



# Managing Content Deployment

---

This chapter contains the following topics:

- [Managing Content Deployment, page 1](#)

## Managing Content Deployment

### Overview

You can use Catalog Deployer for content deployment and configuration management, for service designers, catalog publishers, finance administrators, and organization building resources to migrate application entities between development, test, and production sites.

Catalog Deployer provides customers with a change management process and history, allowing for reliable control over content changes, and changes to the organizational entities used by all modules of Service Catalog.

Catalog Deployer also supports the deployment of preconfigured services packaged within a branded content library by Cisco. Such services can be used as-is or as the templates for customizing the service definition to the organization's requirements, significantly reducing the time and effort required to implement actionable service catalogs.

### Content Deployment using Catalog Deployer

Catalog Deployer offers two methods of content deployment.

- Source sites can use Catalog Deployer to extract or assemble a package of service definitions and entities for transmission to, and deployment on, a target site. In this case, all operations can be performed within the Catalog Deployer module, and there is no need for external programs.
- Alternatively, a package can be produced for export. Exported files are imported via Catalog Deployer into the target site's instance of Service Catalog. Since these are XML files, in text format, they can also be stored in a configuration management or source code control system. Branded content libraries are delivered in the form of an export file, so standard Catalog Deployer facilities may be used to import and deploy services available in the library.

Like all other Service Catalog modules, Catalog Deployer can be permissioned for use. Users may have abilities to view deployment history; to create and assemble a package for deployment; to import or deploy a package to a particular site; or the combination of these capabilities that best fit the users' responsibilities. All such capabilities may be assigned via standard roles or custom roles in Organization Designer module.

## Key Features and Functionality

Key features include:

- Simplicity of use via the user interface. Packages for deployment are created, and can be populated, transmitted to a target site, or exported to the file system.
- Two methods of content transfer: XML package transmission across sites or file-based transmission via export/import functionality.
- Service deployment (inclusive of related entities) on one or more target sites.
- Organizational deployment (OUs, groups, queues, roles, people, functional positions) on target sites.
- Email template and Service Link agent deployment on the target sites.
- Change management support (optional segregation of duties between content developers/managers and catalog publishers).
- Hot deployment—the ability to deploy content while the application is available to users.
- Online options to view extraction and deployment history.
- Support of site protection levels and entity homes.
- Ability to preview a service definition before it is deployed to a target site from a branded content library.

Catalog Deployer may be used at the beginning of a Service Catalog effort to provide template content which can form the basis of a customized service catalog. In addition, Catalog Deployer may be used in the Build and Maintenance phases of a development effort, to promote tested content from development to other environments and to synchronize multiple environments.

## Configuration Management

The following sections discuss capabilities relevant to using Catalog Deployer for release and configuration management:

- [Application Roles and Capabilities](#), on page 6
- [Configuring Catalog Deployer](#), on page 9
- [Catalog Deployer Packages](#), on page 15
- [Sample Deployment Scenarios](#), on page 29

## Service Catalog and Portfolio Development

The following sections of this chapter discuss capabilities relevant to using Catalog Deployer to install Cisco content libraries to provide the basis for a service catalog and service portfolio:

- [Catalog Deployer Packages](#), on page 15 (only the sections on importing and deploying content.)

- [Branded Content Libraries](#), on page 36

## Catalog Deployer Architecture

Catalog Deployer provides database-neutral data-transfer infrastructure and interfaces, organized around the logical entities found in the application. Examples of logical entities are Service Definitions, Data Dictionaries, People and Organizational Units. A complete listing is available later in this section. There is however limited support for custom content across different databases and application environments. See the [Unsupported Entities](#), on page 5 for the list of entities not supported by Catalog Deployer.

An understanding of Implementation, Site, and Entity Home, described in [Table 10: Key Terms](#), on page 40, is critical to the operation of Catalog Deployer.

In particular, Logical Entity Home Sites are an integral part of configuration management with Catalog Deployer. Though optional with Catalog Deployer, creating a system of Home sites for logical entities enables management of both the referential integrity of logical entities across sites, and the integrity of configuration information across sites.

The idea behind Logical Entity Home sites is that certain sites will have a more authoritative version of the data for a logical entity type than others. For example, for customers using their LDAP directories to authenticate and gather information about users, data on People and Organizational Units are most accurate on Production. Therefore, the Production site would be the Home site, the site of record, for the People and Organizational Units. If People can be created, or Organizational Units edited, on sites other than Production, deploying this data to other systems will get them out of sync with the system of record, and the quality of data across the implementation will degrade.

Thus, the application framework is designed to allow protection levels to be assigned to logical entities to keep users from modifying these entities on sites other than the entities' Home sites. The site administrator can choose how much protection to provide for logical entities by choosing one of the four settings, as described in [Configuring Catalog Deployer](#), on page 9.

Catalog Deployer deploys permissions for entities as part of the entity. When permissions are removed from the entity in its Home site, the application does not retain deletion stubs (or a transaction log) that Catalog Deployer may use to replicate this removal. For example, when the permission to order a service is removed from an Organizational Unit on a Service Definition, Service Catalog does not retain this fact, it simply removes the permission data.

When you deploy a service whose permissions have been changed, all associations between the service definition and its permissions are dropped in the target site and recreated according to the permissions in effect in the source site. However, if the permission was granted to a custom role or group, and the role or group was deleted from the source site, the role or group still exists in the target site. Catalog Deployer cannot propagate entity deletions.

## Supported and Unsupported Entities for Catalog Deployer

The following table lists all logical entities supported for Catalog Deployer and their associated application module.

**Table 1: Supported/Unsupported entities**

Module	Supported Entities
Service Designer	<p>Service definitions (Offer, Form, Form Sections, Plan, Authorizations, Permissions)</p> <p>Component entities automatically deployed with service definitions:</p> <ul style="list-style-type: none"> <li>• Service Groups</li> <li>• Dictionaries and Dictionary Groups</li> <li>• Active Form Components and Component Groups</li> <li>• Keywords, Categories, and Presentation Elements</li> <li>• Script Functions and Libraries</li> </ul> <p>Entities referenced by service definitions:</p> <ul style="list-style-type: none"> <li>• Email Templates</li> <li>• Organization Designer entities</li> <li>• Service Link agents and transformations</li> </ul>
Service Item Manager	<p>Component entities automatically deployed with service definitions:</p> <ul style="list-style-type: none"> <li>• Service Items (if referenced by a Service Item-Based Dictionary or by a table-based data retrieval rule)</li> <li>• Standards (if referenced by a table-based data retrieval rule) Service Items and Standards can also be deployed separately.</li> </ul>
Service Link	<ul style="list-style-type: none"> <li>• Agents</li> <li>• Transformations associated with the chosen agent are also deployed.</li> </ul>

Module	Supported Entities
Organization Designer	<ul style="list-style-type: none"> <li>• Queues</li> <li>• Organizational Units</li> <li>• Groups</li> <li>• Roles</li> <li>• Functional Positions</li> <li>• People</li> </ul>
Administration	<ul style="list-style-type: none"> <li>• Email Templates</li> <li>• Target Type allows you to export stack components for Infrastructure/Application templates into a new environment. Target Type is displayed by default in custom packages.</li> </ul>
Demand Management	<ul style="list-style-type: none"> <li>• Account Definition</li> <li>• Agreement Templates</li> <li>• Billing Rates</li> </ul>

Catalog Deployer copies the entities listed above from the source site database to the target site database. Catalog Deployer does not move or copy information stored in the file system. Catalog Deployer copies the definition of libraries associated with JavaScript functions.

### Unsupported Entities

Catalog Deployer migrates logical entities which are stored in the transactional database between two sites which are implemented using the same version of Service Catalog. The entities are extracted to a text stream, formatted in .xml, which is part of a deployment package which also includes the specifications (options) used to create the package. That package may be extracted from the source database, to serve as a backup mechanism as well as the source for deploying the entity definitions into the target database of another site. The entity definitions are deployed unchanged into the target environment.



**Note**

Catalog Deployer **does not support** deploying images (presentation elements) in .bmp format. Service Designer now prevents such images from being specified. Any legacy images should be converted to an alternate format, such as .jpg or .gif, optimized for presentation on the web.

Catalog Deployer has no effect on any components of Service Catalog other than those logical entities. Changes to some of these components are not part of the configuration management scenario handled by Service Link—the synchronization of customer-designed and configured configuration items across an implementation—so need not be considered further here. For example:

- Software components must be installed and configured via the Service Catalog Installer.
- The contents of the data mart and reporting tables must be created via Service Catalog Installer and populated via Extract-Transform-Load (ETL) processes.
- The schema is upgraded as part of the Service Catalog installation.
- Any customized Service Catalog components, or additional components, need to be installed on all servers via the “Customizations” option in the Service Catalog Installer and the procedures documented in this guide. Such customizations typically include support for custom mappings used in directory integrations, and any APIs provided by the Advanced Services organization.

However, additional configuration items may have to be deployed in conjunction with changes to the logical entities which are handled by Catalog Deployer. These are summarized in the following table and discussed in more detail in conjunction with the logical entity affected.

**Table 2: Not supported entities**

Configuration Item	Additional Artifacts to be Controlled and Migrated
External Dictionary	DML and DDL scripts run in the database
Datasource	Datasource specification to reference an external dictionary, a SQL-based option list or a table referenced in a data retrieval rule
Data Retrieval Rules	SQL Statement directly entered into the rules (may not be compatible across different database types)
ISF Script Library	Library (JavaScript) text file deployed on the application server
Custom Adapter	Deployment file produced by the Service Link Adapter Development Kit (ADK)

## Application Roles and Capabilities

The key roles are defined as follows.

**Table 3: Key Roles**

Role	Definition
Catalog Publisher	Creates and maintains service catalogs and is responsible for deploying catalog content to the runtime application, configuring the look and feel and structure of catalogs, and for updating the deployed catalog content on an ongoing basis as the service definitions and delivery plans change.

Role	Definition
Service Designer	<p>Designs service definitions at a customer site. Service designers have significant subject matter knowledge of the services provided to the customers of the customer IT team, and are proficient in the usage of Service Designer and Organization Designer. Services designers should be moderately technical, but are typically business analysts rather than engineers.</p> <p>Service designers may frequently need to create and modify service definitions within their development site. Some are allowed to use the Catalog Deployer for publishing their work to a staging or production site.</p>
Organization Builder	<p>Designs Organization Designer organizational entities at a customer site. Organization builders have significant subject matter knowledge of the organization's needs with regard to configuring the organizational units, groups, and roles necessary to successfully deploy and use Service Catalog.</p> <p>Organization builders should be proficient in the use of the Organization Designer module. Organization builders do not need to be highly technical, but should be an application administration IT resource.</p>
Site Administrator	<p>Performs application permission administration at a customer site. This person may be the first user of the system and is able to access all facets of the back end of the application, such as setting Global Configurations; managing lists such as languages and billing categories.</p>
Change Manager	<p>Approves change requests for the implementation at a customer site. This person is typically responsible for the stability of the production site and needs to understand all changes prior to their deployment to the production site. This role is optional, but is required in those customer sites that have established formal change control processes.</p>

The following table describes the system-defined roles along with the default capabilities granted to each role, relevant to the Catalog Deployer module.

**Table 4: Predefined Roles and Capabilities**

Predefined Roles	Capability					
	Manage Basic Service Deployments	Manage Advanced Service Deployments	Manage Custom Deployments	Import Deployments	Deploy Deployment Packages	Package Branded Content Libraries
Catalog Designer and Administrator	✓	✓	✓			
Organization Designer			✓			

Predefined Roles	Capability					
Site Administrator	✓	✓	✓	✓	✓	✓
Catalog Publisher	✓	✓	✓	✓	✓	
Licensed Content Publisher	✓	✓	✓	✓	✓	✓

## Service Catalog Implementation and Configuration Management

This section describes how to design a Service Catalog implementation and the associated processes that support industry-standard configuration (or change) management practices.

Typical methodologies are reviewed and compared with those available for Service Catalog applications, including Catalog Deployer configuration management. The section discusses different approaches to configuration management, as they are matched to the stage of deployment—from initial development, through testing, deployment, and maintenance.

### Catalog Deployer Best Practices

Catalog Deployer implements best practices embodied by IT industry standard configuration management methodologies and technologies. Therefore, it is useful to review these practices.

- Changes to software configuration items (that is, the individual software modules or components that comprise the IT application) are made in a development environment.
- In the same (development) environment, the changed software typically undergoes preliminary testing (unit testing).
- Once the software has passed unit tests, a copy of the “source” for the software is checked into a source code control system and labeled as the release candidate. “Source” may consist of several types of artifacts, including code written and maintained in a text editor, or, increasingly common, specifications stored in XML files or within a metadata repository that is part of an integrated development environment.
- The saved source is deployed to a tightly controlled test environment, where it undergoes rigorous testing. The test environment may be reinitialized before each set of tests, to ensure that results in different runs are comparable.
- The testers are responsible for finding problems, not diagnosing or fixing them. All problems are reported to the development team, which uses its development environment to fix the problems, retest the code, and save a copy of the revised source.
- The fix, extract, deploy, and test steps are repeated until the testing team certifies that the software meets all test criteria—these may be performance measures or functional requirements or a combination.
- The same source that was deployed to the testing environment (and tested!) is deployed to the production environment.



## Configuration Management

For Service Catalog applications, the software that is developed is typically a service definition with related elements such as categories, groups, tasks, and checklists.

Catalog Deployer can be used by customers to deploy content such as service definitions, as well as any associated services.

The typical configuration management scenario, and the role that Catalog Deployer plays is similar to the following:

- A new or enhanced service definition is developed and unit tested in a development environment.
- Catalog Deployer is used to extract the new or updated service definition from the development environment. The resultant deployment package may be exported and placed under source code control if desired.
- Any other code resources (such as JavaScript libraries) that do not reside in the Service Catalog database may also be placed under source code control.
- Catalog Deployer is used to deploy the service definition in a test or quality assurance (QA) environment.
- The service is tested. If problems are encountered, the previous steps are repeated—code is fixed in development, extracted, and redeployed to the test environment—until the service is certified as having met the stated requirements.
- Once the service is accepted in the Test environment, it is deployed to the production environment, using the same procedure that originally deployed the code to the test environment.

This scenario adheres to industry standards in that the development environment is the only place changes are made to the service definition, and an automated process is used to deploy a controlled set of source code to test and production environments.

However, this scenario is incomplete. In most implementations, people and business units (a type of organization) are dynamically added to the Production environment, as people log in to Service Catalog for the first time, have a service ordered on their behalf, or are assigned to perform a review or authorization. Therefore, Catalog Deployer must also be used to migrate these entities from the production site back to development, so they are available for use in service definitions and other Service Catalog configuration items, such as authorizations, that are still under development. Further, the service definition may refer to related entities such as email templates, groups, queues, service team-organizations, and roles. If any of these do not exist in the target environment, they must be deployed to that environment before the service's deployment package can successfully be deployed.

## Configuring Catalog Deployer

Configuring Catalog Deployer involves:

- 1 Meeting the prerequisites for installing Service Catalog and configuring client workstations to support Catalog Deployer.
- 2 Installing a version of Service Catalog that includes Catalog Deployer. Catalog Deployer is automatically installed as part of all Service Catalog sites.




---

**Note** All sites including service packs should have the same version of Service Catalog.

---

- 3 Configuring implementations and sites within the development and production Service Catalog instances.
- 4 Configuring application server JDBC data sources on source and target sites.
- 5 Using the Administration and Organization Designer modules to ensure that personnel have access to Catalog Deployer capabilities appropriate to their functions in the implementation.




---

**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.

---

## Configuring Client Workstations

The deployment package produced by Catalog Deployer contains a compressed XML representation of the definitions of the included entities. Catalog Deployer uses file-based transmission of packages, in which package contents are transferred from one site to another via the export of the package and its subsequent import into the target site.

To support this file-based transmission, the user's browser must be configured to allow encrypted pages to be saved to disk. To do so:

- 
- Step 1** Choose **Tools > Internet Options > Advanced**.
- Step 2** Click **Do Not Save Encrypted Package to Disk** check box.
- Step 3** Click **Ok**.
- 




---

**Note** This configuration is not required if all deployments are via direct site-to-site transmission.

---

## Configuring Implementations and Sites

Before using Catalog Deployer, you need to use the development and production instances to configure your implementations. An implementation is the collection of sites among which Catalog Deployer migrates Service Catalog service definitions and other Service Catalog entities.

Configuring implementations and sites consists of the steps summarized below, also described in detail in the following section:

- 
- Step 1** Assign a name to the implementation. This name typically reflects the company or the project. This name is for documentation only, and is not used by Catalog Deployer.
- Step 2** Name the Sites within the Implementation.  
Within an implementation, each site must have a unique name. Service Catalog uses this name to identify the site for purposes of assigning protection levels for the Service Designer and Organization Designer pages that allow users to change (add, modify, and delete) entities.  
Two sites are typically required: Development (DEV) and Production (PROD). Additional sites, for example, STAGE, TEST, or QA, may be used if configuration management and migration plans call for their use.
- Step 3** Specify the Home Site for Each Logical Entity:  
Specify the (single) Home site for each logical entity. A few rules are useful in making a decision about the Home sites for logical entities:
- A logical entity should typically be Home at one and only one site. This will allow you to better manage changes to the entity instances.
  - Any logical entity involved in the definition of a service should clearly be home in the development instance.
  - Any logical entity created through production use of the system should be Home at the Production site. If Single Sign-On (SSO) or Directory Integration which includes an Import Person event is enabled, logical entities home in Production will include people (and thus queues, as they are in the same table). If the automated user creation facility creates Organizational Units, then Organizational Units should also be Home at the production site.
- Note** Logical entities are almost never Home at a test or stage site. This is because such sites are typically rebuilt with production data and newly developed code to be tested.
- Step 4** Create a Data Source for Each Site  
For Catalog Deployer to deploy content to a site, a JDBC data source must be configured for that site in the application server running the Service Catalog application. For example, in order to deploy services developed on the development site to test and production, the development site must include data sources for both the production and test sites; to deploy organizational entities from the production site to development, the production instance must include a data source for the development instance.  
For clustered sites, the data source must be accessible to each node that comprises the site.
- Step 5** Repeat the Process for Each Site  
The data that specifies implementations, sites, and logical entities is, in turn, stored within logical entities. These specifications must exist in all Service Catalog instances which comprise the implementation.  
You also need to create appropriate data sources at all sites.
- 

### Configuring Implementations in the Administration Module

You must configure your implementation environment settings in the Administration module to begin using Catalog Deployer. The values you set in Administration allow source sites to recognize target sites.

These steps must be performed after you have defined your data sources in the JDBC data source page on the application server console.

To configure an implementation:

- 
- Step 1** Log in to the development instance as a user with Administration privileges.
- Step 2** From the module drop-down menu, choose **Administration**.
- Step 3** Click the **Settings** tab.
- Step 4** From the menu on the right-hand side, choose **Entity Homes**.  
The Logical Entity Home Specification page appears.
- These settings influence the behavior of Catalog Deployer as well as the Service Designer and Organization Designer modules. If they are set improperly, incorrect data could be written to Service Catalog application sites, including production. Only systems administrators should change these settings.
- Step 5** In the **Implementation Sites** section, in the text field at the bottom of the page, enter a site **name**; for example, "Development".
- Step 6** From the **Select a Data Source** drop-down menu, choose a data source and click **Add New**.  
The site name is added to the list of site names.
- Step 7** Add the name for your production site as another site.
- Step 8** If the implementation includes other sites which need to be refreshed via Catalog Deployer, such as QA or Test, add these as well.
- Step 9** At the top of the page, in the **Implementation Name** field, enter an implementation name; for example, **My Cloud** or **Data Center Management**.
- Step 10** From the **This Site is** drop-down menu, choose the development site to identify the current site and click **Update**.
- Step 11** Review the **Home Site** assignments for the entities, changing any that do not fit your requirements. Click **Update** when finished.
- Step 12** Assign the appropriate **Site Protection Level** to each site you have defined. Click **Update** when finished. These protection levels alter the behavior of the Service Designer, Organization Designer, and Administration pages through which the corresponding logical entity is maintained.

**Table 5: Protection levels**

Protection Level	Effect on User Interface at Nonhome Sites
None	Nothing: all UI elements for creating and modifying entities remain available.  A protection level of "none" should only be used on a development site, in the initial phases of an implementation. This allows all entities to be created, modified, or deleted within the site by users who have appropriate roles.
Create only	Controls for creating new logical entities are disabled.
Create, Modify	Controls for creating and updating logical entities are disabled.

Protection Level	Effect on User Interface at Nonhome Sites
Create, Modify, Delete	<p>All controls for creating, editing and deleting logical entities are disabled.</p> <p>A protection level of “Create, Modify, Delete” should typically be applied to any test, staging, or QA sites. These sites would typically be refreshed by copying a complete database from production (for example, to support performance or volume testing) or by deploying services from development for functional testing prior to promotion to production.</p> <p>A protection level of “Create, Modify, Delete” should typically be applied to the production site. A protection level of “Create only” would allow minor modifications to be applied to protected entities, such as changing an entity name.</p>

**Note** The same site names, entity home settings, and protection levels should be specified at all sites in an implementation. This will prevent users from inadvertently creating or modifying an entity at the wrong site, where it could be overwritten by the next deployment of the “same” entity, developed and maintained at a different site.

*Configuring Data Sources for Sites*

In order for Catalog Deployer to transmit a package to another site, the JDBC data source that corresponds to the target site must be configured in the same application server as the source site. The procedures for configuring the JDBC data source are the same as those for configuring the RequestCenter data source, as outlined in the [Cisco Prime Service Catalog Installation and Upgrade Guide](#).

The same JNDI name prefix should be used to allow Service Catalog to discover the data source. The list of discovered data sources appears in **Administration > Settings > Data Source Registry**.

Certain data sources are used for reporting purposes and are not meant to be seen by Service Designer and Catalog Deployer users. To limit the data sources to just the ones these users need, check the **Use for Entity Home Definition** check boxes for those data sources, and then click **Update**.

*Configuring Sites*

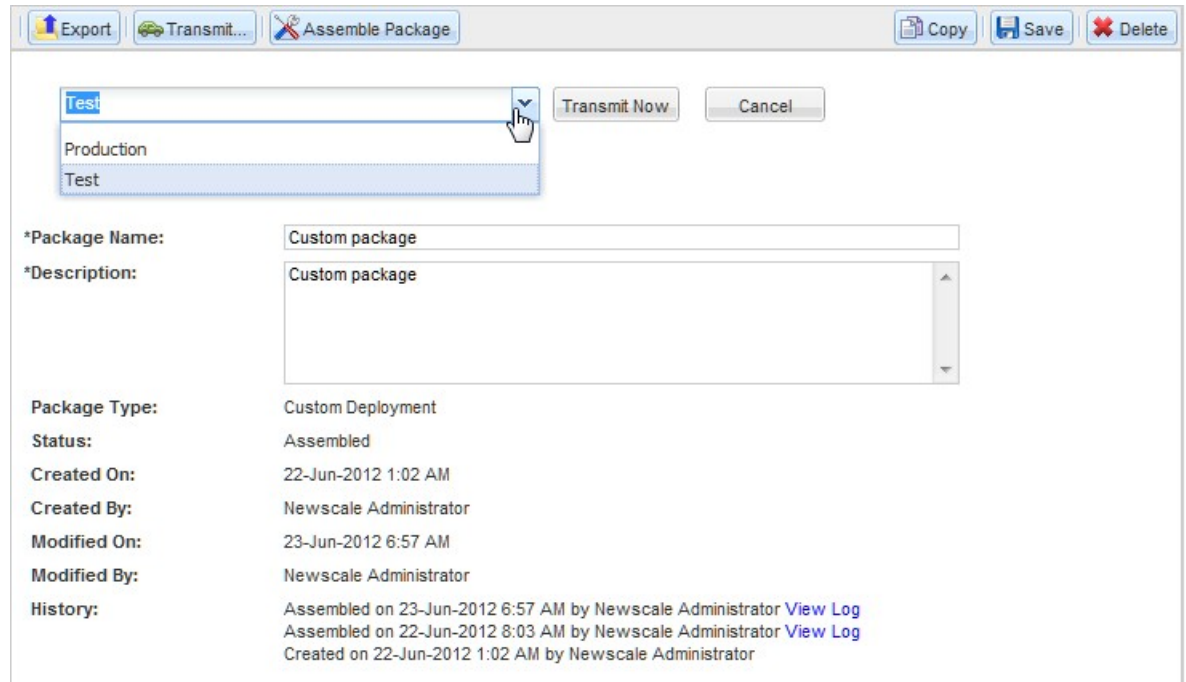
You would initially configure one site - you would typically start with the development site. The development site is the source site for service definitions that are migrated to other target sites that become visible to the source site. Any site can be configured as a source or a target site, or potentially both, depending on the configuration of your overall implementation.

For example, you might configure your development site to see your test and production sites, and configure the production site to see the development and test sites. In this scenario, you would deploy entities from development to test, and then from development to production, assuming that testing was successful. You would migrate organizational information from production to both test and development sites.

Repeat the configuration of potential target sites on all source sites, including the definition of required data sources.

To verify that configuration is successful, log in at each site, navigate to Catalog Deployer, and in the Action pane, transmit a package to one of the target sites you just configured by clicking **Transmit**. You should see your site in the drop-down menu as one of the target sites you can choose.

**Figure 1: Target sites**



362511

## Catalog Deployer Performance Considerations

Catalog Deployer must be run when the source or target site is online and in use. However, it may be advisable to apply some restrictions to this usage.

### *Concurrent Usage of Catalog Deployer*

Catalog Deployer consumes a significant amount of memory for assembling and deploying packages, as well as previewing services. To avoid any issues resulting from memory consumption, Catalog Deployer prevents more than five users from assembling, importing, or previewing a package at the same time. If a sixth user attempts to assemble, import, or preview a package, Catalog Deployer displays an alert to that effect, telling the user to try again later. While this may be an unlikely scenario in a Production environment, since the Production deployment will likely be handled by a single person, it may well occur in Development or Test environments if multiple service designers are packaging their respective content for deployment.

### *Browser Session Time-out*

Package assembly or deployment is still considered interaction with the browser. Hence the session time-out for inactivity configured in the Administration module does not cause large package assembly or deployment to time out.

### *Package Size*

Catalog Deployer only creates/updates associated entities a single time within a package if the entity is required for multiple primary entities. For example, if the "IT Software Configuration" dictionary is necessary for 20 services within a deployment package, Catalog Deployer only creates/updates the dictionary once on the target

site. In grouping services with shared components in the same package, the size of the package and the number of redundant creations/updates Catalog Deployer must perform can be reduced.

Users can expect the performance of a large package (as defined by its size) to be slower. Users should also be aware that assembly or deployment of such packages may occasionally fail. If this occurs, the solution is to simply break the package up into a smaller set of primary entities; for example, creating two packages of 10 services each instead of one consisting of 20 services.

### *Hot Deployment*

In principle, Catalog Deployer can deploy a complete package in one database transaction. In such a case, failure to deploy any one component (for example, a service definition in an advanced package cannot be deployed because a specified queue does not exist in the target) would result in the entire package contents being rolled back. The target site's repository would remain as it was before the deployment was started.

In practice, however, this all-or-nothing deployment may cause problems. Once an entity has been deployed, it is not available to online users until the complete package has been deployed and the database transaction committed. If, during this time frame, a user attempts to order a service that has been updated by the ongoing deployment, the user's session would hang, waiting for the service definition to be available. Eventually, the user would receive an error (worst case) or, after a delay, be able to order to service (best case). (This error may occur only when a user is initially ordering a service, not when task performers or authorizers are working with this service.)

One sure way to avoid this scenario is to not allow deployments while a production system is in use. This complies with the industry-accepted best practice of performing updates to a production system during a regularly scheduled maintenance window. However, waiting to deploy until a maintenance window roles around may not be possible. To minimize the probability of problems arising if a deployment must be run when Service Catalog is operational, it is advisable to group services into smaller logical packages. The service designer can still take advantage of reduced package size for shared components by grouping related services into the same package.

## Catalog Deployer Packages

This section provides a general overview of Catalog Deployer functionality. The deployment package types—Basic Services, Advanced Services, and Custom—differ in the primary entities each can deploy and the options available for governing deployment of associated entities.

### Basic Services Deployment Packages

Basic Services deployment packages work in a similar way as importing service definitions in Service Designer. If any associated entities (for example, a queue or organizational unit) are not found on the target site, the deployment would simply skip the entities not found. Basic packages cannot include bundled services.

Service deployment package also captures Standards and Service Items. The ones that are referenced in lookup Dynamic Data Retrieval (DDR) and Service Item-based dictionaries are automatically included in the package. There is an Administration global setting that controls whether the standard entries are to be deployed as well (they are always captured; the control takes effect on deployment target only).

You can optionally control rendering of the blank/empty values fetched by DDRs as such on the service form, by setting the `serviceform.ddd.null.value.as.empty` property in the `newscale.properties` file to true.

As a default behavior, the basic services checks for the categories on the target site. If the source site categories exist on the target site, the services will be added to the associated category on the target site. If these categories

does not exist on the target site, a new category will be created on the target site and will be merged with the source site entities.



**Note** You do not need permissions in Service Designer to view and choose services for deployment in Catalog Deployer.

The content of a Basic Services deployment package and its associated entities are summarized below.

**Table 6: Basic Service deployment package**

Content Type	Action	Entity Types/Description
Primary Entity	Chosen via the Services content tab	Service Definition—All aspects of the service definition as defined in the Service Catalog option of Service Designer
Component Entities	Automatically deployed	All entities referenced by the Service Definition: <ul style="list-style-type: none"> <li>• Categories</li> <li>• Data Dictionaries</li> <li>• Dictionary Groups</li> <li>• Active Form Components</li> <li>• Active Form Component Groups</li> <li>• Keywords</li> <li>• Objectives</li> <li>• Presentation Elements</li> <li>• Script Functions</li> <li>• Script Libraries</li> <li>• Service Groups</li> <li>• Service Items and Service Item Groups</li> <li>• Standards and Standard Groups</li> <li>• Extensions</li> </ul>



Content Type	Action	Entity Types/Description
Associated Entities	Deployment will skip the entity association if any is not present in the target site	All entities referenced by the Service Definition: <ul style="list-style-type: none"> <li>• Agents</li> <li>• Email Templates</li> <li>• Functional Positions</li> <li>• Groups</li> <li>• Organizational Units</li> <li>• People</li> <li>• Roles</li> <li>• Queues</li> </ul>

By default, Catalog Deployer deploys Standards data as well as the Standard definition. You can override this behavior by disabling the Administration setting to **Deploy Entries (data) in Standards Tables**. You may wish to adopt this, for example, if a different set of data should be used in different environments or data is being provided via a file import from an external source.

## Advanced Services Deployment Packages

An Advanced Services deployment package deploys the specified service definitions and their components. This package type provides the ability to control options for how to process the associated entities of a service definition during the deployment on the target site.

The Advanced Services deployment has two major differences from a Basic deployment:

- All services comprising a bundle can automatically be deployed by choosing the parent service and indicating it is a bundle.
- The user can specify which action to take if associated entities do not exist on the target site at the time of the deployment. This overrides the behavior of a Basic Services deployment, which automatically fails if any associated entity does not exist.

The options which govern behavior of Catalog Deployer in an Advanced Services deployment package with the associated entity or the associated category are summarized below.

**Table 7: Advanced Services Deployment Package for Associated Entities**

Associated Entities Options		
Entities	Available Options	Result
Service Accessories and Prerequisites	Skip	
	Fail	

<b>Associated Entities Options</b>		
<b>Entities</b>	<b>Available Options</b>	<b>Result</b>
Agent	Skip	Reference to the agent is removed from the delivery or authorization task
	Fail	Deployment fails
Email Template	Skip	Reference to the template is removed from the delivery or authorization task
	Fail	Deployment fails
Functional Position	Skip	Association to the functional position is removed; assignment is to the Default Service queue
	Fail	Deployment fails
Group	Skip	Association to the group is removed; assignment is to the Default Service queue
	Fail	Deployment fails
Organizational Unit	Skip	Assign the task to the Default Service Queue
	Fail	Deployment fails
People	Skip	Assign the task to the Default Service Queue
	Fail	Deployment fails
Queues	Skip	Assign the task to the Default Service Queue
	Fail	Deployment fails
Roles	Skip	Assign the task to the Default Service Queue
	Fail	Deployment fails

**Table 8: Advanced Services deployment package for Associated Categories**

Associated Category Options	Result
Update Category	This option will add or skip the associations for a category on the target site only if the category is present on the target site. If the category does not exist on the target site, the associations will be skipped.
Include Categories	<p>Using this option to include the source categories on the target site. This option will create the categories on the target system, if the categories are not present already.</p> <ul style="list-style-type: none"> <li>• <b>Merge:</b> Merge will merge the category and association on the target system with the category and association from the source system. For example, consider the Service (S1) with Category C1 in the source system, and same service S1 with another category (C2) in the target system. After merging, the target system will contain service S1 with both categories (C1 and C2). If the associated entities for a category does not exist on the target site, do one of the following: <ul style="list-style-type: none"> <li>◦ Fail: Stop the deployment.</li> <li>◦ Skip: Skip the associated entities from the target system.</li> </ul> </li> <li>• <b>Replace:</b> If a category already exists on a target system, the existing association of that category would be replaced by the associated entities from the source system. <ul style="list-style-type: none"> <li>◦ Fail: Stop the deployment.</li> <li>◦ Skip: Skip the associated entities from the target system.</li> </ul> </li> <li>• <b>Skip:</b> Retain the associated entities as is from the target system.</li> </ul>

Since the Skip and Create (available only for selected entity types) options substantially change the definition of the service, they should be considered useful for “quick-and-dirty” deployments only. The service's form component would be transferred intact; however, potentially significant changes could be made to the delivery plan.

## Custom Deployment Packages

Custom deployment packages allow you to choose entities individually and control options for how to process associated entities during deployment. Custom deployment is typically used for organizational entities, Functional Positions, Email Templates, such as People, Queues, Roles, Organizational Units, Agents, Services Items, Standards, Billing Rates, Account Definition, Agreement Templates, Categories, and groups.

In the customer deployment packages, you can choose to replace only certain attributes for a service on the target site rather than replacing the complete service definition. Under the **Service Deployment** options, select the options that can be updated on the target site.

There are more granular options controlling Service Items and Standards deployment behavior. The Service Item instances and Standards entries can be optionally included as a part of the deployment. But the option applies to all Service Item types and Standard in the package. To include entries for some Standards or Service Items but not the others, you will have to break them into separate packages.

During import, if the Unit rate flag is set (Source XML) in **Demand Management > Billing Rate > Billing Rate Definition**, the 'Merge' option is selected for Billing rates when creating the custom package in Catalog Deployer. If the rate table already exists in target, the 'Merge' option will be ignored and behaves as an 'Overwrite' by deleting the existing data records in the target.

The following validations should be in place for a Rate table which has 'Unit Rate' flag set. This should also be implemented for Demand Mgmt, NSAPI and REX flows.

- Only one attribute should be Billable.
- Only one data record should exist in the Rate table Data.
- The single billable attribute must be of the data type 'Number' only (Integer/Long/Double/Money).
- If there are multiple records in the data tab, do not allow the user to check 'Unit Rate' flag in the first tab.

Deploying categories via a custom deployment allows you to recreate all or part of the category structure of the source site in the target site. Chosen categories and their subcategories are deployed, and the relationship of the categories to services with which they are associated is re-established to match that in the source site. This supplements the capabilities provided by Basic deployment packages, where only those categories associated with the services being deployed are included in the package.

While deploying a category on the target side, you can choose to :

- Not include the source side association and retain only the target site entities.
- Or include the source side association and merge/replace/skip the target associations in case the category already exists on the target site.

A package may contain more than one type of entity where there are interdependencies between the entities. For example, a person must be associated with an existing organization. Catalog Deployer automatically deploys entities in the correct order, so these interdependencies may be maintained.

Further, there may be interdependencies among entities of the same type: a person may designate another person as a supervisor, or an organizational hierarchy may exist. In these cases, Catalog Deployer also deploys entities in the correct order (the supervisor entity first, for example) so the relationships may be re-established.

For each of the entities (except functional positions) to be deployed, you have an option to overwrite the entity definition at the target site with the new definition contained in the package, or to skip the deployment.

Skipping entity deployment is risky since Catalog Deployer only checks for the existence of the entity in the target site, and does not inspect its content (for example, to see if the source is newer than the target). The Skip option can optimize performance by not reinstalling entities that already exist, but should be used in limited circumstances (for example, if you are deploying a large package for the first time into a target site where the entities are not known to exist). If any aspect of the deployment fails (for example, you attempt to deploy a person whose home OU is not in the target site and not in the current deployment package), you can fix the omission and redeploy the package. Only package contents that still do not exist in the target site would be deployed.

For each entity type (except email templates, functional positions, and agents), you also have the option to specify the behavior of Catalog Deployer with regards to re-establishing relationships to other related entities in the target site.

- Use “Do not include” to deploy only changes to the entity definition, without attempting to duplicate the relationships in the source site. This would be required if, for example, you want to deploy a new organization only, and not the relationships of this organization to numerous people, some of whom might not exist in the target site.
- Including source site associations will typically be required. This will duplicate the relationships in the source site at the target site. Using the “Skip” option will allow the deployment to continue if some of the associated entities are not present in the target site. For example, you may have assigned a queue to a service that is still in development and has not yet been deployed. If you use the “Skip” option, be sure to read the logs carefully, to ensure that no expected associations have been skipped.

You can also merge data imported from source site to target site during deployment. This option is available for service item definitions, standard definitions, billing rate definitions, and agreement templates.

**Note**

---

Portal page and portlet permissions can be deployed with roles. The deployment must be performed after the portal pages and portlets have been imported through Portal Designer.

---

## Creating and Deploying a Deployment Package

A deployment package follows this flow in Catalog Deployer:

- 
- Step 1** Create a Deployment Package. Persons granted permission to access and use Catalog Deployer can create a deployment package and choose the entities to be included in the package. See the [Creating a Deployment Package, on page 22](#).
  - Step 2** Assemble the Package. When the underlying content is considered ready for deployment, a deployment package is assembled by clicking **Assemble Package** in the Action pane. At this time, Catalog Deployer extracts the content and creates a copy saved as part of the deployment package. Once package assembly has occurred, any changes to the entities included in the package are not captured unless the package is reassembled. See the [Assembling a Deployment Package, on page 23](#).
  - Step 3** Transmit the Package. The assembled package is sent to a target site for deployment. Packages may be transmitted via Catalog Deployer, or, if there is no direct connectivity between source and target sites, you may perform this step “off-line” using the export and import process. See the [Transmitting a Deployment Package, on page 23](#).
  - Step 4** Deploy the Package. A person with permission to deploy the package selects the received package on the target site and runs the deployment. See the [Deploying a Package, on page 25](#).
  - Step 5** View Record Log files record all activity that occurs within Catalog Deployer on each site. See the [Log Files, on page 28](#).
  - Step 6** For entities that are not supported during deployment, see [Unsupported Entities, on page 5](#)
-

**Note**

It is only necessary to save packages when the package definition is modified. Package content is automatically saved.

## Creating a Deployment Package

To create a deployment package:

**Step 1** Choose **Action > New Deployment Package** in the View and Search pane.

**Step 2** Enter the following details in the New Deployment Package window:

- Package Name
- Description
- Package Type

**Step 3** Click **Save**.

**Note**

- Standards for naming packages should be developed, to allow users to infer the package contents from its name. For example, the name could consist of a designation of the type of package, the source site, a description of the content, and the build number or date the package was created. You cannot create two packages with the same name in the source or target site.
- The application will not allow the user to enter Package Names that use special characters (for example, no “\/:\*?<>()[]” and so on). Spaces are allowed, but they are replaced with an underscore ( ) in the Log XML Output file.
- Package description is required. Both the package name and description can be modified after the package has been created.
- If needed, click the package name in the View and Search or Content pane to see the Package Name and Description fields. The search results only return packages where the entity type or search value entered were primary entities within the package. Primary entities are those entities (service definitions, organizational units, groups, queues, people, functional positions, roles, categories or email templates) that were chosen for inclusion in the package
- The Package Type cannot be changed once the package has been created. See the [Log Files](#), on [page 28](#) for more information on Package Types.

## Adding Content to a Deployment Package

When a package is created, it is assigned a status of “Not Transmitted”. When it is clicked in the View and Search pane, its contents appears in the Content pane.

To add content to a package:

- 
- Step 1** In the Content pane, click the **Add** drop-down menu and choose the type of entity you wish to add from the list of available entities for that type of package.  
A Search dialog box appears where you can search for and choose content.
- Step 2** Choose the content you want by checking its check box.
- Step 3** Click **Add** to add the chosen content.  
The content appears in the Content pane and is automatically saved. If you need to remove an entity, check its check box, click **Remove** and then **Yes**. Content may be added and removed at any time until the package has been transmitted. If content is changed after the package has been assembled, the package must be reassembled.
- 

## Previewing Package Contents

Basic and Advanced Services Deployment Packages allow you to preview the definition of a service.

For a service that is already been included in a package:

go to the Content pane of the Package, choose the service to be previewed by checking its check box, and click **Preview**.

A service preview consists of a summary of the service definition, as well as a rendering of the service form. All entity references are summarized in the “Additional Content” section at the end of the preview. This may help you ensure that these entities are present in the target environment before you deploy the service.

## Assembling a Deployment Package

In the Action pane, click **Assemble Package** to assemble a package. Click **OK** to confirm that package assembly was successful. Once a package has been assembled, its status remains “Not Transmitted”. It can be reassembled if the definition of any of its components changes, or if you want to add or delete entities to be deployed. The log entry for package assembly includes both the primary entities specified and any component entities that will also be deployed. In the History section of the Action pane, the log may be viewed by clicking the **View Log** link.

## Transmitting a Deployment Package

Content can be transmitted directly to another site if:

- The target site has been defined in **Administration > Settings > Entity Homes**.
- A datasource corresponding to the target site has been defined in the JDBC data source page on the application server console.

To transmit a package:

- 
- Step 1** In the View and Search pane, use the View drop-down menu to view packages with the status of: **Not Transmitted**.
- Step 2** Locate the package within the list. Click the package name to view its information in the other panes.
- Step 3** If the package has not been assembled, in the Action pane, click **Assemble Package**. Click **OK** to confirm that package assembly was successful.
- Step 4** In the Action pane, click **Transmit**.
- Step 5** Choose the target site from the drop-down menu.
- Note** You can choose only those data sources for which your site administrators have configured. See the [Cisco Prime Service Catalog Designer Guide](#) for instructions. In situations where no sites are listed, the site may be configured to only use the export/import functionality of Catalog Deployer.
- Step 6** Click **Transmit Now**.
- Step 7** Click **OK** to confirm that the transmission was successful.  
The deployment package is transmitted to the target site. The status of the current package in the source site is changed to “Transmitted”. A transmitted package may be transmitted to additional sites or exported.
- 

## Exporting a Deployment Package

A package can be exported, instead of or in addition to being transmitted. Exporting a package produces an XML file consisting of the content of the assembled package. The export file can then be imported into a target site and deployed.

Exporting a package provides a textual representation of the package. It can be used as offline archival or checked into a corporate source code control system.

To export a package:

- 
- Step 1** In the View and Search pane, use the View drop-down menu to view packages with the status of: **Not Transmitted** or **Transmitted**.
- Note** Packages which have been received for deployment, or deployed, cannot be exported.
- Step 2** Locate the package within the list. Click the package name to view its information in the other panes.
- Step 3** If the package has not been assembled, in the Action pane, click **Assemble Package**. Click **OK** to confirm that package assembly was successful.
- Step 4** In the Action pane, click **Export**.  
An export dialog box appears, as shown in the example below.
- Step 5** In the dialog box, click the file name link (in the example above, the file name link is **Basic Service Package02**).  
A File Download dialog box appears.
- Step 6** Click **Save**.  
A Save As dialog box appears.
- Step 7** Rename the file if desired and choose your desired destination.



The destination would typically be on a shared drive, accessible to all users with Catalog Deployer capabilities. Standards for naming directories and structuring subdirectories should be established to facilitate tracking packages. For example, a new subdirectory could be created for all packages deployed as part of the same change request.

**Step 8** Click **Save**.

**Step 9** Click **Close** to close the export dialog box.

---

Export/import can be used instead of transmitting a package in cases where security or other concerns do not allow directly transmitting a package from the source to a target site. The results are identical—the package is created in the target site in the “Received for Deployment” status, and can then be deployed.

## Importing a Deployment Package

An exported package needs to be imported into the target system.

To import a package:

---

**Step 1** In the View and Search pane, choose **Action > Import**.  
An import dialog box appears, asking you to browse for the file to be imported.

**Step 2** Click **Browse**.

**Step 3** Find and choose the package file.

**Step 4** Click **Open**.

**Step 5** Click **Import**.

The deployment package is imported, assigned a status of **Received for Deployment**, and opened. It can now be deployed.

**Step 6** Close the import dialog box.

---

**Note**

To import a deployment package back into the source site from which it was originally exported, you must first delete the identical package from the source site. The application will recognize duplicate files even if the filename has been changed.

---

## Deploying a Package

The deployment status is shown while deploying packages. The log shows the name of the package along with the details of entities discovered, total count, processed count, and failed count. The information on entities covered during processing or failure can be inferred from the log. In case of failure of a package deployment, all processed entities from that package will be rolled back.

To deploy a package that has been transmitted to or imported into the target site:

- 
- Step 1** In the View and Search pane, use the View drop-down menu to view packages with the status of: **Received for Deployment**
- Step 2** Locate the package within the list. Click the package name to view its information in the other panes
- Note** The deployment runs according to the deployment options and associated entity rules that were chosen on the source site. Users at the target site can click through the tabs and view the entities chosen, but cannot modify the package in any way.
- Step 3** In the Action pane, click **Deploy**.
- Step 4** Click **OK** to confirm that deployment was successful.
- 

After the deployment completes successfully, the status of the package changes to “Deployed”.

The assembled content of the package to be deployed must match the release level of the target site. Catalog Deployer tracks the application version under which the package was assembled, and will not allow deployment across different versions.

A Basic or Advanced Services package includes all the component design elements used by the services deployed. However, design components which are not stored in the database are not included in the deployment package and must be deployed separately. These elements include:

- JavaScript libraries referenced by any ISF Scripts
- Data sources added to the environment and referenced by data retrieval rules or option lists

In principle, deployment does not affect the service and component definitions used by requests that were in-flight when the deployment occurred. However, because of the dynamic nature of the requisition process, previously submitted requests are affected by:

- Changes to the content of rules or JavaScript functions and libraries
- Changes to the delivery plan of service requests that have not yet passed their final approval step

The deployment schedule and service designers need to take into account that in-flight requests based on the previous version of the service definition are affected by changes to the above elements.

## Transmit and Deploy Multiple Packages

You may transmit and deploy multiple packages in one function. Catalog Deployer processes each package in turn, and provides the status for each. This procedure makes it much easier to deploy a large number of packages with limited user intervention.

To transmit multiple packages:

- 
- Step 1** In the View and Search pane, choose **Action > Transmit Multiple Packages**.
  - Step 2** Click **Add Packages to Transmit** to open a Search dialog box where you can search for and choose packages to be transmitted. The Search dialog box only allows you to choose packages with a status of “Not Transmitted” (which have been assembled) and “Transmitted”.
  - Step 3** Choose the packages you want by checking their check boxes.
  - Step 4** Click **Add** to add the chosen packages.  
The packages appear below the Packages folder in alphabetical order. All chosen packages are transmitted. If you need to remove a package, check its check box, click **Remove** and then **Yes**.
  - Step 5** Under the “Select a target site” folder, choose the target site by checking its check box.
  - Step 6** Click **Transmit** to start the transmission.  
The packages are transmitted in the order in which they are listed (alphabetically). After the transmission process is complete, a status message appears, showing the transmission success or failure for each package. (The most common reason for a transmission failure is a version mismatch between the source and target sites). The log file for each package is also updated.
- 

### Similar Interface to Deploy Multiple Packages

A similar interface is available to deploy multiple packages:

- 
- Step 1** In the View and Search pane, choose **Action > Deploy Multiple Packages**.
  - Step 2** Click **Add Packages to Deploy** in Deploy Multiple Packages window to open a Search dialog box where you can search for and choose packages to be deployed. Any package with a status of “Received for Deployment” or “Deployed” may be chosen.
  - Step 3** Choose the packages to be deployed by checking their check boxes.
  - Step 4** Click **Add** to add the chosen packages.  
The packages appear below the Packages folder in alphabetical order. All chosen packages are deployed. If you need to remove a package, check its check box, click **Remove** and then **Yes**.
  - Step 5** Click **Deploy** to start the deployment.  
The packages are deployed in the order in which they are listed (alphabetically). Like the multiple transmission, multiple deployment indicates the success or failure for the deployment of each package; that information is available in the package's log file as well.
- 

### Closing and Reopening a Deployment Package

A deployment package in any status can be closed in the site in which it was created by clicking it in the View and Search pane, and in the Action pane, clicking **Mark as Closed**. Closing a package simply changes its status to “Closed”, so that it appears only in the view of “Closed” packages, rather than in other views. This

may make working with other packages easier, since there are fewer active packages to search through to find the one you want.

A closed package can be reopened at any time, for example, if you need to transmit it to another site or wish to export its contents. To reopen a package, click it in the View and Search pane, and in the Action pane, click **Reopen**.

## Deleting a Deployment Package

To delete a package, click it in the View and Search pane, and in the Action pane, click **Delete**. Click **Yes** to confirm the deletion. Deleting a deployment package permanently removes the package and its deployment history from the current site. Although package contents are compressed, the XML required to represent the package components may be quite large. Therefore, deleting packages will recover usable space in the repository/database. Package contents could be recovered if the package was previously exported; however, the complete deployment history of the package at the current site could not be recovered.

## Copying a Deployment Package

Unless you are very lucky, you will need to deploy the same entities multiple times as part of the build process. For example, you deploy one or more services from the development to the test environment; the testers find some defects that need to be repaired; the service definitions are repaired in development and redeployed, so that the fixes can be verified in test before being deployed to production.

The ability to Copy a deployment package facilitates this work flow. Once an initial package, with the desired content, is produced, you can copy the package.

To copy a package:

- 
- Step 1** Choose the package in the View and Search pane.
  - Step 2** In the Action pane, click **Copy**.  
A Copy Package dialog box appears.
  - Step 3** Rename the package.
  - Step 4** Click **Copy Package**.
  - Step 5** Click **OK**.  
The new package is created and opened with a status of **Not transmitted**—it contains the entities specified in the Content pane, not the assembled package content. You can then reassemble the package as required.
- 

## Log Files

The **View Log** link in the History section of the Action pane displays the actions by Catalog Deployer in detail. Clicking on the **View Log** link opens a View Log tab with Log Details and XML Output subtabs. Logs record all activity that a package undergoes. Assembly logs list all entities included in the package. Deployment logs list all entities successfully extracted on the target site. If the package failed to deploy, the error also appears.

Logs include all package details (name, description), time/date stamps for the time of assembly or deployment, and all included entities.

## Known Errors and Omissions

Catalog Deployer does not support renaming entities. Catalog Deployer will not behave correctly if you rename an entity at the source site and attempt to deploy it or deploy a service that references the renamed entity. In order to establish (or reestablish) an association with a related entity, Catalog Deployer attempts to find the entity in the target site by name. For deployment purposes, an entity “name” is the name attribute of all entities except for people (identified by their login name) and organizations (identified by the combination of organization type and name). The result is that:

- For component entities of a service definition (dictionaries, form components), a new entity is created, and an association with this entity established for the service being deployed.
- For primary entities being deployed, a new entity, with the new name, is created.
- For associated (directory entities and email templates) entities, the deployment may fail or the entity association may be skipped, depending on the package type and deployment options chosen.

The workaround is to:

- Turn off site protection for the affected entities in the target site, if applicable.
- Rename the entity in the target site so the name matches that in the source site.
- Turn site protection back on.
- Deploy the entity.

During the development process, you should carefully track entity name changes, so they can be applied, as explained above, in the target sites before attempting to deploy the entity to those sites.

The unit of measures referenced in Objectives and Billing Rate Tables are not captured in the deployment packages. You should create any new unit of measures established for your services or billing rates through the Administration module in the target site before deploying packages that reference them.

## Sample Deployment Scenarios

This section gives details on recommended procedures to use in some frequent development and deployment scenarios.

- [Initial Deployment, on page 30](#)
- [Placing Entities in Respective Servers, on page 30](#)
- [Deploying a Service to Use a New Queue, on page 32](#)
- [Deploying Services that use a New Email Template, on page 33](#)
- [Renaming a Queue and Service Team, on page 34](#)
- [Changing a Category and the Icon, on page 35](#)
- [Renaming Entities after a Service Catalog Upgrade, on page 35](#)
- [Adding a Custom Functional Position, on page 35](#)

- [Deploying to an Environment with Browser Cache Enabled](#), on page 36

## Initial Deployment

You need to create a new Service Catalog site. One option is to copy an existing Service Catalog database to the site, and adjust the installation and configuration to use that database. (The procedure for doing this is documented in this guide [http://www.cisco.com/en/US/products/ps13206/prod\\_technical\\_reference\\_list.html](http://www.cisco.com/en/US/products/ps13206/prod_technical_reference_list.html).) However, another option is to perform initial configuration of the database and then use Catalog Deployer to deploy all entities that should populate the new site.

### Performing Preliminary Configuration

Catalog Deployer does not deploy aspects of Service Catalog configuration that typically do not change as the Service Catalog evolves. Therefore, you must define these elements as you did for the initial site.

- Re-enter Administration modules settings for settings, directory integration, authorizations, and entity homes. These settings should be identical to those in Development except for environment-specific configuration items. (For example, you may have separate LDAP directories for Development and Test.)
- Ensure that any Service Link custom adapters were included in the installation procedure, and that any changes you made to Service Catalog adapters are reapplied.
- Re-enter agents and transformations (a known error and omission).

### Deploy Service Foundation Entities

Entities maintained via Organization Designer (people, organizations, groups, roles, and functional positions) must be present in a site before a service definition can refer to them. Therefore, all such entities must be deployed before the service catalog can be deployed. Such entities should be included in one or more Custom deployment packages.

In addition, any custom email templates must be deployed. Email templates can be deployed at any time before the service that uses them is deployed.

### Deploy Services

Once the foundation entities have been deployed, services may be deployed. For the initial deployment, keep the package size manageable (say, group things by service group) and be sure to review the log carefully. A basic deployment should be used, since all associated entities should have been deployed earlier.

## Placing Entities in Respective Servers

You should now have two (or more) sites in operation. All the services in Production should be identical to services in Development. (There may be additional services in Development that were part of the initial deployment, but that's okay.) In which environment will you be maintaining the entity definitions and membership (for groups, roles, and organizations)? In all cases, it is assumed that Entity Protection Levels are set to prevent any maintenance to entities in a non-home environment.

### Directory-Related Entities Reside Only in Production

The most straightforward approach is to home all entities that comprise a service definition in the Development instance and to home all entities related to people and organizations in the Production instance.

This makes using Catalog Deployer and maintaining the entities it deploys very easy—all aspects of all entities are always maintained in one and only one site. However, it complicates the work flow for creating and testing directory-related entities.

- 1 Before developing a new service, take an inventory of the queues, organizations, groups, roles, and functional units that are referenced in the delivery plan or other areas of the service.
- 2 If all of these associated entities already exist in the Development instance, it means they already also exist in Production (since they are homed in Production). Therefore, you can proceed with creating and testing the service.
- 3 If any of these directory-related entities do not already exist, log in to the Production instance and create them. Configure the entity definition, as well as its membership and roles.
- 4 Logged in to Production, create a Custom Deployment package containing the new entities and its associated entities. The package will need to use the option to “Use Source associations” for the new entities, since there are not yet associations in the target instance.
- 5 Deploy the custom package into Development.
- 6 Create the service using the new entities just deployed.
- 7 When the service is ready to be tested, create a Basic Services package containing the service.
- 8 Deploy (in this order) the custom package containing the directory entities to the Test site. Then deploy the Basic Services package.
- 9 Once the service passes testing, deploy the same Basic Services package from Development to Production.

In this scenario, all work regarding the directory-related entities (except actually referring to them in a service definition) is done in one place, the Production environment. This is good, because all work is done in one place and is easy to review and monitor. However, this process does have the following potential disadvantages:

- You are probably adding steps to the development process: if you did not get the entity definitions correct the first (or second or third) time, you will have to go back to Production, fix the entities, repackage, redeploy and retest. When you repackage, you will need to use the option to “Use Target Associations” for the primary entity, since it has already been associated with a service that only exists on the target site.
- Another possible objection might be that you doing extensive development work in a Production environment, which some organizations see as a security risk.

### Directory-Related Entities Reside in Both Production and Development

An alternate approach is to home all entities that comprise a service definition in the Development instance (this should never change) and to home entities related to people and organizations in both the Production and Development instances.

Homing entities such as organizations, groups, roles, and people in both instances has the following advantages:

- You can work on the definition of these entities in the logical place, the Development environment. It may take a few passes, for example, to get a custom role definition just right. You would not need to iteratively develop, deploy, and test—you could just develop and test, deploying when you are done.

- You can assign members to organizations or roles in the logical place, the Production environment, where all person information is automatically refreshed via directory integration and you are guaranteed that all Service Catalog users are represented.

Homing entities such as organizations, groups, roles, and people in both instances has the following disadvantages:

- Role-based access control currently allows access to a particular entity type. No further granularity is possible. For example, it is not possible to allow designers to define groups (and their roles) only in the Development environment, but only allow them to assign group membership in the Production environment.
- Since maintenance is happening in more than one environment, care must be taken when deploying packages. You must ensure that appropriate source and target associations to the entity are maintained.

As an example, assume you are developing a group whose members will consist of people across various service team organizations. The following section describes how to assign members to the group.

### People and Groups Reside in both Production and Development

- Create the group in Development, and assign some (test) members in order to verify that the capabilities and permissions granted via the group are correct. Not all people who must be in the group are in the Development environment, so the member list is incomplete and potentially incompatible with people in production (if you have some “test” people who do not have corresponding entries in Production).
- Use a Custom Deployment Package to deploy the group to Production, using source associations, but skipping an association if the referenced entity does not exist.
- Using Organization Designer in Production, associate the appropriate people with the group. You may do this either via the People or Group pages, whichever is more convenient, since both entities are home in Production and the entities could be maintained even with a “Create, Delete, Modify” protection level.
- If membership in the group changes, you only need to go to Organization Designer in the Production environment and make the appropriate changes. Further, directory integration has the ability to dynamically specify a list of groups in which a person is a member; if this capability is enabled (via updates to the enterprise directory and a corresponding directory mapping) no manual updates would be necessary.

### Deploying a Service to Use a New Queue

You need to develop a new service or enhance an existing service to use a new queue. Following best practices, the queue should have a corresponding service team organization in which it is homed.

There are two possible scenarios here, depending on where you have “homed” queues and organizations. Each has pluses and minuses. (This is just a special, and very frequent case of the discussion in the previous section.)

### Organization Units and Queues Residing in Production

This is a “clean” approach that places all work on organizations, people, and queues in Production. It was the only workable approach in versions on Service Catalog prior to 2007, so people upgrading from those versions may prefer to keep using it.



- A disadvantage to this approach is that you are doing manual work in Production (creating queues and OUs), which some organizations may frown upon for security reasons.
- The advantage of this approach is that all work on the queues and OUs—not only defining them, but assigning their members—is done in one place.
- Create the OU and queue in Production and assign appropriate roles.
- In Production, assign members to the OU.
- Deploy the new OU and queue from Production to Development. Include in the deployment package all service performers in the OU. You may wish to set the People deployed to skip existing people, to optimize deployment performance.
- Create the service in Development.
- Deploy the service from Development to Production.

### Organization Units and Queues Residing in Both Development and Production

In this scenarios consider that the definition of the entity is home in Development, but its membership is home in Production. (Of course, this division cannot be enforced by entity homes, so both sites are designated as home for the entities to allow maintenance via Organization Designer).

- 1 Create the OU and queue in Development and assign appropriate roles. Assign enough members so you can thorough test the new configuration.
- 2 Create the service in Development.
- 3 Deploy the new OU and queue from Development to Production, using source associations, but skipping any that do not exist at the target site (to exclude “Test” people).
- 4 Use Organization Designer (or rely on Directory Integration) in the Production site to assign members to the OU.
- 5 Deploy the service from Development to Production.

### Maintaining Organization Units and Queues after Initial Deployment

In both scenarios above, you are faced with possible changes to the initial queue and OU configuration after initial deployment.

- Changes in membership in the OU are handled as for initial deployment, that is, the OU membership is maintained in Production. It is not critical that all such changes be deployed back to Development, but if this is desired, a Custom Deployment package of the OU and new or affected People can be produced and deployed to Development, using source associations.
- Changes in the permissions assigned to the OU are applied in the site where the entity definition is home. The new OU definition is then deployed to the other site, using target associations.

### Deploying Services that use a New Email Template

Two packages are needed:

- 1 In Development, create a Custom package containing the email templates. Use the default deployment option—replace the existing entity.
- 2 In Development, create a Basic Services package containing the revised services.

- 3 Deploy the Custom package to Production.
- 4 Deploy the Basic Services package to Production.

Each package may be deployed as it is produced, provided the Custom package is deployed before the Basic Services package.

## Renaming a Queue and Service Team

After several services have been deployed, which include tasks assigned to a particular queue, you receive feedback that the queue name is misleading to service performers, who would like it changed. Or perhaps the company has been restructured and organizations (including Service Teams) need to be renamed to reflect the new organization.

This runs smack up against a “Known Error and Omission” documented in the [Known Errors and Omissions, on page 29](#). Since Catalog Deployer works by matching the names of entities across sites, it cannot match an entity that has been renamed. In some cases (as documented above), Catalog Deployer does fail and reports an error. In other cases, however, Catalog Deployer will create a new entity in the target. In the case of a renamed queue, any services previously deployed would still use the old queue. Only services deployed after the queue was renamed and deployed would reference the new queue. This is clearly not the intent of the designer.

The only way to prevent this behavior is to put in place processes that carefully monitor and control entities that need to be renamed. The processes would vary slightly, depending on whether queues and organizations are homed solely in Production, or in both Production and Development, but both involve manually renaming (via Organization Designer) the entities in both sites.

You must consider the following scenarios:

### Entities Reside in both Development and Production

- 1 In response to the requirement, the Organization Designer renames the queue and service team in the Development instance, through normal maintenance procedures.
- 2 An Administrator manually (via Organization Designer) renames the entities in the Production instance. Since the entities are homed in both instances, entity protection levels would normally allow this step.
- 3 Service Designers work on services using the renamed queues and service teams. Such services can be deployed through standard procedures.

### Entities Reside Only in Production

- 1 In response to the requirement, the Organization Designer renames the queue and service team in the Production instance, through normal maintenance procedures.
- 2 An Administrator relaxes entity home protection levels in the Development instance. This is necessary, since no manual changes to queue definition or membership is typically allowed, since the entity is homed in Production.
- 3 The Organization Designer renames the entities in the Development site.
- 4 The Administrator turns entity protection levels back on after the changes have been applied.
- 5 Service Designers work on services using the renamed queues and service teams. Such services can be deployed through standard procedures.

## Changing a Category and the Icon

Categories are deployed as component entities as part of a basic or advanced service deployment. This is an effective strategy when your main objective is to deploy a new or revised service definition and to ensure that the associated categorization is also deployed. However, using a services package is not effective when the category hierarchy or images associated with categories have changed.

A custom deployment package offers the option to deploy all or a portion of the category hierarchy, independent of any services that may reference that hierarchy. When the category structure is deployed to the target environment, Catalog Deployer automatically re-establishes the associations between each category and its services.

As always, the warning against simply renaming an entity, in this case, a category, applies. If a category no longer applies, you should delete its association to services and create a new, replacement category. The alternative is to rename the category in the source and all target instances:

- 1 The Catalog Designer renames the category in Development (where categories are home).
- 2 An Administrator turns off entity protection in the Production site.
- 3 The Catalog Designer renames the categories in the Production site.
- 4 The Administrator turns entity protection levels back on after the changes have been applied.
- 5 Service Designers work on services using the renamed categories. Such services can be deployed through standard procedures.

## Renaming Entities after a Service Catalog Upgrade

Service Designers can recognize entities with distinctive names as having been produced by the upgrade process:

- Every service definition has a corresponding active form component. The name of the form component is: "UPGD: Form", followed by the service name.
- All form components are assigned to a form component group. The name of the group is: "UPGD: Form", followed by the name of the service group where the service definition corresponding to the form resides.

Service Designers will want to, at a minimum, rename the artifacts produced by the upgrade process, so the names are more user friendly. (They may also want to refactor the structure of the form components, to promote reuse, but that is another issue.) This is a "standard" Rename scenario:

- 1 The Service Designer renames the form component and form component group in Development (where these entities are home).
- 2 An Administrator turns off entity protection in the Production site.
- 3 The Catalog Designer renames the form component and group in the Production site.
- 4 The Administrator turns entity protection levels back on after the changes have been applied.
- 5 Service Designers work on services using the renamed form components. Such services can be deployed through standard procedures.

## Adding a Custom Functional Position

You defined a Functional Position in Development (related to an OU) and specified the person holding that Functional Position for one or two organizations. You've changed the service in Development to route tasks to that Functional Position. How do you deploy the new and changed entities?

- 1 Create a Custom package in Development.
- 2 The Custom package contains the functional position you created and the people who are assigned to that position. The deployment option for the people is to use source associations.
- 3 Deploy the Custom package to Production.
- 4 Create a Basic Services package containing the revised service.
- 5 Deploy the Basic Services package to Production.

## Deploying to an Environment with Browser Cache Enabled

If the Browser Cache setting is enabled in the Administration Settings, changes made to icons and embedded images in the service catalog presentation will not take effect until the browser cache has been deleted. To prompt the application users to delete their browser cache, follow the instructions in this guide to increment the browser cache version.

## Importing/Exporting Teams

In a scenario where you wish to deploy a team along with its associated roles, services, and members, to another system. You must keep the below points in mind:

- Team Management should be activated to import teams.
- If the parent does not exist in the system, is inactive, parent is not specified, or parent team name does not match in the package, the import package operation fails.
- If the team already exists in the system but the one in the package has a different parent it skips.
- All the entities that you decide to add in the packages must be present in the system. The association is skipped, if the entities that are associated with the team are not there in the system.
- Inactive teams cannot be exported.

It is recommended to turn off the notification in team management when you create team packages. Otherwise, all the team members will receive notifications. Once the import is complete, you may turn on notifications if you wish.

## Branded Content Libraries

Cisco distributes content in the form of Branded Content Libraries in certain product releases. Contact the Cisco Technical Assistance Center (TAC) if you want to find out the availability of such content libraries for the release you are using. Two types of library packages are available:

- Service Library, containing service definitions and their component entities
- Custom Library, containing entities associated with a Service Library

The services in these libraries offer models for commonly required services for End User Management, Access Management, Data Center Management and other areas. Library contents can be previewed and desired items deployed to a development environment. This provides a head-start for designing, configuring, and customizing these services to enterprise requirements.

Custom libraries contain entities such as queues and service teams (organizations), which are referenced in a service. These libraries can optionally be installed in conjunction with the corresponding service library. If

the Custom Library is deployed, the service will use the predefined references. If the custom library is not installed, the service will refer to the “Default Service Queue” or remove the reference, as appropriate. Service designers are responsible for completing the task (delivery) plan configuration.

## Deploying a Branded Library

Typically, a site administrator is responsible for administering and deploying content from branded libraries. If desired, a custom role which includes the capability to Deploy Branded Content Libraries may be created. See the [Structuring the Organization](#) or *Online Help* for details on creating and assigning roles.

Use the following procedure to deploy branded libraries:

- 
- Step 1** Obtain the libraries. Libraries and corresponding user guides are available for download from the Cisco software site.
  - Step 2** Install the library. The library must be installed on a directory accessible to the client workstation from which Catalog Deployer is run. A shared network drive will allow central storage of libraries.
  - Step 3** Ensure that users responsible for deploying library contents have roles that include this capability.
  - Step 4** Import the library. Import the library into Catalog Deployer. See the [Importing a Library Package](#), on page 37.
  - Step 5** Review the library contents. A person with permission to deploy the library package selects the received package on the target site and optionally previews the library contents (services).  
To preview library content, choose the service to be previewed by checking its check box, and then click **Preview**.
  - Step 6** Deploy chosen library contents. Choose the received package, choose the package contents to be deployed and run the deployment. See the [Deploying a Library](#), on page 39.
  - Step 7** Review the deployment log. Deploying library contents generates a log detailing all deployment activities in regards to creating or updating entity definitions.
- 

Site Administrator and Catalog Publisher roles include the Import Deployments and Deploy Deployment Package capabilities in order to import, preview or deploy content from Branded Content Libraries.

## Importing a Library Package

Importing a library package is analogous to importing a deployment package:

- 
- Step 1** In the View and Search pane, choose **Action > Import**. (If that option is not available, the current user does not have a role that includes the “Deploy Branded Library Content” capability.)  
An import dialog box appears, asking you to browse for the library file to be imported.
  - Step 2** Click **Browse**.
  - Step 3** Find and choose the library file.
  - Step 4** Click **Open**.
  - Step 5** Click **Import**.  
The library is imported, assigned a status of “Received for Deployment”, and opened.
  - Step 6** Close the import dialog box.
-

A library package has attributes which identify the library and guide you in its deployment:


**Table 9: Import library package attributes**

Attribute	Description
Package Name and Description	The name for the library and a brief description of its contents.
Library Version	The version of the library as released by the vendor.
Vendor Name	Cisco, or the name of the content provider.
Application Version	The version of the database for which deployment of the library is supported. Libraries are tailored to a specific version of Service Catalog.
Image	An image icon associated with the library.
Package Type	Service Library or Custom Library.
Status	The current status of the library. Either "Received" or "Deployed".
Created On/By Modified On/By	Catalog Deployer automatically tracks the person who created the library in this site (that is, who imported the library), and the date of the latest deployment activity and who performed it.
History	A brief log of all deployment activities performed against the library in this site with links to more detailed logs for deployment details.

### Creating a Branded Library Package

Licensed users with appropriate roles can create a new library by choosing **Action > New Library Package** in the View and Search pane.

The procedure is identical to the procedure for creating a deployment package (see the [Creating a Deployment Package, on page 22](#)) with the following exceptions:

- **Library Version** and **Vendor Name** are required. These are free-format text fields which identify the library.
- An orange ball icon  is associated with a new library by default and is displayed in the first column of the View and Search pane. A custom image icon may be used by clicking **Upload Image** in the Action pane after the Library Package has been saved. Images can only be in JPG, PNG, or GIF format. They are displayed with a width of 16 pixels and a height of 16 pixels, so they should be created to maintain

this aspect ratio. If the image is not sized correctly, it is resized by the browser and may appear blurry or pixilated.

- The Package Types available are:
  - **Service Library**, containing service definitions and their component entities
  - **Custom Library**, containing entities associated with a Service Library
- The library is also identified by the application version, the version of the Service Catalog database under which it was created. Libraries can only be deployed into a Service Catalog site running the same application version with which they were created.
- Library packages can only be exported, not transmitted. Just as for a deployment package, a library package must be assembled before export.

## Deploying a Library

Libraries are the vehicles for new content to be distributed from Cisco to customer sites. A library should be deployed into a development site only. Catalog Deployer is designed so that libraries can be deployed without disrupting or overwriting existing content that may have previously been customized to the customer's requirements. Therefore, the options for deploying a library package are configured so that library contents can supplement, but not replace, any previously created content with the same name on the target site.

- Primary entities in the package (a service or directory-related entity in a Custom package) will only be deployed if an entity by the same name is not found on the target site. Deploying from a library never replaces existing content.
- A service bundle always includes the parent service definition as well as all child Service Definitions. As is the case for all primary entities, the service definition is skipped (not deployed) if a service with the same name already exists in the target site.
- Component Entities are entities that are automatically deployed along with a primary entity. For example, deploying a service entails deploying the dictionaries, active form components, categories, and keywords used by that service. Such component entities are not deployed if an entity with the same name already exists in the target site.
- Associated Entities are those entities referred to by the primary entity. For example, queues, peoples, and organizations can be referred to by a service task plan, and are the service's associated entity types. If an associate entity exists in the target site at the time the primary entity is deployed, a relationship is established between the entities. If the associated entity does not exist, a default relationship (for example, to the "Default Service Queue") is established.

The entire contents of a library containing supporting entities (such as queues and organizational units) must be deployed simultaneously. Conversely, Catalog Deployer allows you to choose the contents of a Service Library to be deployed. Each service is deployed in a separate transaction. If the deployment fails, the current transaction is rolled back.

All import and deployment actions are logged for the library package. The **View Log** link in the History section displays the actions by Catalog Deployer in detail. See the [Known Errors and Omissions](#), on page 29.

## Terminology

**Table 10: Key Terms**

Term	Definition
Assemble a Package	Add content to a package. Assembling a package extracts the current definitions of the objects which have been included in the package from the repository and writes them to the package. Any subsequent changes to these objects will not be reflected in the assembled package.
Associated Entity	An entity that is related to the primary entity and required in the target site for a deployment to be successful. For example, a service definition (primary entity) may refer to several organizations, service teams to whom tasks are assigned (the organizational unit is the associated entity).
Catalog Deployer	The Service Catalog module responsible for deployment of catalog content and directory information from one site to another.
Component Entity	Entities that are automatically deployed when a service is deployed. Service component entities include categories, presentation elements, dictionaries, dictionary groups, service groups, keywords, and objectives.
Data Source	A data source, defined in the JDBC data source page on the application server console, which specifies connection information for all sites which become targets for Catalog Deployer direct site-to-site deployment.
Deployment Package	The primary object managed by Catalog Deployer. The deployment package contains the chosen entities that are to be deployed (such as service definitions or queues and groups) and the deployment activity history. A deployment package can be transmitted or exported/imported into another site for deployment of its contents into the target site. A deployment package does not contain entity content until it is "assembled," at which point the included data is extracted from the source site.
Entity	One of the objects created and maintained within Service Catalog software. Examples are: organization units, service definitions, queues, and email templates.



Term	Definition
Entity Home	The page of the Administration module's settings where administrators specify which site is the site of record for each supported entity type.
Export a Package	Create an XML file containing the contents of a previously assembled package and write the file to the file system. Exporting a package from a source site to a target site is an alternative to transmitting the package.
Implementation	A group of one or more Service Catalog sites, either directly or indirectly connected.
Import a Package	Create a package by importing a previously exported XML deployment package into a target site. The package is created with the status "Received for Deployment".
Library Package	A package from Cisco that has been preassembled and is available as a file import. The package content consists of template services or associated entities that can be used to provide the basis for a service catalog. Unlike deployment packages, chosen contents of a library can be deployed.
Primary Entity	An entity that can be chosen for inclusion in a Catalog Deployer package.
Site	A collection of one or many computer systems (single computer or cluster) that share a database and an http address, and can be categorized by function. For instance: a development site, a testing site, and a production site.
Source Site	The site in which a package is created. This is the site which transmits or exports a package for deployment on another site.
Target Site	The site in which a package is deployed. This is the site which receives a package via transmission or import.
Transmit	Send a package from one site to another via a database connection.  Once a job has been transmitted, it cannot be edited (reassembled and have its content changed) on the source site. The target site receives the transmission via Catalog Deployer.

