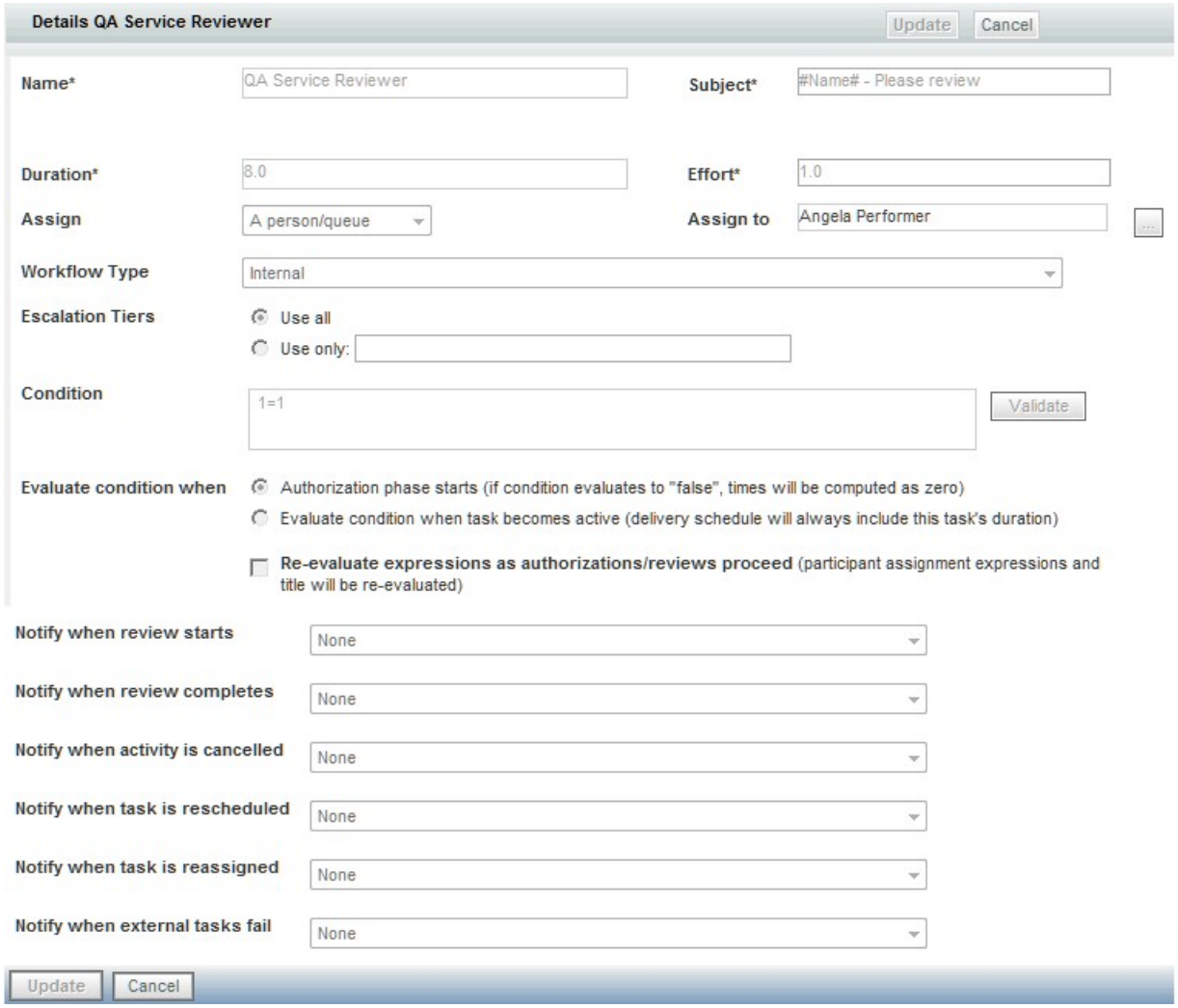
Step 2 Enter the Condition under which the particular authorization or review task should be executed.

Step 3 Validate the condition by clicking **Validate**.
A Validate window appears. The message "unexpected token" indicates that the namespace used is not valid in this context or, perhaps, that you have forgotten to enclose an alphanumeric literal in quotes.

Step 4 Click **OK** to dismiss the Validate window.

Step 5 Click **Update** to save the Review/Authorization Details.

Figure 28: Review Details Window



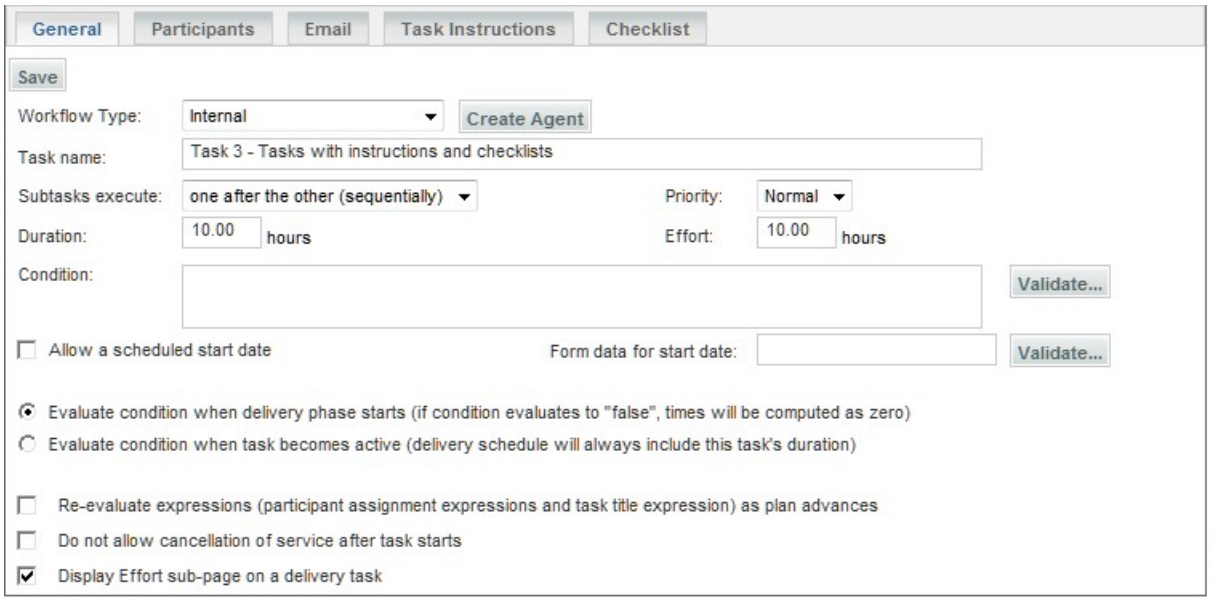
Validation checks that any namespace specified is valid for the current scope (the specific level of review/authorization or task), with the exception of dictionary fields (Data.**DictionaryName.FieldName**). This is perfectly legitimate, since the Review/Authorization may be defined at a site- or organization- level, independent of the service with which it is integrated. However, it may cause a runtime error: if the specified namespace, for example, Data.EUIT_ACCESS.Access_Type, does not exist.

Validation also checks that correct relational and arithmetic operators are used.

Conditional Delivery Plan Tasks

As with authorization and review tasks, a plan task in the service delivery moment may also be conditional.

Figure 29: Task Definition General Tab



Evaluating the Condition

Once the condition has been entered, you must decide when that statement will be evaluated. Each task (approval, review, or delivery) with a condition can be evaluated either

- At the beginning of a phase (Authorization or Delivery moment), or
- When the activity becomes active.

Evaluate condition when authorization/delivery phase starts

The designer can choose to evaluate a conditional statement when a **phase** starts by choosing the "*Evaluate condition when delivery phase starts (if condition evaluates to "false", times will be computed as zero)*" option within a specific task, as shown above for delivery tasks. A " phase" corresponds to any of the system moments defined for processing a requisition. Each authorization or review has its own moment; all delivery tasks are performed within the Service Delivery moment.

Authorization tasks are always serial. You could put one conditional on one task saying if field= "somevalue" and a conditional on another saying field<> "somevalue". That way, you know one authorization task will always be executed, and if you choose "*when authorization phase starts* ", only one authorization task will appear in the process view. If you choose "when task becomes active", both tasks appear, but one would be skipped.

The "*if conditions evaluate to "false", times will be computed as zero*" statement means that Service Catalog will evaluate the conditions at the beginning of the phase. If these conditions are false, then the corresponding tasks will not be executed, and the **Due Date** for the service will be calculated without including the duration of these tasks.

Evaluate condition when activity becomes active

Alternatively, the designer can choose to evaluate a conditional statement when the task starts by choosing the "*Evaluate condition when activity becomes active (times will not be affected, scheduling will be done by using these efforts)* " option.

Service Catalog will evaluate the condition at the beginning of each task. If the condition is false, the corresponding task will not be executed. Durations for any task configured with this option will be used to calculate the due date upon submission.

Re-evaluate Expressions as plan advances

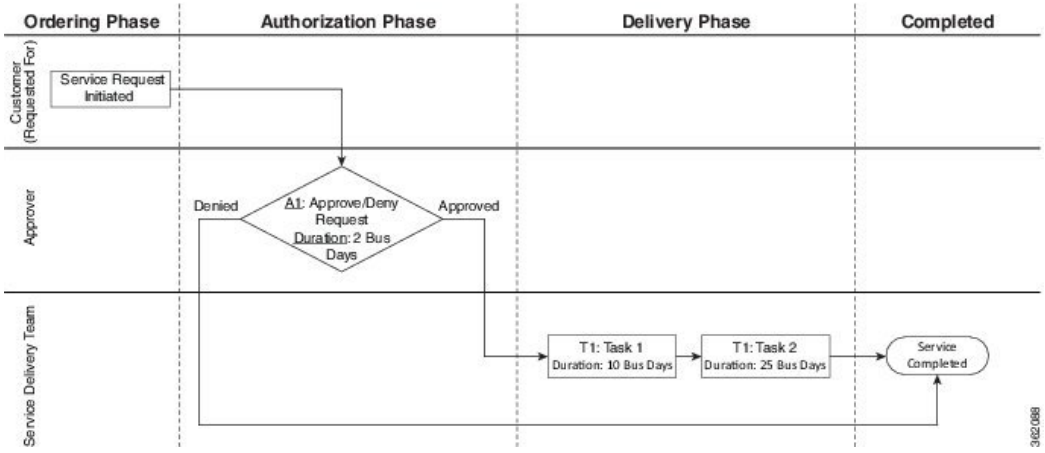
The Re-evaluate Expressions feature is useful for designs in which there are multiple authorizations. It enables the person performing an authorization task to enter information in the service form which is then used to re-evaluate the expression used to assign the performer for a *subsequent* authorization task. If this option is not checked, all information used in expressions in the authorization task must be present during the Ordering moment.

This feature allows dynamic assignment of an approval or review task to a user (person or queue) and dynamic adjustment of the task title during the authorization or review moment. Once the authorization/review becomes active, the expression will be evaluated and the task will be assigned appropriately. This capability is available only for authorization and review tasks, not for delivery tasks.

Start of a Phase and Start of a Task

The following figure pinpoints and differentiates the start of a phase and the start of a task.

Figure 30: Start of a Phase and Start of a Task



1	Start of Phase	2	Start of Task

Syntax for Conditional Statements

Conditional statements may include arithmetic operators as well as relational (logic) operators.

The type of logical operators you can use depend on the HTML representation of the field being tested. Most fields have (input elements) that allow only one value to be assigned; these include text, text area, radio button, single-select (drop-down) list, and radio buttons. The following logical operators can be used in conditional statements applied to such fields:

Table 89: Syntax for Conditional Statements

Operator	Usage
<	Less than.
>	Greater than.
<=	Less than or equal to.
>=	Greater than or equal to.
<>	Not equal to.
=	Equal to.
OR	Performs the logical OR of two or more statements. The result is true if all of the statements are true and false otherwise; that is, if any of the statements are false.
AND	Performs the logical AND of two or more statements. The result is true if any of the statements is true and false otherwise.

Operators for conditionals have the standard order of operation:

```
-  (negative)
*  (multiplication), /  (division)
+  (addition),   -  (subtraction)
<, >, >=, <=, <> (not equal), =
NOT
OR
AND
```

Parentheses can be used to change the operator precedence or clarify the condition.

EXAMPLES:

```
ActivityID >= 50
Customer.FirstName = "Ann"
Requisition.ActualCost >= 2000
Service.Quantity * Service.PricePerUnit <= 1000
Data.Approver.Custom_2 = "VP" OR Data.Approver.Custom_2 = “Director”
```

The INCLUDES operator can only be applied to fields that can hold multiple values. These include multi-select (drop-down boxes) and check boxes. The following logical operators can be used in conditional statements applied to such fields:

Operator	Usage
OR	Performs the logical OR of two or more statements. The result is true if all of the statements are true and false otherwise; that is, if any of the statements are false.

Operator	Usage
AND	Performs the logical AND of two or more statements. The result is true if any of the statements are true and false otherwise.
NOT	Negates the value of the condition.
INCLUDES	True if the specified value is a currently chosen option for the Namespace variable; applicable only to fields designated as “Multi-value” in the dictionary, typically fields with HTML representations of multi-select and check box.

EXAMPLE:

```
Data.EUIT_ACCESS.AccessType INCLUDES “DSL” AND NOT
Data.EUIT_ACCESS.AccessType INCLUDES “Dial-Up”
```

Note No “#” signs are used to enclose the Namespace.

- An alphanumeric value included in the condition must be enclosed within double quotation marks (“”).
- Namespace names are not case sensitive. The recommended standard is to use Title case (capitalizing the first letter of all words).
- All alphanumeric comparisons are case sensitive. For example, the condition “Data.MoveIndividual.FirstName=”Matt”” would be true only if the value of the FirstName field in the MoveIndividual dictionary were “Matt”, with an initial capital letter and the rest lower case letters.
- Any Boolean Namespaces (those whose names start with “Is”) have different values, depending on the underlying database. In SQLServer, the value of a Boolean is either true or false. In Oracle the corresponding values are 1 (for true) and 0 (for false).
- Less than and greater than operators for numeric value comparisons are not supported for dictionary fields. They are treated as text during comparisons. For example, the condition “Service.Data.VM.MemoryGB > 4” is evaluated to false if the dictionary field VM.MemoryGB has a value of 16.
- Multibyte characters are not supported for text string comparisons.
- An ampersand (“&”) included in a literal must be encoded as “&”. For example, the value “two & three” would appear in an expression as “two & three”.
- The following operators are not supported for boolean type fields:
  - is greater than
  - is less than
  - is equal to ignore case
  - begins with

- ends with
- contains
- does not contain

Tips and Techniques

Use a condition that always evaluates to false (for example, "1=2") in a conditional statement to specify a task that will automatically be skipped.

The most common use cases for this are:

- When a service needs to "autocomplete" without having any tasks completed. The skipped task will mark the end of the delivery plan, and the requisition is marked as complete.
- When an email needs to be sent without having to complete a task or when multiple emails are needed during a given moment in the following ways:
  - Create a parent task. On the email tab, choose an email to be sent out at completion (you may also configure one to go out when the activity becomes active).
  - Create a child task with the condition 1=2. Set the condition to evaluate when the activity becomes active.

Namespace Reference

This reference section lists all Namespaces and the contexts in which each can be used.

Namespace Objects and their Relationships

The following diagram illustrates the nodes in the Namespace and the relationships between the nodes. The type of a node determines not only its own properties, but also the subcontexts (other nodes) to which it provides access.

The labeled boxes in this figure represent the types of Namespace nodes. Labeled arrows show the Namespace elements that allow the properties of one node to be accessed from another node. This figure can serve as a guide for service designers and administrators who need to navigate the Namespace to get access to Service Catalog data. For example, tracing the arrow labeled "Customer" from the Process node, we can see that the "Customer" element gives the Process node access to the properties of the Person node. So, for example, to access the Customer's login name in a conditional statement for a task in a delivery plan, the condition would reference the namespace:

```
Requisition.Customer.LoginName
```

To access the same namespace from an email for a delivery plan, the namespace reference would be:

```
#Service.Requisition.Customer.LoginName#
```

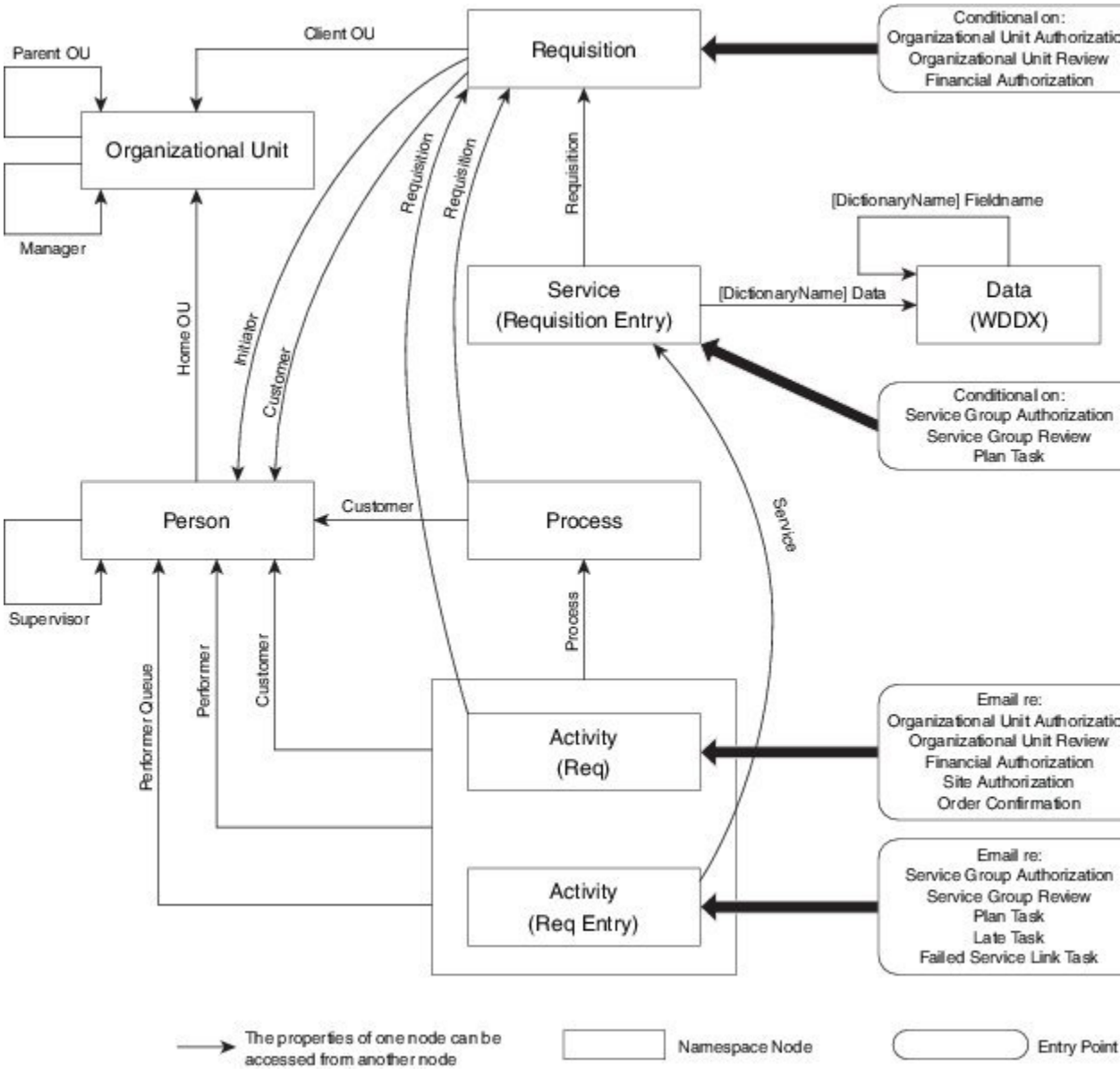
Node relationships are not bidirectional. The fact that the Process node has access to the properties of the Person node does not imply that the properties of the Process node are also available to the Person node.

From the perspective of the service designer or administrator who wishes to use Service Catalog data in formatted emails or conditional statements, there are multiple "entry points" to the paths between nodes:

- Activities associated with Requisition Entries (services), such as delivery plan tasks, Service Group Authorizations, Service group reviews, and emails generated by any of these activities.
- Activities associated with Requisitions, such as Departmental Authorizations, Departmental Reviews, and Financial Authorizations.

These entry points determine the Namespace context that applies when a service designer or administrator is configuring email to be sent from an activity, or defining a conditional execution statement. Think of them as the starting point for navigating the paths between nodes. Where you enter the Namespace determines what nodes, and ultimately what properties and data values, will be available to you from that particular activity.

Figure 31: Namespace Objects and their Relationships



Email Namespace Elements

The # character must be used to delimit Namespace elements in email notifications.

OU-based Authorizations and Reviews

The authorizations and reviews at an organizational level may trigger an email:

- Organizational Unit Authorization
- Organizational Unit Review
- Financial Authorization

These email entry points support the Namespace elements listed below:

```
#ActivityID#
#Priority# - priority assigned to the task: 1=high, 2=normal, 3=low
#DueOn#
#Subject# - the name of the task
#Waiting#
#ScheduledStart#
#StartedOn#
#CompletedOn#
#ExpectedDuration# - typical task duration, in hours
#ExpectedDurationUnits# - units in which task duration is displayed
#ActualDuration# - actual task duration, in hours
#CurrentDate#
#StateName# - the status of the task
#URL#
#Customer.*# - where * denotes any Customer element
#Performer.*# - where * denotes any Performer element
#PerformerQueue.*# where * denotes any PerformerQueue element
#Process.*# -- where * denotes any Process (task) element
#Requisition.*# - where * denotes any Requisition element
```

Tasks and Service Group Authorizations/Reviews

Tasks which are part of the delivery plan, as well as service group authorizations and reviews may trigger an email:

- Service Group Authorization
- Service Group Review
- Plan Task
- Late Task
- Ad-hoc Task
- Failed Service Link Task

These email entry points support the Namespace elements listed below:

```
#ActivityID#
#Priority#
#DueOn#
#Subject#
#Waiting#
#ScheduledStart#
```

```
#StartedOn#
#CompletedOn#
#ExpectedDuration#
#ActualDuration#
#Instructions# -- for Adhoc tasks only
#CurrentDate# -- the current date and time in GMT
#StateName# -- the current status (state) of the activity
#URL# -- the URL for directly accessing Task Details for this activity
#Customer.*# -- all Customer elements; for an adhoc task only, this   refers to the service
 performer who created the task
#Performer.*# -- all Performer elements
#PerformerQueue.*# -- all Queue elements
#Process.*# - all Process elements
#Service.ServiceID#
#Service.ProcessTrackingID#
#Service.Quantity#
#Service.PricePerUnit#
#Service.HasPrice# -- false (0) if the service has no price, 1 (true)otherwise
#Service.Bundled# -- true (1)if the service is a child service in a   bundle, false (0)
otherwise
#Service.isBundle# -- true (1) if the service is itself a bundle, false  (0) otherwise
#Service.BundledServices# -- the number of child services bundled with  the current service
#Service.ServiceDefinition.Name#
#Service.ServiceDefinition.IsEntitlement#
#Service.ServiceDefinition.DescriptionURL#
#Service.ServiceDefinition.ExpectedDuration#
#Service.ServiceDefinition.FunctionalPosition.PersonTypeElement#
#Service.ServiceDefinition.ServiceGroup.Name#
#Service.ServiceDefinition.ServiceGroup.FunctionalPosition
.PersonElement#
#Service.RequisitionEntryID#
#Service.Requisition.URL#
#Service.Requisition.ProcessTrackingID#
#Service.Requisition.ExpectedDuration#
#Service.Requisition.StartedDate#
#Service.Requisition.ActualCost#
#Service.Requisition.ExpectedCost#
#Service.Requisition.RequisitionID#
#Service.Requisition.Name#
#Service.Requisition.Customer.*# --all Customer elements
#Service.Requisition.Initiator.*# -- all Initiator element
#Service.Requisition.ClientOU.*# -- all ClientOU elements
#Service.Data.DictionaryName.FieldName#
```

The service data namespaces generally have the format

```
#Service.Data.DictionaryName.FieldName#
```

Conditional Namespace Elements

The # character is NOT used to access variables in conditionals. Some characters such as #, ", &, +, and – on the left side of the condition will cause database problems.

OU-based Authorizations and Reviews

The authorizations and reviews at an organizational level may be conditionally executed.

- Organizational Unit Authorization
- Organizational Unit Review
- Financial Authorization

These conditional entry points support the following Namespaces:

```
Site.URL
```

```
Customer.*
Initiator.*
ClientOU.*
```

Tasks and Service Group Authorizations/Reviews

Service group authorizations and reviews, as well as tasks which are part of the delivery plan allow conditions to determine whether the task/review is executed:

- Service Group Authorization
- Service Group Review
- Plan Task
- Late Task

These conditional entry points support the following Namespaces:

```
Requisition.*
Data.DictionaryName.FieldName
```

Organizational Unit-Based Namespaces

Information is available via Namespaces about organizational units. Organizational units may occur in many contexts, denoted by these namespace entity types:

Table 90: Organizational Unit-Based Namespaces

Entity Type	Description	Example
ClientOU	Organizational unit of the client for the requisition	#Requisition.HomeOU.**OrgElementType#**
HomeOU	Home organizational unit of the performer, performer queue, or customer or of the manager of any of these people	#Performer.HomeOU.**OrgElementType#**
ParentOU	The parent organizational unit of the current OU	#Customer.HomeOU.ParentOU.**OrgElementType#**

Organizational Unit Element Types (OrgElementType) are listed below, in the context of the ClientOU.

```
#ClientOU.Name#
#ClientOU.Description#
#ClientOU.OrganizationalUnitID#
#ClientOU.OrganizationalUnitTypeID#
#ClientOU.CostCenterCode#
#ClientOU.FunctionalPosition.PersonElementType#
#ClientOU.Manager.PersonTypeElement#
#ClientOU.ParentOU.Name#
#ClientOU.ParentOU.Description#
#ClientOU.ParentOU.OrganizationalUnitID#
#ClientOU.ParentOU.OrganizationalUnitTypeID#
```

```
#ClientOU.ParentOU.CostCenterCode#
#ClientOU.ParentOU.FunctionalPosition.PersonElementType#
```

Person-Based Namespaces

The Customer, Initiator, Alte rnate, and Performer Namespace elements expose information about a person stored in Organization Designer. Therefore, all entity types support the same variables; only the element denoting the entity type (“Customer”, “Initiator”, “Alternate”, or “Person”) will vary.

A list of person-based Namespace nodes is given below. Namespaces are available for all basic and extended person attributes. To form a valid Namespace variable, use these as the last part of the Namespace name, where the first part is the entity type. As always, the Namespace must be enclosed in hash marks (#) when used in an email.

PersonType is one of:	
Alternate	The designated delegate for the current authorization task performer. The following are the only namespaces supported for Alternate namespaces: #Alternate.FirstName# #Alternate.LastName# #Alternate.Title# #Alternate.TimeZoneID# #Alternate.Email#
Customer	Generally, the person for whom the current requisition was ordered; for task-based emails and escalations, the Supervisor of the task performer; for ad-hoc tasks, the task performer who created the ad-hoc task.
Customer.Supervisor	The supervisor of the customer for the current order.
Customer.HomeOU.Manager	The manager of the home organizational unit of the customer for the current order.
Initiator	The person who ordered the current requisition.
Initiator.Supervisor	The supervisor of the person who ordered the current requisition.
Initiator.HomeOU.Manager	The manager of the home organizational unit of the person who ordered the current requisition.
Performer	The person responsible for performing the current task.
Performer.Supervisor	The supervisor of the task performer.
Performer.HomeOU.Manager	The manager of the home organizational unit of the task performer.

PersonType is one of:	
ClientOU.Manager	The manager of the organizational unit for which an organizational unit review or authorization is being performed.
FunctionalPosition	A Service Catalog-defined functional position within an organization, service, or service group.
Position.**FunctionalPosition**	A user-defined functional position within an organization, service, or service group.

Elements of the Person-type namespace node are listed below. If no element is specified, the expression returns the person's unique identifier in the database.

```
#PersonType
.FirstName#
#PersonType
.LastName#
#PersonType
.Title#
#PersonType
.Birthdate#
#PersonType
.Hiredate#
#PersonType
.SSN#
#PersonType
.EmployeeCode#
#PersonType
.IsOffice#
#PersonType
.TimeZoneID#
#PersonType
.Email#
#PersonType
.CompanyAddress# -- a concatenation of all lines of the company address, including formatting
 into multiple lines
#PersonType
.PersonalAddress# -- a concatenation of all lines of the personal address, including
formatting into multiple lines
#PersonType
.SimpleCompanyAddress# -- a concatenation of all lines of the company (business) address
#PersonType
.SimplePersonalAddress# -- a concatenation of all lines of the personal address
#PersonType
.DetailedCompanyAddress.Street1#
#PersonType
.DetailedCompanyAddress.Street2#
#PersonType
.DetailedCompanyAddress.City#
#PersonType
.DetailedCompanyAddress.StateProvince#
#PersonType
.DetailedCompanyAddress.Zip#
#PersonType
.DetailedCompanyAddress.Country#
#PersonType
.DetailedPersonalAddress.Street1#
#PersonType
.DetailedPersonalAddress.Street2#
#PersonType
.DetailedPersonalAddress.City#
```

```
#PersonType
.DetailedPersonalAddress.StateProvince#
#PersonType
.DetailedPersonalAddress.Zip#
#PersonType
.DetailedPersonalAddress.Country#
#PersonType
.Location# - a concatenation of all aspects of the location, with formatting to place
elements on separate lines
#PersonType
.DetailedLocation.Building#
#PersonType
.DetailedLocation.BuildingLevel#
#PersonType
.DetailedLocation.Office#
#PersonType
.DetailedLocation.Cubicle#
#PersonType
.SimpleLocation# -- a concatenation of all aspects of the location, with spaces separating
 the elements
#PersonType
.WorkPhone#
#PersonType
.HomePhone#
#PersonType
.Fax#
#PersonType
.Mobile#
#PersonType
.Pager#
#PersonType
.LoginName#
#PersonType
.TimeZone#
#PersonType
.ExtManager#
#PersonType
.CompanyCode#
#PersonType
.Division#
#PersonType
.BusinessUnit#
#PersonType
.DepartmentNumber#
#PersonType
.CostCenter#
#PersonType
.ManagementLevel#
#PersonType
.Region#
#PersonType
.EmployeeType#
#PersonType
.LocationCode#
#PersonType
.Custom1#
#PersonType
.Custom2#
#PersonType
.Custom3#
#PersonType
.Custom4#
#PersonType
.Custom5#
#PersonType
.Custom6#
#PersonType
.Custom7#
#PersonType
.Custom8#
#PersonType
```

```
.Custom9#
#PersonType
.Custom10#
```

EXAMPLES:

```
#Person.FirstName# #Person.LastName#
#Customer.Supervisor.Fax#
```

Customer Namespaces

In delivery tasks, including escalations, the customer is the Task Supervisor. For ad-hoc tasks, the customer is the task performer who initiated the ad-hoc task. For all other tasks, and for a requisition, the customer is the person for whom the service has been requested. Customer namespaces allow access to customer information, including:

- Customer (person) information as defined in the person's profile and accessible via **Organization Designer > People**
- All person/profile information defined for the customer's supervisor
- Information about the home Organizational Unit of the customer
- All person/profile information defined for the manager of the customer's home OU

The namespaces below must be enclosed in hash marks (#) when used in emails, but entered without the hash marks when used in expressions. A complete list of person-type namespaces is listed in the Person-Based Namespaces, on page 343.

Customer Namespaces.

```
#Customer.PersonTypeElement#
#Customer.Supervisor.PersonTypeElement#
#Customer.HomeOU.Name#
#Customer.HomeOU.OrganizationalUnitID#
#Customer.HomeOU.OrganizationalUnitTypeID# -- the type of unit, where 1=service team, and
2=business unit
#Customer.HomeOU.CostCenterCode#
#Customer.HomeOU.Manager.PersonTypeElement#
#Customer.HomeOU.ParentOU.Name#
#Customer.HomeOU.ParentOU.OrganizationalUnitID#
#Customer.HomeOU.ParentOU.OrganizationalUnitTypeID# -- the type of unit, where 1=service
team, and 2=business unit
#Customer.HomeOU.ParentOU.CostCenterCode#
```

Performer Namespaces

The performer is the person responsible for performing a task in the service's delivery plan. Performer namespaces are not available in context in which no task is current—these include the email for Organization Unit reviews and authorizations; and financial authorizations.

Performer namespaces allow access to performer information, including:

- Performer (person) information as defined in the person's profile and accessible via **Organization Designer > People**
- All person/profile information defined for the performer's supervisor
- Information about the home Organizational Unit of the performer
- All person/profile information defined for the manager of the performer's home OU

The namespaces below must be enclosed in hash marks (#) when used in emails, but entered without the hash marks when used in expressions. A complete list of person-type namespaces is listed in the Person-Based Namespaces, on page 343.

```
#Performer.PersonTypeElement#
#Performer.Supervisor.PersonTypeElement#
#Performer.HomeOU.Name#
#Performer.HomeOU.OrganizationalUnitID#
#Performer.HomeOU.OrganizationalUnitTypeID#
#Performer.HomeOU.CostCenterCode#
#Performer.HomeOU.Manager.PersonTypeElement#
```

PeformerQueue Namespaces

Performer Queue namespaces provide information about the queue to which a review, authorization, or task was assigned. A subset of the Person-based namespace elements and properties are meaningful—those which are exposed in the user interface for maintaining queues in Organization Designer.

```
#PerformerQueue.FirstName#
#PerformerQueue.LastName#
#PerformerQueue.TimeZoneID#
#PerformerQueue.Email#
#PerformerQueue.WorkPhone#
#PerformerQueue.HomePhone#
#PerformerQueue.Fax#
#PerformerQueue.Mobile#
#PerformerQueue.Pager#
#PerformerQueue.TimeZone#
#PerformerQueue.HomeOU.Name#
#PerformerQueue.HomeOU.OrganizationalUnitID#
#PerformerQueue.HomeOU.OrganizationalUnitTypeID#
#PerformerQueue.HomeOU.CostCenterCode#
#PerformerQueue.HomeOU.ParentOU.Name#
#PerformerQueue.HomeOU.ParentOU.OrganizationalUnitID#
#PerformerQueue.HomeOU.ParentOU.OrganizationalUnitTypeID#
#PerformerQueue.HomeOU.ParentOU.CostCenterCode#
#PerformerQueue.HomeOU.Manager.PersonTypeElement#
```

Initiator Namespaces

The initiator is the person who orders a service.

Initiator namespaces allow access to initiator information, including:

- Initiator (person) information as defined in the person's profile and accessible via **Organization Designer > People**
- All person/profile information defined for the initiator's supervisor
- Information about the home Organizational Unit of the initiator
- All person/profile information defined for the manager of the initiator's home OU

The namespaces below must be enclosed in hash marks (#) when used in emails, but entered without the hash marks when used in expressions. A complete list of person-type namespaces is listed in the Person-Based Namespaces, on page 343.

```
#Initiator.PersonTypeElement#
#Initiator.Supervisor.PersonTypeElement#
#Initiator.HomeOU.Name#
#Initiator.HomeOU.OrganizationalUnitID#
#Initiator.HomeOU.OrganizationalUnitTypeID#
```

```
#Initiator.HomeOU.CostCenterCode#
#Initiator.HomeOU.Manager.PersonTypeElement#
#Initiator.HomeOU.ParentOU.Name#
#Initiator.HomeOU.ParentOU.OrganizationalUnitID#
#Initiator.HomeOU.ParentOU.OrganizationalUnitTypeID#
#Initiator.HomeOU.ParentOU.CostCenterCode#
```

Functional Positions

Functional positions allow you to access the Person information for individuals who have been assigned to these positions. You can access this information both for the default functional positions, and for those that have been configured for a particular implementation.

Functional positions are defined through the module selection **Organization Designer > Functional Positions**. Each functional position is associated with a particular entity—this may be a service, a service group, or an organizational unit.

Once the position has been defined, you may assign a person to the position through the Positions page for the organizational unit, or on the General tab for the service or service group. Some sample usages include:

Use **Requisition.Customer.HomeOU.BudgetManager.Emai**l to access the email address of the Budget Manager of the customer's home organizational unit.

Use **Performer.HomeOU.ParentOU.Manager.FirstName** to access the first name of the Manager of the parent organizational unit of the task performer's home organizational unit.

Default functional positions may have spaces in the position name, for example, "Budget Manager". The space is omitted with the functional position is used within a namespace reference.

User-specified functional positions may not include spaces. The namespace reference must prefix the functional position with the keyword "Position" as in the following example:

```
#Service.ServiceDefinition.Position.EscalationManager.Email#
```

For all of the functional positions, you have access to all the variables defined in the Person table above. If no person variable is included, the expression returns the person's database ID.

You can access information about Home OU, Parent OU, or Client OU functional positions from any person role that you can access in either the Requisition or the Service context. Information about Service and Service Group functional positions is only available in the Service context.

When using these expressions to include dynamic data in emails and task names, you must add the pound separator ('#') at the beginning and end of the expression, for example:

```
#Customer.HomeOU.Manager.Email#
```

Namespace Variables for Bundled Services

Namespace is a term used to describe a set of valid names that address the data objects used within Service Catalog, exposing these objects to service designers. This allows designers to use these elements in these contexts:

- Within an email, to dynamically resolve the recipient, subject, or references within the email body
- To conditionally execute reviews, authorizations, or tasks in a delivery plan
- In expressions which determine the person or queue to which an authorization or delivery task is assigned
- In task names

Namespaces are hierarchical. Namespace names reflect the structure of the data which defines a service, and the data entered on the service form when that service is requested. The key to manipulating namespace variables is knowledge of the hierarchy which defines Service Catalog data and the contexts in which particular Namespace variables can be used.

Each element in the hierarchy is case-insensitive. The elements are separated by periods ("."). The last element may also be referred to as a "property". All elements before the last one are "nodes" in the hierarchy.

The system provides a number of namespace variables that you can use when working with data in bundled services.

Namespace Variable	Data Type	Purpose/Usage
Service.Bundled	Boolean	True for a requisition entry if the service is a child on a bundle.
Service.IsBundle	Boolean	True for a requisition entry if the service is itself a bundle.
Service.BundledServices	Numeric	Indicates the number of child services belonging to a parent.
Requisition.Services	Numeric	Indicates the number of services in a requisition.
ParentService.Data.DictionaryName.FieldName		Syntax for referring to a data element within a bundle.

- If the value of the site configuration parameter **ShowBundleData** is **On**, ParentService.Data.*DictionaryName.FieldName* is equivalent to: Data.*DictionaryName.FieldName* .
- If the value of **ShowBundleData** is **Off**, ParentService.*Data.DictionaryName.FieldName* is the required syntax for referring to data elements in the parent service.

Lightweight Namespaces

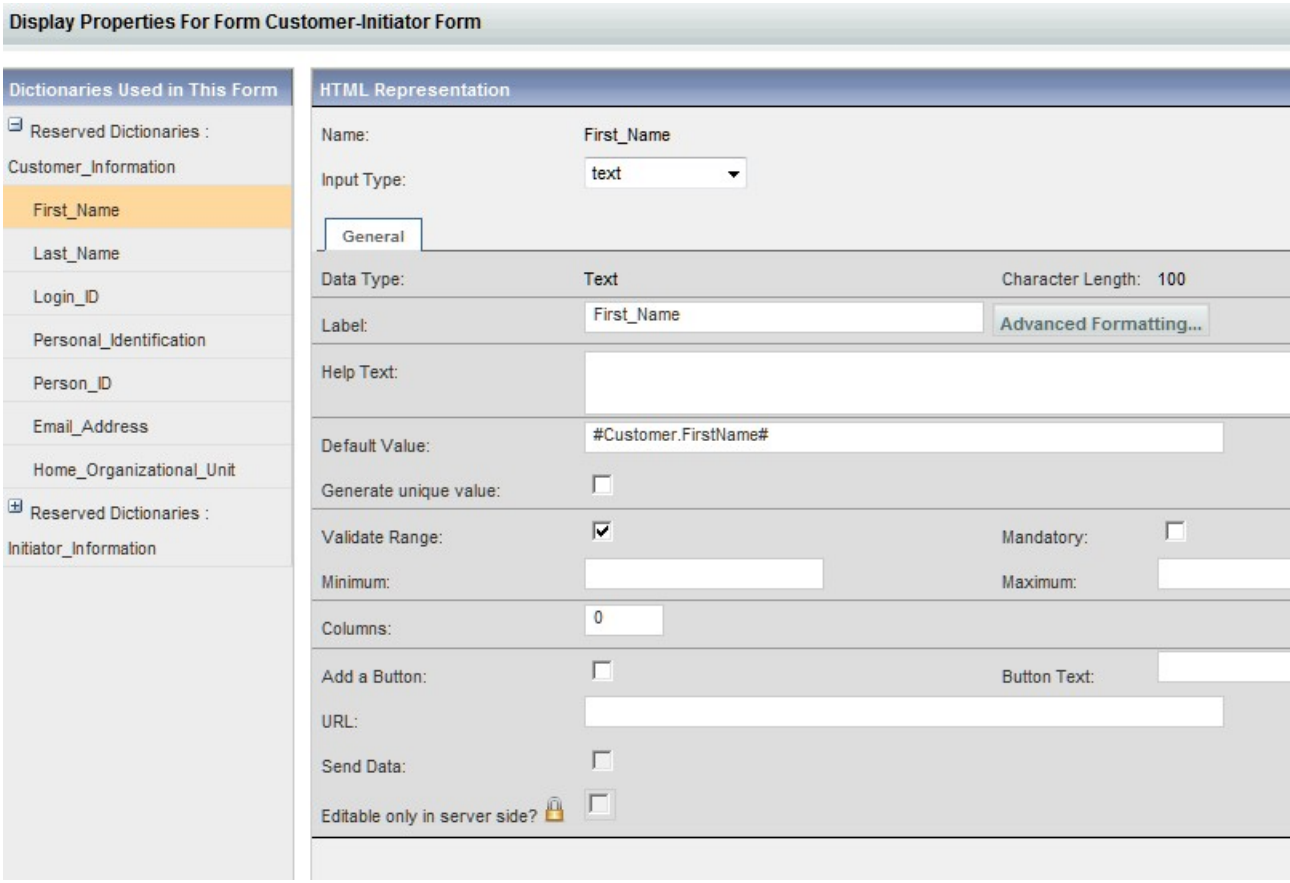
Active form rules need the equivalent of namespaces in order to dynamically access field values to be used or evaluated. For example, the service may need to display an additional dictionary or field if the user entered “Other” in a previous field; the current customer's organization may need to be used as the criteria for building a drop-down list to display valid locations for a service delivery; default values need to be provided for customer and initiator data.

Lightweight namespaces provide these capabilities. They are “lightweight” since only the information accessible to the form (not, for example, details about the service's delivery plan or task performers) can be used within the rules.

Any forms that include person-based dictionaries use lightweight namespaces to provide the values to the form fields, based on corresponding values in fields stored in the profile for the selected person. This includes both the Customer-Initiator form and any user-defined forms. Lightweight namespaces have the format #Customer.**FieldName**#, or #Initiator.**FieldName**#.

Note The use of grid dictionary fields for lightweight namespaces is not supported.

Figure 32: Grid Dictionary Display



The namespace is automatically supplied as the Default Value for the field. If desired, the initial assignments can be replaced.

Lightweight namespaces referring to customer or initiator data can also be used as default values for fields in dictionaries that are not person-based. In this case the service designer will, of course, be responsible for mapping from the dictionary field to the appropriate person attribute. This capability allows you to define dictionaries that contain both person-based and other data.

Customer-Based Namespaces

Customer-based namespaces used for the Customer Information reserved dictionary are summarized below.

Table 91: Customer-Based Namespaces

Dictionary Field Name	Field Type	Lightweight Namespace
First_Name	Text	#Customer.FirstName#

Dictionary Field Name	Field Type	Lightweight Namespace
Last_Name	Text	#Customer.LastName#
Login_ID	Text	#Customer.LoginID#
Person_ID	Number	#Customer.PersonID#
Personal_Identification	Text	#Customer.PersonIdentification#
Email_Address	Text	#Customer.Email#
Home_Organizational_Unit	Text	#Customer.HomeOU.Name#
Title	Text	#Customer.Title#
Social_Security_Number	Text	#Customer.SSN#
Birthdate	Date	#Customer.Birthdate#
Hiredate	Date	#Customer.Hiredate#
Timezone	Text	#Customer.TimeZoneID#
Locale	Text	#Customer.LocaleID#
Supervisor	Text	#Customer.Supervisor.Name#
Employee_Code	Text	#Customer.EmployeeCode#
Supervisor_Email	Text	#Customer.Supervisor.Email#
Supervisor_ID	Number	#Customer.Supervisor.PersonID#
Supervisor_Phone	Text	#Customer.Supervisor.Phone#
Notes	Text	#Customer.Notes#
Company_Street_1	Text	#Customer.DetailedCompanyAddress.Street1#
Company_Street_2	Text	#Customer.DetailedCompanyAddress.Street2#
Company_City	Text	#Customer.DetailedCompanyAddress.City#
Company_State	Text	#Customer.DetailedCompanyAddress.StateProvince#
Company_Country	Text	#Customer.DetailedCompanyAddress.Country#
Company_Postal_Code	Text	#Customer.DetailedCompanyAddress.Zip#

Dictionary Field Name	Field Type	Lightweight Namespace
Building	Text	#Customer.DetailedLocation.Building#
Level	Text	#Customer.DetailedLocation.BuildingLevel#
Office	Text	#Customer.DetailedLocation.Office#
Cubicle	Text	#Customer.DetailedLocation.Cubicle#
Personal_Street_1	Text	#Customer.DetailedPersonalAddress.Street1#
Personal_Street2	Text	#Customer.DetailedPersonalAddress.Street2#
Personal_City	Text	#Customer.DetailedPersonalAddress.City#
Personal_State	Text	#Customer.DetailedPersonalAddress.StateProvince#
Personal_Country	Text	#Customer.DetailedPersonalAddress.Country#
Personal_Postal_Code	Text	#Customer.DetailedPersonalAddress.Zip#
Work_Phone	Text	#Customer.WorkPhone#
Home_Phone	Text	#Customer.HomePhone#
Fax	Text	#Customer.Fax#
Mobile_Phone	Text	#Customer.Mobile#
Pager	Text	#Customer.Pager#
Other	Text	#Customer.OtherPhone#
Main_Phone	Text	#Customer.MainPhone#
Primary_Phone	Text	#Customer.PrimaryPhone#
Primary_Fax	Text	#Customer.PrimaryFax#
Sales_Phone	Text	#Customer.SalesPhone#
Support_Phone	Text	#Customer.SupportPhone#
Billing_Phone	Text	#Customer.BillingPhone#
Other_Contact_Information	Text	#Customer.OtherContactInfo#
Company_Code	Text	#Customer.CompanyCode#

Dictionary Field Name	Field Type	Lightweight Namespace
Division	Text	#Customer.Division#
Business_Unit	Text	#Customer.BusinessUnit#
Department_Number	Text	#Customer.DepartmentNumber#
Cost_Center	Text	#Customer.CostCenter#
Management_Level	Text	#Customer.ManagementLevel#
Region	Text	#Customer.Region#
Employee_Type	Text	#Customer.EmployeeType#
Custom_1	Text	#Customer.Custom1#
Location_Code	Text	#Customer.LocationCode#
Custom_2	Text	#Customer.Custom2#
Custom_3	Text	#Customer.Custom3#
Custom_4	Text	#Customer.Custom4#
Custom_5	Text	#Customer.Custom5#
Custom_6	Text	#Customer.Custom6#
Custom_7	Text	#Customer.Custom7#
Custom_8	Text	#Customer.Custom8#
Custom_9	Text	#Customer.Custom9#
Custom_10	Text	#Customer.Custom10#

Initiator-Based Namespaces

Initiator-based namespaces used for the Initiator Information reserved dictionary are summarized below.

Table 92: Initiator-Based Namespaces

Dictionary Field Name	Field Type	Lightweight Namespace
First_Name	Text	#Initiator.FirstName#
Last_Name	Text	#Initiator.LastName#

Dictionary Field Name	Field Type	Lightweight Namespace
Login_ID	Text	#Initiator.LoginID#
Person_ID	Number	#Initiator.PersonID#
Email_Address	Text	#Initiator.Email#
Personal_Identification	Text	#Initiator.PersonIdentification#
Home_Organizational_Unit	Text	#Initiator.HomeOU.Name#
Title	Text	#Initiator.Title#
Social_Security_Number	Text	#Initiator.SSN#
Birthdate	Date	#Initiator.Birthdate#
Hiredate	Date	#Initiator.Hiredate#
Timezone	Text	#Initiator.TimeZoneID#
Locale	Text	#Initiator.LocaleID#
Employee_Code	Text	#Initiator.EmployeeCode#
Supervisor	Text	#Initiator.Supervisor.Name#
Supervisor_ID	Number	#Initiator.Supervisor.ID#
Supervisor_Phone	Text	#Initiator.Supervisor.Phone#
Supervisor_Email	Text	#Initiator.Supervisor.Email#
Notes	Text	#Initiator.Notes#
Company_Street_1	Text	#Initiator.DetailedCompanyAddress.Street1#
Company_Street_2	Text	#Initiator.DetailedCompanyAddress.Street2#
Company_City	Text	#Initiator.DetailedCompanyAddress.City#
Company_State	Text	#Initiator.DetailedCompanyAddress.StateProvince#
Company_Country	Text	#Initiator.DetailedCompanyAddress.Country#
Company_Postal_Code	Text	#Initiator.DetailedCompanyAddress.Zip#
Building	Text	#Initiator.DetailedLocation.Building#

Dictionary Field Name	Field Type	Lightweight Namespace
Level	Text	#Initiator.DetailedLocation.BuildingLevel#
Office	Text	#Initiator.DetailedLocation.Office#
Cubicle	Text	#Initiator.DetailedLocation.Cubicle#
Personal_Street_1	Text	#Initiator.DetailedPersonalAddress.Street1#
Personal_Street2	Text	#Initiator.DetailedPersonalAddress.Street2#
Personal_City	Text	#Initiator.DetailedPersonalAddress.City#
Personal_State	Text	#Initiator.DetailedPersonalAddress.StateProvince#
Personal_Country	Text	#Initiator.DetailedPersonalAddress.Country#
Personal_Postal_Code	Text	#Initiator.DetailedPersonalAddress.Zip#
Work_Phone	Text	#Initiator.WorkPhone#
Home_Phone	Text	#Initiator.HomePhone#
Fax	Text	#Initiator.Fax#
Mobile_Phone	Text	#Initiator.Mobile#
Pager	Text	#Initiator.Pager#
Other	Text	#Initiator.OtherPhone#
Main_Phone	Text	#Initiator.MainPhone#
Primary_Phone	Text	#Initiator.PrimaryPhone#
Primary_Fax	Text	#Initiator.PrimaryFax#
Sales_Phone	Text	#Initiator.SalesPhone#
Support_Phone	Text	#Initiator.SupportPhone#
Billing_Phone	Text	#Initiator.BillingPhone#
Other_Contact_Information	Text	#Initiator.OtherContactInfo#
Company_Code	Text	#Initiator.CompanyCode#
Division	Text	#Initiator.Division#

Dictionary Field Name	Field Type	Lightweight Namespace
Business_Unit	Text	#Initiator.BusinessUnit#
Department_Number	Text	#Initiator.DepartmentNumber#
Cost_Center	Text	#Initiator.CostCenter#
Management_Level	Text	#Initiator.ManagementLevel#
Region	Text	#Initiator.Region#
Employee_Type	Text	#Initiator.EmployeeType#
Location_Code	Text	#Initiator.LocationCode#
Custom_1	Text	#Initiator.Custom1#
Custom_2	Text	#Initiator.Custom2#
Custom_3	Text	#Initiator.Custom3#
Custom_4	Text	#Initiator.Custom4#
Custom_5	Text	#Initiator.Custom5#
Custom_6	Text	#Initiator.Custom6#
Custom_7	Text	#Initiator.Custom7#
Custom_8	Text	#Initiator.Custom8#
Custom_9	Text	#Initiator.Custom9#
Custom_10	Text	#Initiator.Custom10#

Process Namespaces

Process namespaces are available only for use in emails. They cannot be used in conditions. The Process object includes all namespaces regarding the Customer and Requisition. The Process refers to the current task.

```
#Process.Name#
#Process.Status#
#Process.StatusID#
#Process.StartedOn#
#Process.CompletedOn#
#Process.ExpectedDuration#
#Process.ActualDuration#
#Process.CostCenterID#
#Process.DueOn#
#Process.EscalationLevel#
#Process.TicketID#
```

```
#Process.TicketObjectID#
#Process.DueOnTZ#
#Process.StartedOnTZ#
#Process.DateNow# -- the current date and time in GMT
#Process.Customer.*# -- any Customer element
#Process.Requisition.*# -- any Requisition element
```

Requisition Namespaces

```
#Requisition.URL#
#Requisition.ProcessTrackingID#
#Requisition.ExpectedDuration#
#Requisition.StartedDate#
#Requisition.ActualCost#
#Requisition.ExpectedCost#
#Requisition.RequisitionID#
#Requisition.Name#
#Requisition.Services# -- The number of requisition entries in the requisition
#Requisition.Customer.*# -- Any Customer element
#Requisition.ClientOU.*# -- Any ClientOU element
#Requisition.Initiator.*# -- Any Initiator element
```

Message Namespaces

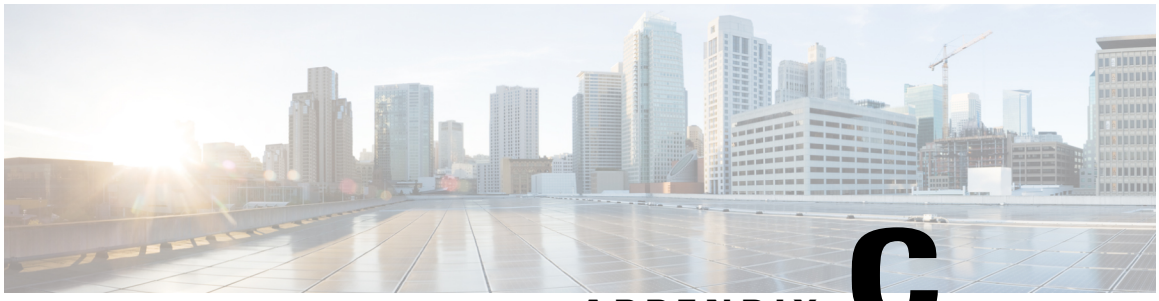
Message namespaces are available only for use in emails generated as a result of a failed Service Link task. They cannot be used in any other emails or in any conditions. The Message elements may be helpful in diagnosing the Service Link failure, and may eliminate having to consult the log files for diagnostics.

```
##Message.ExternalContent# -- Co
```

Table 93: Field Types for Active Forms

Type	Description and Active Form Implications
Text	Default data type, supports alphanumeric data; should be used for fields to be rendered as single- and multi-line text.
Number	Data type for all numbers, including integers and whole numbers; it is critical to specify the precision in the Decimals column, as the application will validate decimal precision as well as length.
Account	Documentation only; data treated as alphanumeric.
Date	Data compatible with the database's native datetime type, but display restricted to the date; calendar widget supplied for data entry.
Boolean	Data object whose possible values are “true” and “false”, presented as “Yes” and “No”.
Phone	Documentation only; data treated as alphanumeric.
SSN	Documentation only; data treated as alphanumeric.

Type	Description and Active Form Implications
Money	Data validated to contain only a valid number; monetary symbols or commas cannot be typed; accepts numerical characters and up to 3 decimal places.
Person	Data validated against a person ID in the personnel profiles; a Person Search dialog box is available via a “Search” button automatically rendered on the service form. The Person data type is provided primarily for backward compatibility; if this capability is required, a person-based dictionary should be created.
URL	Data stored as alphanumeric; a saved value is represented both as text and as an HTML representation of the value, providing a link to the specified URL.
Date and Time	Data stored as a date and time; calendar widget supplied for data entry contains a time-selection widget.



APPENDIX C

Form Rule and ISF JavaScripts UseCase Analysis

This appendix contains the following topics:

Form Rule and ISF JavaScripts Use Case Analysis

In a complex service catalog you can easily have hundreds of instances where active form rules or ISF functions are used to enhance the service form's usability and interactivity.

1. Perform a use case analysis in order to determine what rules are required and when they will fire.
2. Perform a detailed design analysis to determine what dictionaries and fields are needed to support the specified use cases; how the dictionaries should be combined into forms; and what tools you will need to implement the requirements.
3. Define the dictionaries and forms. Specify the display and access control properties of the forms.
4. Write active form rules and sequence these to meet the detail design criteria. If required, write JavaScript functions and libraries; attach the functions to the appropriate HTML event or events.
5. Define the services which use the reusable form components previously defined.
6. Test and make fixes as required.

This section includes some examples of both rules and ISF code integrated into a service form, and the sorts of implementation decisions made to meet the requirements. These examples illustrate some common design patterns, such as:

- Adjust the appearance or behavior of dictionaries based on a specific task
- Show/hide dictionaries or fields when a service form is loaded or when the user changes the value in another field
- Manager lookup on change for Person fields
- Drill-Downs
- Ensure that Select Lists are populated during the delivery moment
- Hide dictionaries and clear fields if their value is not relevant to the current context

Use Case Analysis

Review user requirements to determine when a service form's default behavior and appearance need to be supplemented by rules or ISF. While you may document these requirements discursively, a better format might be a table which explicitly defines the behavior and its triggering events. An example is shown below.

Table 94: Example

Service Name	Use Case Description	Moment
All	If a person is high profile, display his status as read-only; if not, hide the status	Ordering

Where:

Service Name is either "all" or a list of the services which are affected by the requirement.

Use Case Description specifies, in language accessible to business users, the desired behavior of the service form.

Moment is either "all" or one or more of the moments that occur during the fulfillment of a requisition.

Try to collect all the use cases for the current project, so you can estimate the scope of the work involved.

Detailed Design

In the detailed design phase the programmer must review the use cases previously defined and specify, at a high level, the rules or JavaScript components that need to be written, and the triggering events for these rules/functions. The design incorporates a detailed specification of the forms, dictionaries and fields involved. In particular, the analysis should cover:

- Will you need to use ISF to supplement the form rules? If so, ensure that your development environment is set up to support ISF (JavaScript) development, testing and debugging, and that personnel are available with the requisite skill set.
- Will you need to use data retrieval rules or SQL option lists? If so, ensure that the development environment includes a means to test and debug SQL queries; that datasources are defined to allow you to access the desired data; and that personnel are available with knowledge of the structure of the source data and the requisite skill set.

Scenario #1: Dynamically Adjusting Form Appearance and Behavior

Functional Requirements

In requesting a database to be created, the user must choose whether the database server type is Oracle, SQLServer, or some "Other" database. If some other, nonenterprise standard database is chosen, additional information must be gathered from the user; otherwise, the additional fields are hidden.

Scenario:

Table 95: Dynamically Adjusting Form Appearance and Behavior

Is the current value of the DatabaseType field in the Database dictionary equal to “Other”?	If so, display the Description field in the same dictionary, and require the user to enter additional information.

Dictionary/Form Design

Define a dictionary called, for example, NewDatabase.

- The dictionary includes a field named DatabaseType, rendered as a radio button which allows the user to designate whether the database is Oracle, SQLServer, or Other.
- If the user selects “Other”, the type of database must be provided and is mandatory. Otherwise, this field is hidden.

Detailed Rules Design

This use case can be implemented entirely through the use of conditional rules.

The AIT_DATABASE.DatabaseServer field needs rules which fires when the field's value is changed.

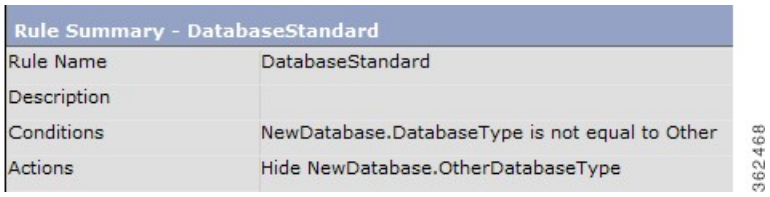
- If the value which the user selects is 'Other', a rule must display the “Other”-dependent dictionary fields and ensure that all such fields are mandatory.
- If the value which the user selects is not 'Other', a rule must ensure that the “Other”-dependent fields are not visible.

Unfortunately, this release of Service Catalog does not include if/then/else logic in the rules, so you will need two rules to implement this design.

Conditional Rule Implementation

Define two conditional rules as follows:

Figure 33: Conditional Rule Implementation



Rule Summary - DatabaseOther	
Rule Name	DatabaseOther
Description	
Conditions	NewDatabase.DatabaseType is equal to Other
Actions	Show NewDatabase.OtherDatabaseType Make Mandatory NewDatabase.OtherDatabaseType

362465

The Triggering Events

When should this rule be applied? In more technical terms, what is the “triggering event”, during the course of a user entering data in the service form, when this rule should “fire”?

It seems obvious that the rule needs to be executed when the user selects a value from the DatabaseServer radio button. So, use the Active Form Behavior tab for this form to associate these two rules with the “When the field changes” event. In this case, it really doesn't matter what order the rules are applied in—either one or the other will fire, but not both.

Build the Service Definition

To test this scenario, you need to complete the definition of the service which contains the active form you have just defined. You could initially create a service containing just that form, for a quick-and-dirty test, but this is clearly just a first step. Forms and their rules can interactive with other forms and their rules, so the best test is the most realistic.

Test

It's easiest to test this scenario by using several browser windows. Keep Service Designer open in one, with the Active Form Components option displayed. Then, start a new session, log in as a My Services user who has permission to order the service. See what happens.

Testing Follow up and Results

The rule as previously defined should work correctly. However, the implementation is incomplete, as the behavior is only triggered when the customer changes (or initially selects) the operating system. If the form is saved and reviewed, or submitted and displayed in a subsequent system moment when the dictionary is editable, the saved value of the DatabaseServer field must be used to adjust the appearance of the form. Therefore, an additional triggering event is required, to fire when the form is loaded.

Scenario #2: Manipulating Customer and Initiator Information

Functional Requirements

Usage needs to take into account two different scenarios that affect the one Customer dictionary and the one Customer-Initiator Form:

Scenario:

Table 96: Manipulating Customer and Initiator Information

If the service is delivered electronically ...	Display the standard set of fields required for collecting Customer information.
If the service needs to be delivered to a specific physical location ...	In addition to the standard fields, display fields pertaining to the Customer's location, as stored in the Customer profile, but allow him/her to specify an alternate service location, for example, if the location on file is out of date or the customer is temporarily at a different location.

Dictionary Design

The design can be implemented using three dictionaries:

- Design the Customer dictionary so that it includes all of the fields that must be available in both types of services—those fields that are always required and those that are required only when the requestor must specify or confirm the location where the service must be delivered.
- Design a Perform Work dictionary that will only be displayed for services to be delivered in person. The form has one field, which asks if the work is to be performed at the customer's location or a different (service) location.
- Design a Service Location dictionary, to be filled in, by default, with the location information from the Customer's profile, but which can be overridden if the requestor indicates that this is different than the location in the profile.

Form Design

The Customer and Initiator dictionaries are typically read-only in all moments. That is, data is displayed from the person's profile and cannot be changed by either the customer or any task performer.

There are at least three ways to make all fields in a dictionary read-only:

- Use the Access Control tab for the form to specify that the dictionary is viewable (not editable) for appropriate moments and participants. This control cannot be overridden by either rules or ISF—a viewable dictionary can be hidden, by it can't be made editable. And field values display as boilerplate, rather than enclosed in input fields. (Q: Would lightweight namespaces work?)
- Make the dictionary editable in the Access Control tab, but define the default fields as having an HTML input type of read-only and the location-related fields as having an input type of hidden. Both hidden and read-only fields are supplied values from an associated lightweight namespace.
- Make the dictionary editable in the Access Control tab, assign appropriate input types to the standard fields, but create a rule, applied when the form is loaded, to make All Fields in the dictionary read-only. This rule would not affect any hidden fields.

Option #2 makes the most sense in this case. Fields could potentially be made writeable, for example, if users are allowed to update out-of-date data from their person profile. And there is no extra rule to keep track off. That takes care of the Customer-Initiator form.

You also need a form in which the PerformWork and ServiceLocation dictionaries are used. Call this form ServiceLocation. (Q: Naming conventions?) The field in the PerformWork dictionary can be implemented as a check box? (Is work performed at the Customer Site?) The ServiceLocation dictionary would have fields corresponding to the person fields included in the Customer dictionary.

Detailed Rules Design

Now you need a rule for the services where a service location is required. The rule is needed to:

- Copy the default location values from the Customer dictionary to the ServiceLocation dictionary. This can be done in an onLoad event.
- If the user says that a different service location is needed, make the ServiceLocation fields writeable. This needs to be done in an onChange event for the PerformWork field. (Q: Nomenclature, again).

The rules are included in the ServiceLocation form.

Conditional Rule Implementation

Does anybody else out there remember COBOL? Most of the time, coding in COBOL was painfully verbose, but it had one really neat command: COPY CORR(esponding). The COPY CORR command copied all values, by name, from one structure to values with corresponding names in a second structure. It would be great to COPY CORR Dictionary1 TO Dictionary2, but that is not possible. So, the first rule (ServiceLocation_onLoad) should be applied in the ordering moment, and Copy Value for all location fields.

The second rule (PerformWork_onChange) is straightforward, reminiscent of the rule written in the initial scenario.

Build the Service Definitions and Test

You'll need two services to test this scenario—one that doesn't include the ServiceLocation form, and one that does.

Scenario #3: Securing Sensitive Data

Functional Requirements

The service requires the user to specify a Social Security Number, credit card information, or other sensitive data that should be available only to people who “need to know”, not to every authorizer or task performer involved in fulfilling the service. The data needs to be protected from attempted hacks as well.

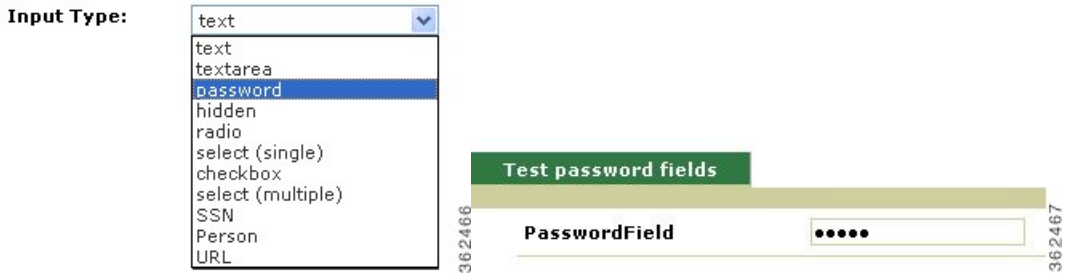
Approach 1 – Hide the Dictionary and Fields

By default, when a dictionary's display property is set to “none” in Service Designer, the dictionary and the values for any fields previously entered are not part of the generated service form seen by users. (This setting may be overridden for backward compatibility with behavior of Service Catalog versions prior to 2007. Be sure to check with your Service Catalog Administrator to verify the setting for dictionary.permission.none.show

newScale property.) Therefore, the data in the field would not be visible, even to users savvy enough to view the page source from the browser.

To allow the field value to be initially provided by the user, the field must (obviously) be visible on the form, and the dictionary display setting set to Read/Write. The field's HTML representation can be set to “password”. The field value will then be displayed as a series of asterisks.

Figure 34: Hide the Dictionary and Fields



So, the value is protected from people doing shoulder-surfing. If you view the source of the page, however, the value of the field is visible. The value of the field is not accessible to ISF’s getValue() function—“undefined” is returned. The value of the field is available to the DOM, that is, via JavaScript methods such as:

```
document.getElementById('Dictionary.PasswordField').value.
```

The value of the field is also accessible to Service Link.

Password field values are hidden from user when sent back to the browser, as in the case of reviewing an existing request or performing a task with a password field value in the service form.

Approach 2 – Use Encryption

Encryption can be used in conjunction with some of the practices above to better secure sensitive data. Instead of (or in addition to) using a password field, use JavaScript functions to encrypt the sensitive data before you save it and decrypt it before displaying it on the form. An open-source algorithm on the web called “tiny encryption algorithm” could be used.

```
// Algorithm: David Wheeler & Roger Needham, Cambridge University Computer Lab
//    http://www.cl.cam.ac.uk/ftp/papers/djw-rmn/djw-rmn-tea.html (1994)
//    http://www.cl.cam.ac.uk/ftp/users/djw3/xtea.ps (1997)
//
// JavaScript implementation: Chris Veness, Movable Type Ltd
```

If you define the function to encrypt/decrypt code in a Script, it would be visible in the form if the user did a View Source. However, if you include the function in a library, it would not be visible to users who attempt to view the source, and not subject to reverse engineering.

Approach 3 – Use Secure String

Approach 4– Use Server-side Rules

Data elements that are governed by entitlement or RBAC permissions should have values re-validated on the server-side to prevent the breach of permissions by malicious attempts to manipulate form data before they

are sent to the server. Always enable the implicit validation for data retrieval rules unless the rules are executed post-submission. See the Guidelines for Designing Optimal Service Forms section for more information.

Scenario #4: Computing a Value in a Form

Functional Requirements

The service requires the user to specify a “Quantity” and “Price” for an item to be ordered. The service should compute the “Extended Price”, and show this value on the service form.

Dictionary/Form Design Requirements

A dictionary, call it SVC_PRICE, needs to be created, containing three Number fields. The fields may have decimal precision or not, according to detailed requirements. The dictionary is included in the service form and is writeable for the customer in the ordering moment. All fields are rendered as text fields.

ISF Detailed Design

The Total field needs to be read-only, since users are not allowed to enter a value—it must be computed. The computation needs to take place when the user changes either the Price or the Quantity.

This task requires three custom events:

- The SVC_PRICE_onLoad event sets the ExtendedPrice field to be read-only.
- The SVC_PRICE_Quantity_onChange event computes the ExtendedPrice.
- The SVC_PRICE_Price_onChange event also computes the ExtendedPrice.

JavaScript Code and Events

The first task is better accomplished when the form loads the first time. To do this, create a function in Script Manager called SVC_PRICE_onLoad:

```
SVC_PRICE_onLoad ()
{
  serviceForm.SVC_PRICE.ExtendedPrice.setReadOnly(true);
}
```

This code is associated to the “When the form is loaded (browser-side)” event in the Behavior tab.

The second task is to compute the ExtendedPrice based on the Quantity and Price. Use the “When the item is changed” event for both Quantity and Price. The code needs to verify that both fields have valid values, and then compute the ExtendedPrice. The function is called SVCPRICE_Price_onChange.

```
SVC_PRICE_Price_onChange ()
{
  serviceForm.SVC_PRICE.Total.setReadOnly(true);
  var Price = serviceForm.SVC_PRICE.Price.getValue()[0];
  var Quantity = serviceForm.SVC_PRICE.Quantity.getValue()[0];
/* Blank out current value (if any) of ExtendedPrice */
  serviceForm.SVC_PRICE.Total.setValue(['']);
/* Check is required, since check for Numeric data happens only on Submit. */
  if (isNaN (Price))
```

```
  {
    alert ('Price is not a number');
    serviceForm.SVC_PRICE.Price.setFocus(true);
    return;
  }
  if (isNaN (Quantity))
  {
    alert ('Quantity is not a number');
    serviceForm.SVC_PRICE.Quantity.setFocus(true);
    return;
  }
  var Total = Price * Quantity;
  serviceForm.SVC_PRICE.Total.setValue([Total]);
}
```

Refactored JavaScript Code

The code above, defined as a Script in Service Designer, could be attached to the onChange event for two different fields: the Price and the Quantity. In fact, this may be an efficient way to initially test the code. However, this approach, with a function name that doesn't reflect this usage and that doesn't use a library, is harder to maintain in the long run. Therefore, the following refactoring is recommended:

- Edit the function code, renaming the function something generic, like “ComputeExtendedPrice”, and extract the code from Scripts, placing it into a custom library for your application.
- Be sure the custom library is defined in **Scripts > Libraries** and that it is included in the service form when this function is required.
- Create two Scripts, which look like the following:

```
SVC_PRICE_Price_onChange ()
{
  ComputeExtendedPrice();
}
SVC_PRICE_Quantity_onChange ()
{
  ComputeExtendedPrice();
}
```

- Attach these functions to the onChange events of the Price and Quantity fields, respectively.
- Remember to upload the revised library to the application server.

Scenario #5: Formatting Two Fields in a Form

Requirements

Create a JavaScript function to format two fields in a form: A social security number and a phone number. The SSN must verify that there are 9 digits, and is formatted like “999-99-9999” Any formatting done by the user is ignored. The phone number must have 10 digits and it is formatted like (999) 999-9999.

JavaScript

Create two functions, one called **formatSSN** and the other called **formatPhoneNo**. Put both functions in a file called “isfprimerlib.js”. As documented in the ISF Coding and Best Practices, on page 123, this file may

reside on any directory on the application server beneath the RequestCenter.war directory; by convention, ISF libraries are placed on a directory named “isfcode”.

Create a library reference to your JavaScript file in Script Manager. Be sure to specify the library in the Libraries tab for a JavaScript function that are attached to your service.

The resulting code is:

```
function getOnlyDigits (inValue)
{
  var outValue = '';
  var aChar;
  for (i=0; i < inValue.length; i++)
  {
    aChar = inValue.charAt (i);
    if ('0' <= aChar && aChar <= '9')
    {
      outValue = outValue + aChar;
    }
  }
  return outValue;
}
function testValueLength (inValue, inLen, obField, fieldName)
{
  if ((inValue.length > inLen) || (inValue.length < inLen))
  {
    alert (fieldName + ' must have ' + inLen + ' digits and it has ' + inValue.length);
    eval('serviceForm.'+obField).setFocus(true);
    return false;
  }
  return true;
}
function formatSSN (obField)
{
  var SSNString = getOnlyDigits (eval('serviceForm.'+obField).getValue()[0]);
  if (testValueLength (SSNString, 9, obField, 'SSN'))
  {
     eval('serviceForm.'+obField).setValue([SSNString.slice (0,3) +
       '-' + SSNString.slice (3,5) + '-' + SSNString.slice (5)]);
  }
}
function formatPhoneNo (obField)
{
  var phoneString = getOnlyDigits (eval('serviceForm.'+obField).getValue()[0]);
  if (testValueLength (phoneString, 10, obField, 'Phone Number'))
  {
    eval('serviceForm.'+obField).setValue(['(' + phoneString.slice (0,3) + ') '
      + phoneString.slice (3,6) + '-' + phoneString.slice (6)]);
  }
For the field SSN create a function called Customer_SSN_onChange
 that calls formatSSN
:
Customer_SSN_onChange ()
{
  formatSSN('Customer.SSN');
}
```

For the PhoneNo field, create another function called **Customer_PhoneNo_onChange**, that calls formatPhoneNo when the value in the field changes. Just to show an alternative implementation, these functions pass the name of the field and acts “by-reference” rather than “by-value”.

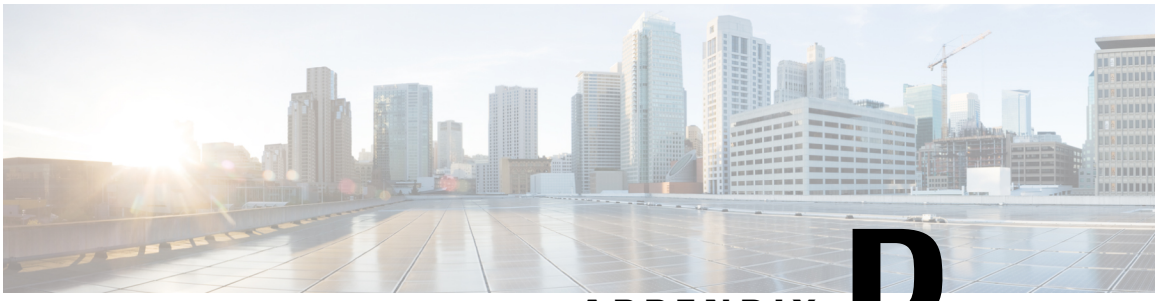
```
Customer_PhoneNo_onChange ()
{
  formatPhoneNo ('Customer.PhoneNo');
}
```

Both functions are called when the value in its respective field changes.

Server-Side Associated Controls

This section explains how to transfer service form data to and from an outside server through an HTTP request. The purpose of this functionality is to allow the Service Designer to use Web widgets that reside on Web servers separate from the application server. This section only deals with the actual transfer and handling of the service form data.

The service form data is sent to the outside Web widget via an HTTP form post. The data is passed in the **wddxdataform** form in the WDDXData variable as a WDDX packet. WDDX is an XML schema for storing data structures in a serialized packet. This allows the data to be passed into or out of Service Catalog independent of the programming language used. You can use an HTTP request to a Service Catalog page to place the resulting data set back into the service form



APPENDIX D

Key Terms

This appendix contains the following topics:

Key Terms

The following are some key terms to be familiar with when using Service Designer.

Table 97: Key Terms Table

Term	Definition
Active Form Components	Reusable forms that are built from one or more dictionaries and configured for use in one or more service forms. Active form components are the building blocks of a service form, and dictionaries, along with active form rules, are the building blocks of a form component.
Authorization	A task during which the performer reviews, approves, or rejects the service requested.
Permission	A Permission grants rights to act upon an object. For example: Order for Others (of a person or OU).
Search Facets	Search Facets allows the service designer to specify one or more facets, which each service can assign a value to. For example, for a “Available Location” facet, a service may have the values, Europe and Americas, while another service has the value Japan. An end user can narrow down the list of available services in their view but select the facet value they want.

Term	Definition
Customer	The individual to whom a service is being delivered. The customer and the initiator are typically the same person, except in cases when an initiator orders a service for other people, such as an executive assistant ordering a service for the executive.
Dictionary	Reusable groups of fields created for use on a form component that may, in turn, be used in multiple service forms. A dictionary defines the individual data items that are used in a service request.
Email Template	A standard email that can be sent upon the initiation, completion or other milestone associated with of a particular task. Email templates can be associated with particular services.
Escalation	Notifications triggered at specified intervals after a task is not completed by its due date.
Initiator	The requestor of, or person who orders, a service from the service catalog. The customer and initiator can be the same person.
Interactive Service Forms (ISF)	A JavaScript API that allows designers to customize the behavior of a service form using JavaScript. ISF coding supplements the use of active form rules to add further interactivity and a richer user interface to the service form.
Moments	Service Catalog manages the events from ordering through service completion as a sequence of discrete system moments or phases. The completion of one moment is the prerequisite for the beginning of the next moment in the sequence.
System Moment	Tracks what point of the requisition life cycle the requisition is in. The system moment evolves from ordering state to finally service completed state, as shown below: Ordering > Pricing > Authorizations > Service Delivery > Service Completed
Service	A process packaged and presented as a product the end user can order/request.

Term	Definition
Service Link	The module that defines integrations with external systems; such integrations can be used within a delivery plan as external tasks, reviews, or authorizations.
Service Group	A folder that contains a group of similar services. Services are organized into service groups as a way to facilitate the service design process.
Service Item	A product or intangible asset that can be provisioned via a service request and whose history can be tracked in the My Services and Service Item Manager modules.
Service Team	The individuals (or groups of individuals) who perform the steps to deliver the service. Service teams are organizational units created and managed in the Organization Designer module. It is critical to specify the appropriate service team for a service group. “Service Team” is listed as a default participant when configuring dictionary Access Control. This allows members of the service team to view or edit dictionaries in the service delivery moment of service fulfillment. All members of the service team are automatically able to perform work on all tasks defined in services in their service group. Other service teams need to be listed as “Additional Participants” in the Access Control subtab for the form components used in the service in order to perform tasks in the service.

Term	Definition
Functional Position	Functional positions are associated with service groups and their member of the specified service team who is currently assigned to that position. A functional position is a job description associated with one of the following: • Organizational Unit • Service • Service Group In Service Designer module, you may assign functional positions to be performers of activities in the authorization, review, and delivery processes to avoid referring directly to people or queues. You may also use functional positions to identify the recipients of escalation notifications. For example, an email may be directed to the "Escalation Manager" for a particular organization or service group, rather than being routed to a specific person or queue. Or you may simply use some of the functional positions to document the person or other entity responsible for a particular service group or service.

INDEX

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W