MICRO FOCUS

Deployment Automation
User's Guide

Part number: 6.3

Publication date: 2019-09-19

Table of Contents

Part 1: Introduction

This section contains introductory information on Deployment Automation. Included are the following topics:

Chapter 1: Welcome to Deployment Automation

Deployment Automation enables you to automate the deployment of application changes. Benefits include continuous delivery and DevOps automation, reduction of development costs, and increased deployment frequency without increased risk.

About This Documentation

This documentation guides you through installing and using the Deployment Automation product and is intended for all users. This information is also included in the User's online Help in HTML format.

The information is organized into parts to help you find the information you need.

Chapter 2: Deployment Automation Overview

Deployment Automation automates software deployment, which is the process of moving software through various pre-production stages to final production. Typically, each stage represents a step of higher criticality, such as quality assurance to production.

Software deployment complexity increases with more releases to deploy, more deployment targets, more types of deployment targets, shortened deployment cycles, and changes in technology.

Deployment Automation helps you meet the deployment challenge by providing tools that improve deployment speeds while simultaneously improving their reliability.

With Deployment Automation, you can:

- Model processes that orchestrate complex deployments across every environment and approval gate with complete visibility into *n*-tiered deployments

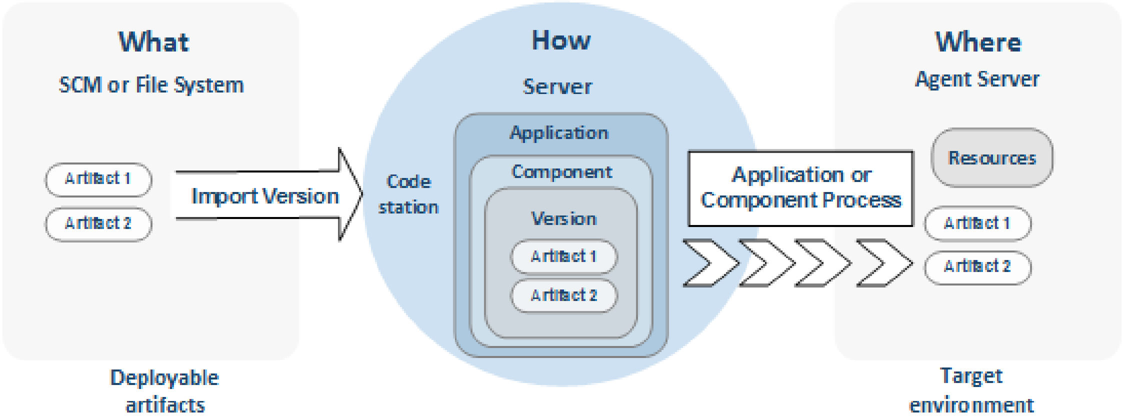
- Visualize the end-to-end deployment process and develop the big picture, the *What*, *How*, and *Where* of the deployment workflow, using drag-and-drop design tools

  - **What**: the deployable artifacts that Deployment Automation delivers to target destinations: binaries, static content, middleware updates, database changes and configurations, and anything else associated with the software.

  - **How**: by combining deployable artifacts with processes to create components and designing applications that coordinate and orchestrate multi-component deployments.

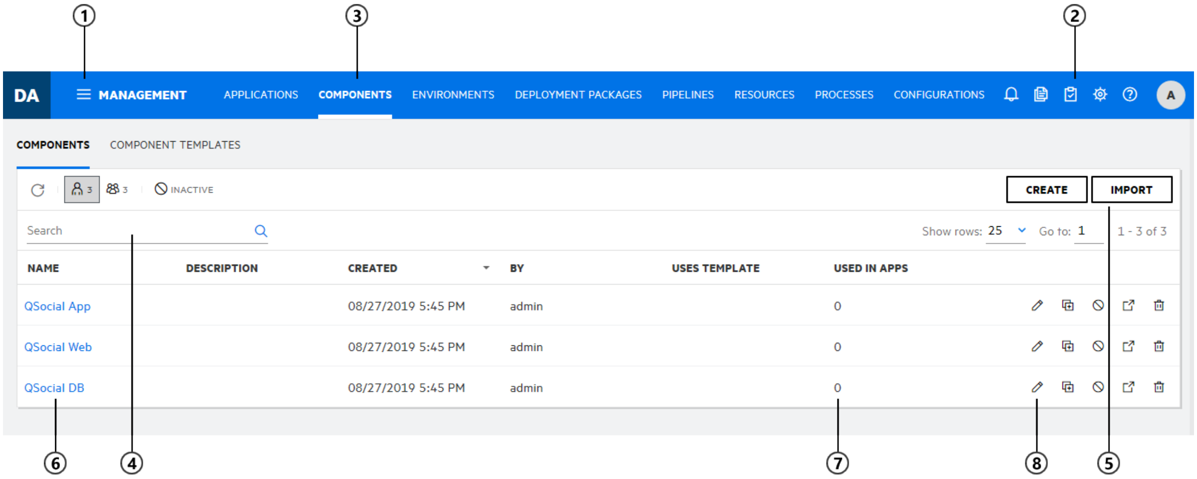
  - **Where:** the target destination's hosts and environments

Deployment Automation



User Interface Overview

Deployment Automation enables you to create and configure elements and initiate the deployment of your component versions. You can only view and access the elements for which you are authorized. These are some of the common user interface elements:



NAME	DESCRIPTION	CREATED	BY	USES TEMPLATE	USED IN APPS
QSocial App		08/27/2019 5:45 PM	admin		0
QSocial Web		08/27/2019 5:45 PM	admin		0
QSocial DB		08/27/2019 5:45 PM	admin		0

1. Click the navigation button ☰ and select the module you want to work in. Depending on your role and privileges, you will see some or all of the following modules:
   - **Management**: create and configure the primary elements of Deployment Automation, design your processes, and run deployments.
   - **Deployment:** view information about deployments on a timeline, run deployments, and schedule deployments and blackouts.
   - **Administration:** configure selections that will be available to users when they work with Deployment Automation .
2. The dashboard is shown at the top of the page. Options include:
   - **Version Import Queue:** view active version import requests.
   - **Pending Approval Requests:** view requests awaiting approval.
   - **Processes:** view processes executed within a given range of time.
   - **User Profile menu:** select from options to configure preferences such as locale, get help, visit the user community, view information about the product version, and sign out.

   In the dashboard you may also see a message indicator. Click to see important messages such as upgrade suggestions for your server and agent versions.
3. Select an option from the navigation menu to view and manage information. This example shows **Components** selected.
4. To filter the information shown in the content area, click the filter buttons to select or toggle filter criteria. The filter Search box searches various columns and further filters the list by showing only items that match the search string. In this example,

the list is filtered by components that belong to the current user, and inactive components are not shown.

5. Create or import an instance of the selected element.

6. The content area of the page. In this example, the list of components is shown.

7. Click any links in the content area to see more information. For example, for components, click the number under **Used in Apps** to see the names of the applications the component is associated with.

8. Select from actions available for listed items. For example, in the component list click the **Delete** button to delete a component.

Examples and the Community

After you have installed and configured Deployment Automation, log in to the user interface.

Get started by configuring and running a simple deployment. See the *Deployment Automation Walk-through* or visit the Community website to get pre-configured processes that you can import and run. A number of videos and other resources are also available on the Community website.

Terminology

Before you get started with Deployment Automation, here are some terms you should know.

- **Application**

  Applications bring together components with their deployment targets and orchestrate multi-component deployments.

- **Component**

  Components typically map to a functional part of a real world application, and represent deployable items, or artifacts, such as files, images, databases, and configuration materials.

- **Plugins**

  Plugins provide functionality in discrete steps to be used in component and global processes for configuration of or deployment into target environments.

- **Environment**

  Environments represent logical deployment locations. Your deployment processes must run in at least one environment. Environments and their resources are used by applications and components at runtime.

- **Resource**

  Resources represent a deployment target on a Deployment Automation environment. Examples include physical machines, virtual machines, databases, or J2EE containers.

- **Pipeline**

A pipeline is a pre-defined sequence of environments in which application process requests are executed.

- **Agent**

  Agents are physical resources for deployment. To run a deployment, an agent must be installed on the target server.

- **Agent Relay**

  Agent relays are used to manage communication between servers and agents. Agent relays are typically used when agents are dispersed across geographic locations or must communicate through firewalls. Agent relays can also be used to manage network traffic in implementations where you have many agents.

- **Deployment Package**

  Deployment packages enable you to deploy artifacts for multiple applications. They may also include component processes where components are shared among multiple applications and associated versions are to be deployed as part of the larger package.

A Typical Sequence

Although there is no set order, a typical sequence for configuring Deployment Automation elements and deploying artifacts is as follows:

1. Create components and set your source configuration type and version import approach. The source configuration type points to where the artifacts for a component are stored. You can import the artifacts into Deployment Automation CodeStation automatically or manually.

2. Create and design component processes, defining the processes you want performed on target machines before, during, and after deployment of the component artifacts.

3. Create resources to represent logical locations on agent machines.

4. Create environments and assign resources to them.

5. Create and configure pipelines, sequencing environments and adding approvals.

6. Create an application, selecting the pipeline it will use.

7. Add components to the application.

8. Add environments to the application that are not included in the pipeline and map components to the environment resources they will use.

9. Create and design an application process to control the deployment. An application process typically initiates multiple component processes.

10. Run the application process to deploy the components.

11. Create and design deployment packages to run multi-application deployments.

Related Topics

Creating Components [page 200]

Designing Component Processes [page 241]

Creating Resources [page 308]

Creating Environments [page 298]

Mapping Agents and other Resources to Environments [page 299]

Configuring Pipelines [page 304]

Application Configuration [page 265]

Adding Environments to Applications [page 270]

Mapping Application Components to Resources [page 273]

Designing Application Processes [page 276]

Chapter 28: Deployment Overview [page 365]

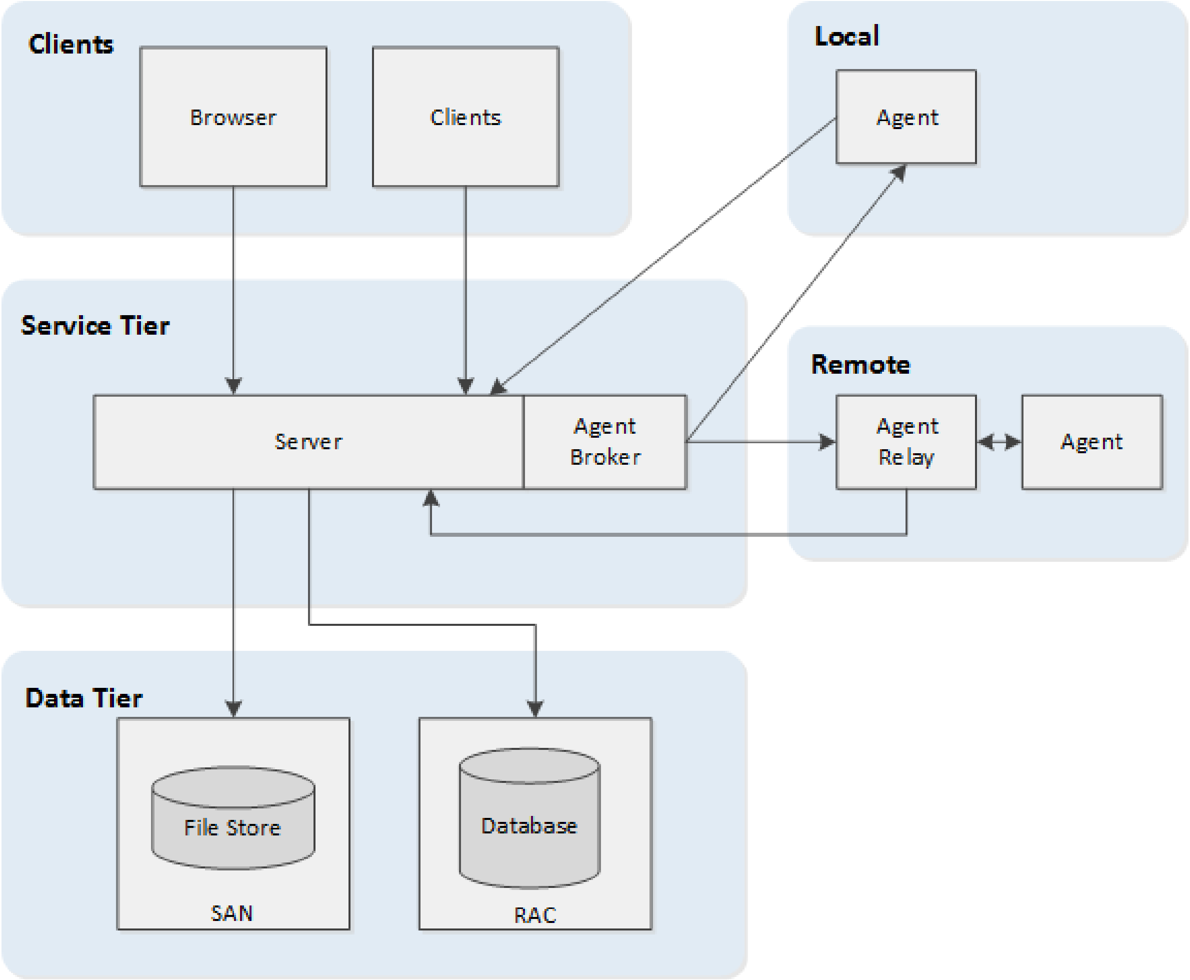
Part 2: Installation

This section contains information on installing and configuring Deployment Automation as detailed in the following topics:

Chapter 3: Architecture Overview

Deployment Automation architecture consists of clients, a service tier, a data tier, and local and remote agents. Local agents are agents that communicate directly with the server; remote agents are those that communicate with the server through agent relays.

Deployment Automation Architecture



The following topics give more information on the architecture.

- Clients [page 30]
- Data Tier [page 30]
- Service Tier [page 30]
- Agent Architecture [page 31]
- Agent Communication [page 32]

- Agent Relay Communication [page 33]
- Services [page 34]

Clients

Deployment Automation clients include the following:

- *Web browser-based user interface*: a Rich Internet Application (RIA) that maintains much of its functionality in the web browser
- *Command line*: provides most of features found in the Web browser-based user interface

Clients:

- communicate with the server through HTTP or HTTPS
- may be deployed locally, on the same LAN as the server
- may be deployed remotely
- interact with RESTful (representational state transfer) services on the server as needed

Note: WebSocket communication is used for notification counters shown in the upper right of the web browser-based user interface.

Data Tier

The data tier's relational database stores configuration and run-time data. The data tier's file store, CodeStation, contains log files, artifacts, and other non-structured data objects. Reporting tools can connect directly to the relational database.

Service Tier

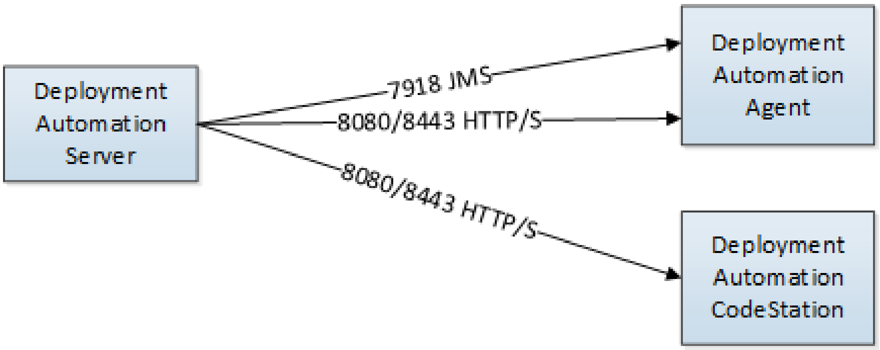
The service tier has a central server that provides a web server front-end and core services, such as workflow, agent management, deployment, inventory, and security. Workflow requests are initiated with either the user interface, the command line interface (CLI), or through REST service calls through integrating products, such as SBM.

Deployment Automation uses stateless communications for the JMS-based server-agent communications as well as client-web service communications. Stateless, as used here, means the server retains little session information between requests and each request contains all the information required to handle it.

The server sets up listening sockets and listens for agents, relays, and users (clients). By default the server listens on only three ports: port 7918 for JMS, 8080 for HTTP, and 8443 for HTTPS.

The default server communication is shown in the following figure.

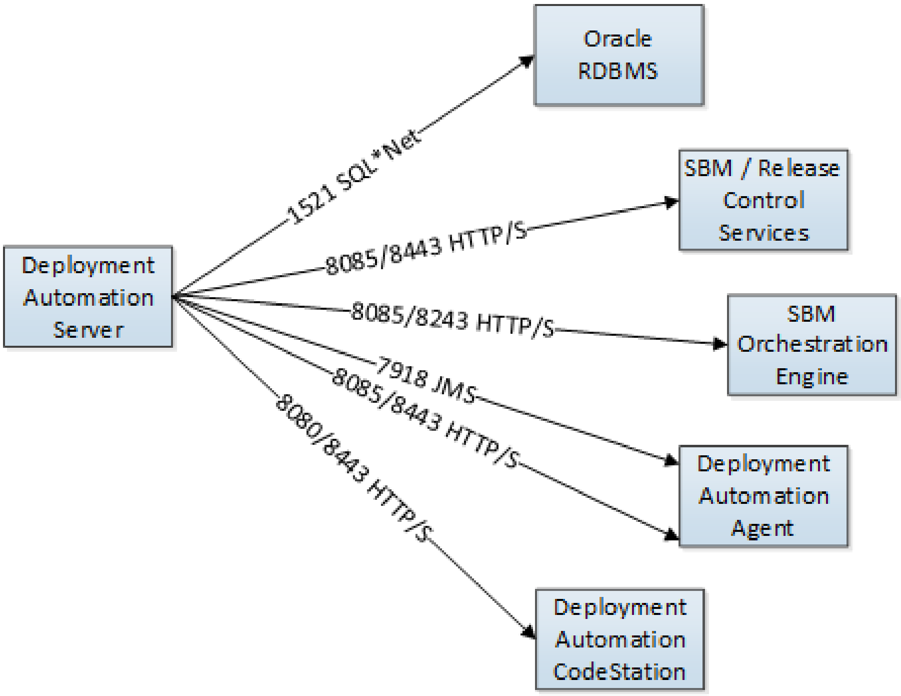
Default Server Communication


Deployment Automation Server
7918 JMS
8080/8443 HTTP/S
8080/8443 HTTP/S
Deployment Automation Agent
Deployment Automation CodeStation

The actual ports used depend on the specific implementation, and if Deployment Automation is used natively with an SBM solution such as Release Control, the ports used to access Deployment Automation would be the same ports as those used for the SBM solution.

The communication mechanisms and default ports used by the server to access an Oracle relational database management system, Release Control and SBM, agents or agent relays, and CodeStation are shown in the following example.

Extended Server Communication



Agent Architecture

All processes, packaging, configuration, deployments, and so on, requested by the Deployment Automation server are executed on hardware assigned to agents.

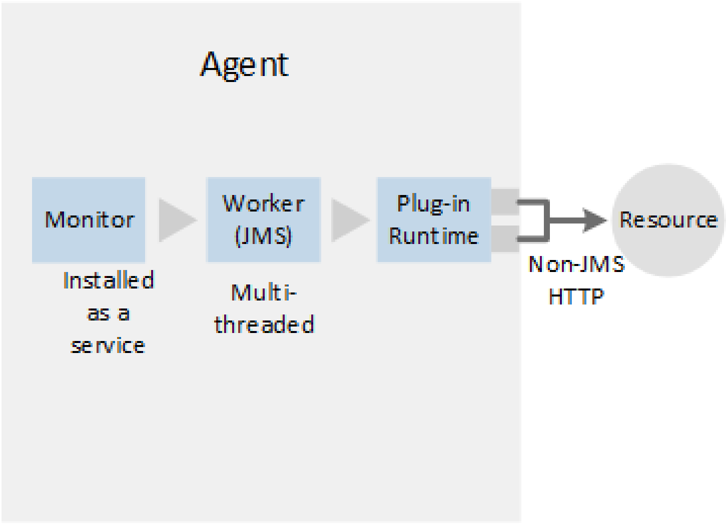
Typically, an agent runs on the host where the resources it handles are located. A single agent can handle all resources on its host. If a host has several resources, an agent process is invoked separately for each resource. For example, a test environment might contain a single web server, a single middleware server, and a single database server all running on the same host machine. A deployment to this environment might have one agent and three separate resources.

Agents are an important part of Deployment Automation scalability. By adding more agents, the throughput and capacity of the system increases almost exponentially and so can scale to fit even the largest enterprise.

Depending on the number of hosts in an environment, a deployment might require a large number of agents. Agents are unobtrusive and secure. Once an installed agent has been started, the agent opens a socket connection to the Deployment Automation server. Communication between server and agents uses a JMS-based (Java Message Service) protocol and can be secured using SSL, with optional mutual key-based authentication for each end-point. This communication protocol is stateless and resilient to network outages. The benefits of statelessness are discussed in Agent Communication [page 32].

While we characterize an agent as a single process, technically an agent consists of a *monitor* process and a *worker* process, as shown in the following figure.

Agent Processes



- The *monitor* is a service that manages the worker process, for example, starting and stopping and handling restarts, upgrades, and security.
- The *worker* is a multi-threaded process that performs the actual deployment work after receiving commands from the server.
- Work commands come from *plugin* steps, which provide seamless integration with many third-party tools.

Agent Communication

Most agent communication is done with JMS, but some agent activities use the web tier through HTTP as needed. For example:

- The server uses JMS to send agent commands.
- The agent monitor service uses JMS for all server communications and for sending commands, such as "run step," to the worker process.

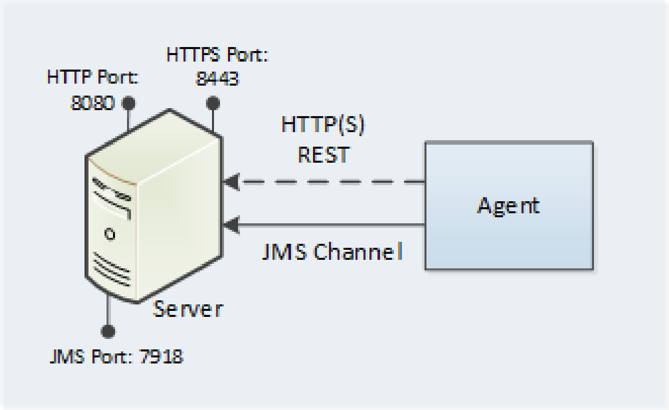
- The worker process uses JMS for system communications, and HTTP REST services when performing plugin steps or retrieving information from the server.
- Agent activities such as posting logs, transmitting test results, and posting files to CodeStation use the web tier through HTTP or HTTPS
- Most clients use browsers to communicate with the web server through HTTP or HTTPS

Stateless server-agent communication provides significant benefits to performance, security, availability, and disaster recovery. Because each agent request is self-contained, a transaction consists of an independent message that can be synchronized to secondary storage as it occurs. Either the server or agent can be taken down and brought back up without repercussion, other than lost time. If communications fail mid-transaction, no messages are lost.

Once reconnected, the server and agent automatically determine which messages got through and what work was successfully completed. After an outage, the system synchronizes the endpoints and recovers affected processes. The results of any work performed by an agent during the outage are communicated to the server.

In the following figure, the arrow represents the direction in which the stateless communication was established, but the flow is in both directions with JMS.

Stateless Communication



For added security, agents do not listen on ports. Agents send requests when they are ready to make the transition to a new state. Because JMS connections are persistent and not based on a request-response protocol, Deployment Automation does not have to continually open and close ports, which enables the server to communicate with agents at any time while remaining secure and scalable.

REST-style services achieve statelessness by ensuring that requests include all the data needed by the server to make a coherent response.

Agent Relay Communication

As long as there is at least a low bandwidth WAN connection between the server and remote agents, the server can send work to agents located in other geographic locations. To aid performance and ease maintenance, Deployment Automation uses agent relays to communicate with remote agents. When you configure agents to connect though an agent relay, this requires only a single machine in the remote network, the agent relay, to contact the server. This enables all agent-server communication from the remote network to go through the agent relay to the server.

The following, a simple artifact move, illustrates the mechanics of remote communications:

1. A remote agent starts and establishes a connection to the agent relay through JMS, and the agent relay then alerts the server through JMS that the remote agent is online.

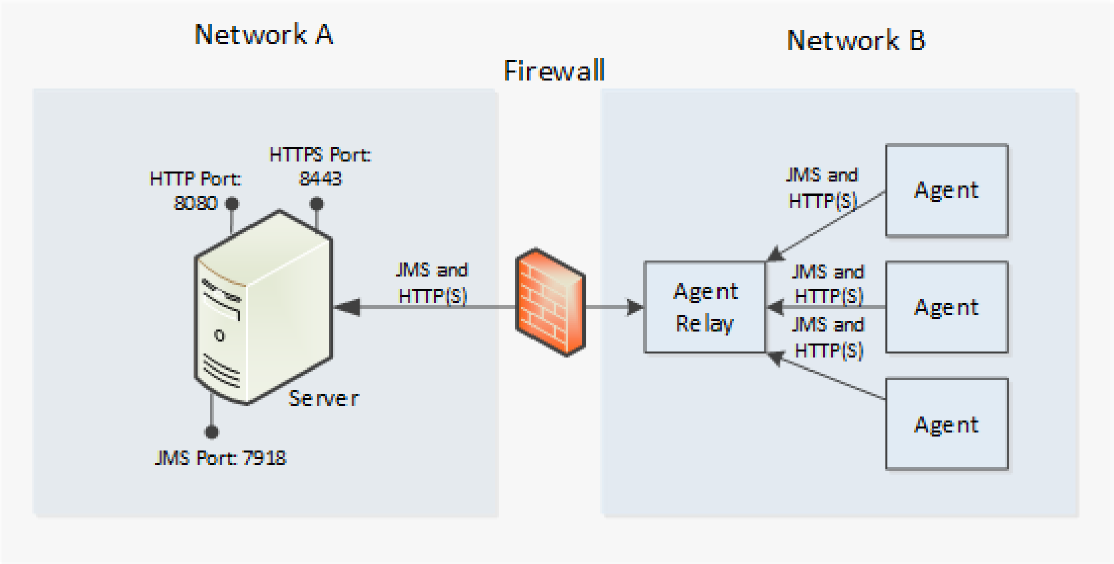
2. The server sends an artifact download command to the agent relay through JMS, and the agent relay delivers the message to the remote agent, also through JMS.

3. The server locates the artifacts, and then:

    a. Uploads the artifacts over HTTP(S) to the agent relay, which begins streaming them directly to the agent over the server-to-agent relay HTTP(S) connection.

    b. Once the remote agent completes the work, it informs the server through JMS.

Agent Relays in a Cross-Network Configuration



Remote agents open connections to the agent relay. By default, agent relays open the connection to the server, but the direction can be reversed to be server-initiated if your firewall requires it. See Server-initiated HTTP Communication Using Agent Relays [page 74].

Services

Deployment Automation is a services-based system. This section explains what happens behind the scenes when a deployment process is requested in the Deployment Automation user interface. Use this section to understand deployment process runtime activities, to help troubleshoot any execution issues, and to gain an understanding of integration points.

The Deployment Automation server provides a variety of services, such as:

- the user interface
- component and application configuration tools
- workflow engine
- security services

A service can be thought of as a self-contained mechanism for hosting a piece of business logic. Services can be consumed by clients, agents, or other services.

Many Deployment Automation services are REST-type (representational state transfer). REST-type services are web services that focus on transferring resources over HTTP.

Workflow requests are initiated with either the user interface user interface or the CLI (command line interface). When a workflow is requested, many services are used to fulfill the request.

Services and Process Workflow

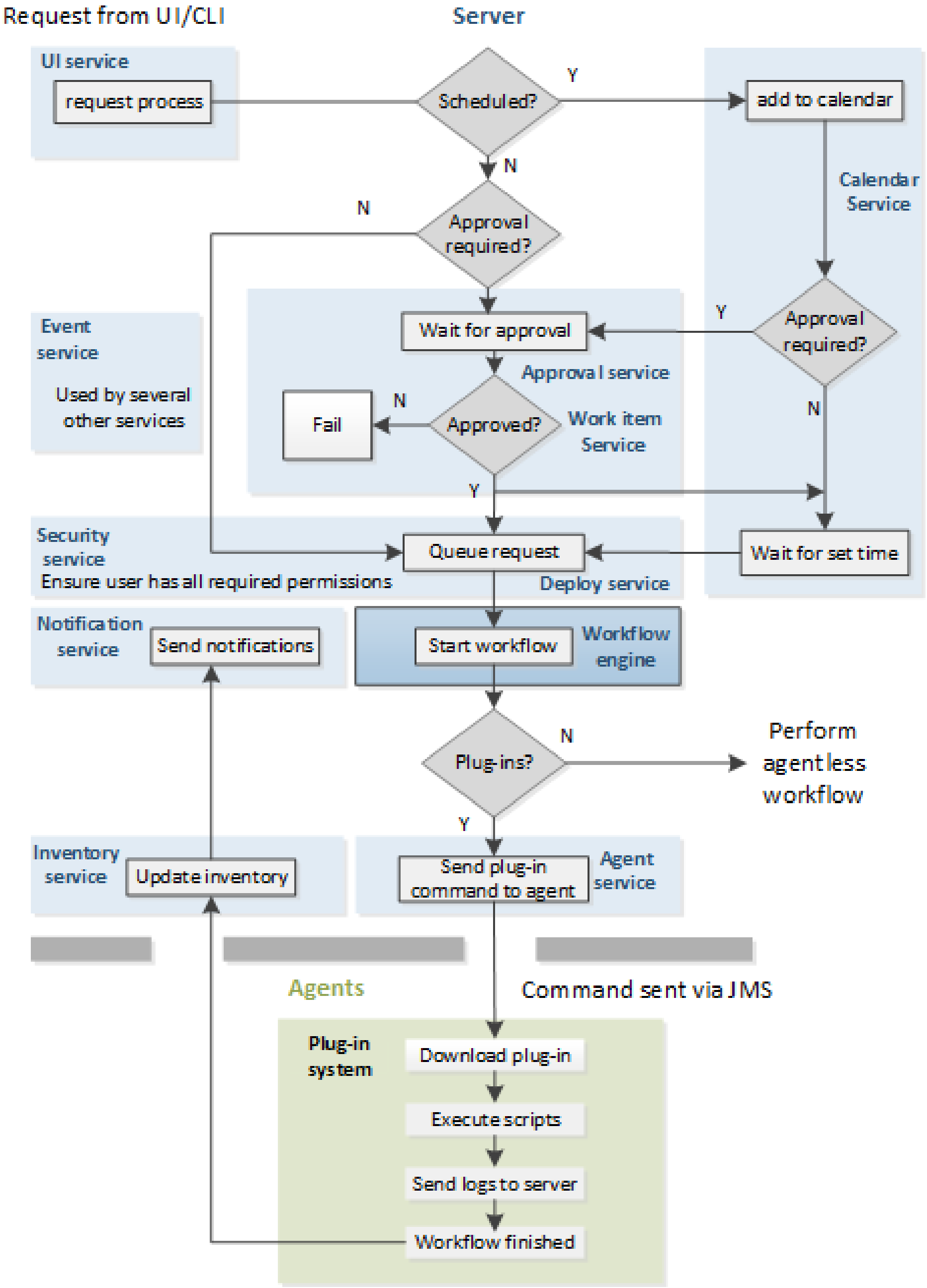


Table 1. Services Table

Service	Description
User Interface	Used to create components and fashion workflows, request processes and manage security and resources, among other things. REST-based.

Service	Description
Workflow Engine	Manages workflows, application and component processes. Calls the agent responsible for performing the workflow's current plugin step. When the workflow is finished, alerts the notification and inventory services. Called by the deploy service. REST-based.
Agent	Tracks installed agents and routes plugin commands to affected agents. Commands come from plugin steps. Invoked by the workflow service. REST-based.
Work Item	Operates in tandem with the approval service; provides approver alerts and enables approvers to accept or reject workflows. If a scheduled workflow remains unapproved at run-time, the job fails automatically. REST-based.
Plugin Manager	Deployment Automation can interact with virtually any system through its extensible plugin system; plugins provide functions by breaking-down tool features into automated steps. Plugins can be configured at design- and run-time. When a plugin step executes, the controlling agent invokes its run-time process to execute the step. When a new component version is available, the agent compares the current component version and downloads and only new or changed artifacts.
Event	The event service is ubiquitous; it alerts other services as various trigger conditions occur.
Deployment Service	Manages deployments. When a deployment process is requested, invokes the workflow engine to perform the process. Works in tandem with the security service to ensure users have required permissions. REST-based.
Notification Manager	Notifies users about the status of deployments; notifications are sent to approvers if the system is configured with an email server and the user has an email address. Invoked by the workflow manager. REST-based.
Inventory Manager	When a workflow finishes, the inventory manager updates affected inventory records. Deployment Automation maintains an inventory of every deployed artifact in every environment, which provides complete visibility across environments. REST-type service.
Approval Engine	Enables creation of approval-requiring jobs and approver roles. Works in tandem with the work item service to ensure required approvals are made before scheduled jobs. REST-based.
Security	Controls what users can do and see; maps to organizational structures by teams, roles, activities, etc. REST-based.

Service	Description
Calendar	Used to schedule processes to being at some future point; works in tandem with the approval and work item services. REST-based.
CodeStation	Handles versioning of artifacts; agents invoke it when downloading component versions. REST-based.

Chapter 4: Installing Servers and Agents

The following topics lead you through a Deployment Automation installation.

- Installation Checklist [page 39]
- Installation Considerations for Optimal Performance [page 41]
- System Requirements [page 41]
- Preparing Your Database [page 43]
- Server Installation [page 44]
- Agent Relay Installation [page 61]
- Agent Installation [page 75]
- Installing for a High Availability Implementation [page 88]

Installation Checklist

This checklist gives an overview of each of the installation steps for a basic installation and links to the detailed procedures.

Tip: Keep this open or print it and follow along as you install to ensure you perform all the necessary steps.

Note:

For instructions on upgrading an existing server installation, see the Deployment Automation Release Notes.

A single-server installation consists of a server, a database, and at least one agent. The steps to install a single-server installation are as follows:

1. **Review installation recommendations and system requirements**

   Review the installation considerations and system requirements.

   See Installation Considerations for Optimal Performance [page 41] and System Requirements [page 41].

2. **Create your database**

   Create an empty database for Deployment Automation. If you want to install Deployment Automation for evaluation purposes, you can install the lightweight Derby database that is provided by the installer.

   See Preparing Your Database [page 43].

3. **Download installation files**

   If you haven't already, from the Support website download the server and agent installers and any others you want to use, such as the agent relay installer or command line client zip file. Some of the installers are specific to platform and there are 32-bit and 64-bit versions. Extract compressed (zipped) files before running the embedded executables.

4. **Install the server**

   Install the server using the Deployment Automation server installer for your platform.

   The installer automatically installs or points to the supporting application server and JRE. It will prompt you to supply values for the IP address, ports for HTTP communication, and other connection information.

   Note: (UNIX/Linux) Root privileges are required to install the Deployment Automation server.

   See Server Installation [page 44].

5. **Install agent relays (optional)**

   If you are using an agent relay, install the agent relay using the Deployment Automation agent relay installer for your platform. Agent relays are not needed in a simple installation, but are needed if the server and agents need to communicate over firewalls or if you are configuring a high availability, enterprise-level implementation.

   See Agent Relay Installation [page 61].

6. **Install agents**

   Install agents on target machines that will communicate with the server using the Deployment Automation agent installer for your platform. When installing an agent, you supply several values defined during server installation. If you are using an agent relay, the agent relay must be installed first so that you can specify the agent relay host information for the agent to use for connection.

   Solaris: JRE version 8 or higher is required.

   See Agent Installation [page 75].

7. **Validate the installation**

   Start the server and agents. To determine if the agent is in communication with the server, in the user interface, navigate to `Management > Resources > Agents`. A value of `Online` in the agent's Status field means the agent is successfully connected.

   See Chapter 5: Running Deployment Automation [page 99].

8. **Configure your Deployment Automation system as needed**

   Configure Server Communication, Single Sign-On (SSO), Secure Socket Layer (SSL), File Versioning, Users, Groups, Roles, and so on as needed for your organization. This may include loading additional automation plugins to use in your component processes. See corresponding sections for details as listed in Part 3: Administration [page 103].

Installation Considerations for Optimal Performance

For optimal performance, do the following:

- **Install the server as the root user or system user account.** The server should be installed as the root user on UNIX/Linux and as a local system user on Windows. Running in this manner avoids all permission errors. Deployment Automation needs this level of permissions because it writes into protected location such as `/opt` and `/var/opt`.

- **Install each agent as a dedicated system account.** Ideally, the account should only be used by Deployment Automation. Because Deployment Automation agents are command execution engines, it is advisable to limit what they can do on host machines by creating dedicated users and then granting them appropriate privileges. If you install an agent as the root user (or local system user on Windows), ensure that agent processes cannot adversely affect the host file system.

- **Install a single agent per host machine.** If multiple agents are running on the same machine, they can negatively impact each other's performance. If you install multiple agents on the same machine, you must ensure the machine has adequate capacity to meet the system requirements for all of the agents. Otherwise, you may see performance degradation when multiple agents are busy simultaneously.

System Requirements

Deployment Automation will run on Windows and UNIX-based systems.

While the minimum requirements provided are sufficient for an evaluation, you must use server-class machines for production deployments.

The Deployment Automation server uses threading and takes advantage of any additional CPU cores assigned to it. A small to midrange server with 2 to 4 multi-core CPUs is ideal. However, because you can move an existing Deployment Automation server installation to a new machine, starting small and scaling as needed is a viable strategy. The memory available to the application tier should also be increased from the default 256 MB to approximately 1 GB.

Specific requirements are given in the following topics:

- Server Minimum Requirements [page 41]
- Server Recommended Configuration [page 42]
- Agent Minimum Requirements [page 43]
- Agent Relay Minimum Requirements [page 43]

Server Minimum Requirements

The minimum server installation requirements are as follows. The minimum configuration should be used only for demonstration or evaluation purposes.

- **Processor**: Single core, 1.5 GHz or better
- **Memory**: 2.5 GB or more, with 256 MB available to Deployment Automation
- **Disk Space**: 300 MB or more

- **Java version**: Java JRE or JDK 1.8.0.25 or later; either 32-bit or 64-bit as is appropriate to the operating system.

  Note: This is installed automatically as part of the server installation.

Note: For a list of supported platforms, see the Supported Platform List for your version of Deployment Automation on the Support website.

Server Recommended Configuration

The recommended server installation configuration is as follows. The recommended configuration should be used for enterprise-level implementations.

- **Multiple server-class machines**: For production environments, it is recommended to use a multiple-server, active-active approach to distribute the processing load of agent communication.

- **Separate machine for the database**

- **Network** Gigabit (1000) Ethernet with low-latency to the database.

- **Processor**: 2 CPUs, 2+ cores for each, with 1.5 GHz or better for each core

- **Memory**: 8 GB or more

- **Disk Space and Database Storage**: Individual requirements depend on usage, retention policies, and application types. The amount of storage depends on the number of artifacts kept in the Deployment Automation artifact repository (CodeStation).

  Note: CodeStation is installed when the server is installed.

  For production environments, use the following guidelines to determine storage requirements:

  - 10-20 GB of database storage should be sufficient for most environments.

  - To calculate CodeStation storage requirements:

    ```
    average artifact size * number of versions imported per day * average
    number of days before cleanup
    ```

  - Approximately 1MB per deployment of database storage; varies based on local requirements.

  For further assistance in determining storage requirements, contact Support.

- **Java version**: Java JRE or JDK 1.8.0.25 or later; either 32-bit or 64-bit as is appropriate to the operating system.

  Note: This is installed automatically as part of the server installation.

Agent Minimum Requirements

Designed to be minimally intrusive, agents require the following minimum system configuration:

- **Processor**: An idle agent uses about 5MHz of CPU. Additional requirements are determined by the processes the agent will run.
- **Disk Space**: 100 MB
- **Memory**: 256 MB (768 MB recommended)
- **Java version**: Java JRE or JDK 1.7.0.25 or later, 1.8.0.25 or later; either 32-bit or 64-bit as is appropriate to the operating system.

Important: To restrict security protocols, you must use version Java 8.0.

Agent Relay Minimum Requirements

Agent relays, if used, require the following minimum system configuration:

- **Processor**: Single core, 1.5 GHz or better
- **Disk Space**: 100 MB
- **Memory**: 256 MB (768 MB recommended)
- **Java version**: Java JRE or JDK 1.8.0.25 or later; either 32-bit or 64-bit as is appropriate to the operating system.

Preparing Your Database

Your relational database is a critical element for performance and disaster recovery. Consult your DBA, and ensure that your database is configured for high availability and high performance and that database will be backed up regularly. For details on supported versions, see the Supported Platform List for your version of Deployment Automation on the Support website.

Before you install Deployment Automation, you must create an empty database in your DBMS. If you are evaluating Deployment Automation, you can install the database on the same machine that will host the Deployment Automation server. You can optionally use the lightweight Derby database that is provided by the installer for evaluation purposes. If you select the built-in Derby database, you do not need to prepare the database before you run the installer; the database is created as part of the installation.

Creating a SQL Server Database

For SQL Server databases, a clustered configuration is recommended. When you are ready to install the Deployment Automation server, you will need to provide the values you enter here, including the SQL Server connection information and the `da` user account that has table creation privileges.

For more information, see http://technet.microsoft.com/en-us/sqlserver/default.

To create the SQL Server database, execute the following commands:

```
CREATE DATABASE da;
USE da;
CREATE LOGIN da WITH PASSWORD = 'password';
CREATE USER da FOR LOGIN da WITH DEFAULT_SCHEMA =
da;
CREATE SCHEMA da AUTHORIZATION da;
GRANT BACKUP DATABASE, BACKUP LOG, CREATE DATABASE,
 CREATE DEFAULT, CREATE FUNCTION, CREATE PROCEDURE,
 CREATE RULE, CREATE TABLE, CREATE VIEW to da;
```

Creating an Oracle Database

For Oracle databases, an architecture based on Oracle RAC is recommended. When you are ready to install the Deployment Automation server, you will need to provide the values you enter here, including the Oracle connection information and the `da` user account that has table creation privileges.

Use the instructions for your Oracle DBMS version in the following sections.

Oracle 12 and later

To create the database in Oracle 12 or later, execute the following commands:

```
ALTER SESSION SET "_ORACLE_SCRIPT"=true;
CREATE USER da IDENTIFIED by password;
GRANT CONNECT TO da;
GRANT RESOURCE TO da;
GRANT UNLIMITED TABLESPACE TO da;
```

Oracle 10g or 11g

To create the database in Oracle 10g or 11g, execute the following commands:

```
CREATE USER da IDENTIFIED by da;
GRANT CONNECT TO da;
GRANT RESOURCE TO da;
```

Server Installation

This section tells how to install the Deployment Automation server using the Deployment Automation installer.

The server provides services such as the user interface used to configure application deployments, the work flow engine, the security service, and the artifact repository, among others. After installing the server, you must add a license and configure the server according to your implementation needs.

Important: If you are installing the server in a production environment, install and configure the database you intend to use *before* installing the server. See Preparing Your Database [page 43]. Otherwise, you can use the built-in Derby database.

You typically install using the interactive installer.

Upgrading Servers

If upgrading the server, use the server installer. The **Upgrades** section gives more information, including the versions from which you can upgrade directly. See the Release Notes for this version on Documentation Center.

The following topics lead you through a server installation.

- Running the Interactive Installer [page 45]
- Manual Server Installation [page 48]
- Silent Mode Server Installation [page 50]
- Uninstalling Servers [page 58]
- Reconfiguring Server Installation Settings [page 59]

Running the Interactive Installer

The Deployment Automation interactive server installer uses a wizard that guides you through the installation process.

To install Deployment Automation:

1. Download and unzip the installer zip file.
2. Launch the Deployment Automation installer.

   For Windows

   Run the following installer executable as a Windows local system user:

   - Windows 64-bit: `MicroFocusDA-Server64.exe`

   For UNIX/Linux

   Run the following installer executable as a user with root privileges:

   ```
   MicroFocusDA-Server.bin
   ```

   Note: The interactive server installer is available only for Linux and AIX. For other UNIX platforms, you must manually install. See Manual Server Installation [page 48].

   After the welcome dialog box appears, click **Next**
3. Accept the license agreement. Click **Next**.
4. Specify the path where you want to install Deployment Automation. Click **Next.**
5. In the **General Program Settings** dialog box, select the location where settings will be saved. Click **Change** to select a different directory. Configure the following options:

   - **Use existing settings**

     Select this option to install the server using existing settings and maintain the current database.
   - **Skip database creation**

   Select this option if you are installing Deployment Automation on an additional server for high availability and you do not want to overwrite the settings that you provided during the initial install.

   - **JMS connection port**

     Enter the JMS port that agents will use to connect to the server.

   - **Agent Mutual Authentication**

     Select this option to configure increased security between agents and the server.

   Click **Next**.

6. In the **Database Selection** dialog box, select the type of database you will use. Click **Next**.

7. In the **Database Details** dialog box, edit the **Database Connection String** for your database, and then enter the database user name and password.

Database	Database Connection String
Derby	To use the built-in Derby database (for evaluation purposes only), enter a password for the database user, and then click **Next**.
SQL Server	Use the format: `jdbc:sqlserver://`*`serverName`*`:1433;DatabaseName=da`

Oracle	If you are using a *service name*, use the format:
	`jdbc:oracle:thin:@//`*`serverName`*`:1521/da`
	If you are using an *instance name*, use the format:
	`jdbc:oracle:thin:@`*`serverName`*`:1521:da`
	Where `da` represents your Oracle SID.
	Note:
	For Oracle databases, the schema name must be the same as the user name.

Tip: If you need to change the database password after the install, navigate to the profile location `conf\server` directory and update the `hibernate.connection.password` property. By default for Windows this is in the following file:

```
C:\Users\username\.microfocus\da\conf\server\installed.properties
```

After the password is updated, restart the Common Tomcat where the Deployment Automation server is installed. The `hibernate.connection.password` property will be encrypted after the server starts if it is not already encrypted.

After you have entered the connection information, click **Next**.

8. In the **Administrator Details** dialog box, enter the user name and password to create a Deployment Automation administrator account. You will use this user name and password to log in to Deployment Automation for the first time.

   Note: You cannot change this account later; therefore, enter the user name according to your organization's naming standards.

   Click **Next**.

9. In the **Configuration Details** dialog box, enter the port number for Common Tomcat. The default is 8080.

   Note: For Windows installations, Common Tomcat is run under the Windows System Account by default. If it's necessary to run this under a specific user, such as when you are using network shares that Deployment Automation must have permission to access as part of its core processing, you must change the login information in the Common Tomcat service properties. The default service name is **Micro Focus Common Tomcat 8**. For details on changing Tomcat service properties, see the Apache Tomcat documentation.

   Click **Next**.

10. Click **Install**. The Deployment Automation server installation finishes and the summary dialog box appears.

The properties set during the server installation are recorded in the `installed.properties` file located in the profile directory. By default for Windows this is:

```
C:\Users\username\.microfocus\da\conf\server
```

Deployment Automation is installed and connected to your database. You can now configure Deployment Automation by logging in to the Web application using the administrator account that you created.

If you want to start configuring your system and defer agent installation until later, continue with Chapter 6: Administration Overview [page 105].

If you are ready to install agent relays or agents, continue with the respective section at Agent Relay Installation [page 61] or Agent Installation [page 75].

Manual Server Installation

If you want to install the server on a UNIX platform other than Linux or AIX, you cannot use the interactive or silent installers. Therefore, you will need to install manually using the following procedure.

If you do not need to manually install a server, either continue with one of the other installation options as described in Server Installation [page 44] or continue with the next step in the Installation Checklist [page 39].

To install the server on UNIX platforms for which no interactive installer is provided:

1. Prepare your database. For a typical UNIX/Linux installation, you should use Oracle. See Preparing Your Database [page 43].

   Note: You should use Derby only if this installation is for evaluation or test purposes. Derby is included in the `da.war` file and you configure it as part of the Deployment Automation configuration.

2. Install Tomcat if it is not already installed. If you can run the Common Tomcat installer, you should install that. If not, you must install Tomcat according to the Apache instructions. See http://tomcat.apache.org/.

- You must install the version of Tomcat supported by your version of Deployment Automation. For supported versions, see the Supported Platform List for your version of Deployment Automation on the Support website.

- You should NOT use the root user or super user to own and run Tomcat. Any new files created will be owned by the user running Tomcat, so this should be a user who you want to be authorized to remove or overwrite any files under Tomcat.

- You should set 755 permissions for the Tomcat installation location.

- Before continuing, start Tomcat and verify that it is installed properly by entering its URL in a web browser. For example: `http://localhost:8080`

  As only the owning user has write permission in the Tomcat location, this is the user who should be used to start and stop Tomcat.

3. Download and install the JDBC driver specific to the database edition you are using and save it in the Tomcat `lib` directory. For the Oracle JDBC driver, see http://www.oracle.com/technetwork/indexes/downloads/index.html.

4. Install JRE 8.0 if it is not already installed. See http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html. Set the necessary environment variables for the owning user's session to point to this location, such as adding the JRE paths to the `PATH` environment variable and setting `JAVA_HOME`.

5. From the Support website, download the **Deployment Automation Server WAR Installer** (`MicroFocusDA-Other.zip`) and extract the `da.war` file.

6. Stop Tomcat.

7. Copy and paste the `da.war` file into your Common Tomcat `webapps` folder.

8. Start Tomcat. Starting Tomcat automatically unpacks the `da.war` file.

9. Launch a browser and navigate to http://serverName:8080/da.

   The configuration setup displays.

10. Modify any configuration parameters as needed:

    - Install location

    - External Web URL

    - Agent port and choice of mutual authentication

    - Database (port, schema, user, password)

11. Click **Install**.

12. You should set 755 permissions for the profile directory and its subdirectories for the user that the installation will run under. The default directory structure is: `/opt/MicroFocus/da/microfocus/.microfocus/da`

    Note: The Deployment Automation repository, CodeStation, is under the profile directory structure by default. If you relocate it, you need to set the same level of permissions for the alternate location. See Relocating the Repository [page 178].

When the server installation is complete, the Deployment Automation login page displays.

Silent Mode Server Installation

You can install a Deployment Automation server in silent mode if you want to automate the installation.

This section contains information about how to implement a silent installation of the Deployment Automation server on Windows, Linux, and AIX. For other platforms, you must manually install. See Manual Server Installation [page 48].

If you do not plan to install using silent mode but have not yet installed the server, continue with one of the other installation options as described in Server Installation [page 44]. If you have already installed your server, continue with the next step in the Installation Checklist [page 39].

The following topics lead you through a silent mode server installation.

- (Windows) Server Silent Installation [page 50]
- (UNIX/Linux) Server Silent Installation [page 54]

(Windows) Server Silent Installation

You can install in silent mode as an alternative to using the interactive installer.

To install the server in silent mode:

1. Download the installation files.
2. Create an options file and save as `C:/optionsFile.txt` (see (Windows) Server Silent Installation Options [page 51]).
3. In Windows as a local system user, issue the command:

   ```
   cmd /c MicroFocusDA-Server.exe /s /V"/qn /L*vx "%TEMP%\silent-
   install.log" PROPFILE=\"c:\optionsFile.txt\" "
   ```

   Where:

   - `"%TEMP%\silent-install.log"` is an absolute path to a log file.
   - `PROPFILE=\"c:\optionsFile.txt\"` is an absolute path to the options file you created in Step 2.
   - You have included the **required** space between the final two quotes.

For examples of the `optionsFile.txt`, see (Windows) Server Silent Installation: OptionsFile.txt Examples [page 52]).

(Windows) Server Silent Installation Options

This section provides the list and description of the Windows silent installation options.

For examples of the option settings needed for each database type, see (Windows) Server Silent Installation: OptionsFile.txt Examples [page 52].

Option	Default/if not specified	Description
AgreeToLicense	No	Must be set to Yes
USE_EXISTING_SETTINGS		Set to UseExisting if you want to skip the database installation and administration part of the installation. **Important:** Use only if there are existing Deployment Automation settings populated in the directory specified by: SRA_USER_INSTALLDIR.
SRA_USER_INSTALLDIR	c:\Documents and Settings\→ Administrator\→ .microfocus\da	Directory in which to install the server, or if using the USE_EXISTING_SETTINGS=→ UseExisting option, the directory where your Deployment Automation settings already exist.
SKIP_DB		To skip the database creation, include and set this option: SKIP_DB=SkipDB **Note:** To use this option, the SRA_USER_INSTALLDIR must *not* already exist.
AGENT_MUTUAL_AUTH	N	If using agent mutual authentication, set this to Y. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].
JMS_PORT	7918	Server port
DB_TYPE	derby	Use this to specify a database vendor other than the default: ORA|SQLSVR
<DB>_USER	da	Database user ID, where <DB> is: DERBY|ORA|SQLSVR

Option	Default/if not specified	Description
<DB>_PASSWORD	password	Password for the database user ID, where <DB> is: DERBY|ORA|SQLSVR
ORA_DB_SCHEMA		Required for Oracle, the database schema
DERBY_PORT		Required for Derby, specify the Derby port 11377
<DB>_DB_CONN		Required for databases other than Derby, the database connection, where <DB> is ORA|SQLSVR
IS_INSTALL_MODE		Required; installation mode must be set to silent
SRA_ADMIN		Required installation owner's administration user name **CAUTION:** ⚠ Once created, you cannot remove the built-in administrative user through the user interface. Therefore, you should create this user using your organization's naming standards.
SRA_ADMIN_PWD		Required user-defined password for the user name set by the SRA_ADMIN option
TC_PORT	8080	Common Tomcat port, required only for new installations

(Windows) Server Silent Installation: OptionsFile.txt Examples

This section contains examples of how to configure the installation options in an optionsFile.txt file for a server silent installation on Windows.

Examples are given for Oracle and SQL Server as those are expected to be used in a production environment.

Oracle Database optionsFile.txt example

In the following example, all defaults are used except for the `TC_PORT`, which is set to `8081`.

```
AgreeToLicense=Yes
SRA_USER_INSTALLDIR=C:\Documents and Settings\Administrator\.microfocus\da
SKIP_DB=""
AGENT_MUTUAL_AUTH=""
JMS_PORT=7918
DB_TYPE=oracle
ORA_DB_CONN=jdbc:oracle:thin:@//serverName:1521/da
ORA_DB_SCHEMA=da
ORA_PASSWORD=password123
IS_INSTALL_MODE=silent
SRA_ADMIN=admin1
SRA_ADMIN_PWD=1
TC_PORT=8081
```

MS SQL Server Database optionsFile.txt example

In the following example, all defaults are used except for the `TC_PORT`, which is set to `8088`.

```
AgreeToLicense=Yes
SRA_USER_INSTALLDIR=C:\Documents and Settings\Administrator\.microfocus\da
SKIP_DB=""
AGENT_MUTUAL_AUTH=""
JMS_PORT=7918
DB_TYPE=sqlserver
SQLSVR_DB_CONN=jdbc:sqlserver://serverName:1433;DatabaseName=da
SRA_ADMIN=admin1
SRA_ADMIN_PWD=1
SQLSVR_USER=da
SQLSVR_PASSWORD=mypassword
IS_INSTALL_MODE=silent
TC_PORT=8088
```

MS SQL Database "Skip Database Creation" optionsFile.txt example

In the following example, the `SRA_USER_INSTALLDIR` specified must *not* already exist. Using this `optionsFile.txt` file, because the `SKIP_DB=skipDB` option is specified, the `DB_TYPE` option cannot be set to `derby`, and the `SRA_ADMIN` and `SRA_ADMIN_PWD` options cannot be set during the server installation.

```
AgreeToLicense=Yes
SRA_USER_INSTALLDIR=C:\Documents and Settings\Administrator\.microfocus\da
SKIP_DB=SkipDB
AGENT_MUTUAL_AUTH=""
JMS_PORT=7918
DB_TYPE=sqlserver
SQLSVR_DB_CONN=jdbc:sqlserver://serverName:1433;DatabaseName=da
SQLSVR_USER=da
SQLSVR_PASSWORD=mypassword
```

```
IS_INSTALL_MODE=silent
TC_PORT=8080
```

MS SQL Database "Use Existing Settings" optionsFile.txt example

This is an example of a new installation of the server that points to an existing profile. The Deployment Automation profile location already exists as specified by the `SRA_USER_INSTALLDIR` option. Using this `optionsFile.txt` file, the server will be installed and the existing settings will be used for the installation.

```
AgreeToLicense=Yes
USE_EXISTING_SETTINGS=UseExisting
SRA_USER_INSTALLDIR=C:\Documents and Settings\Administrator\.microfocus\da
SRA_ADMIN=admin1234
SRA_ADMIN_PWD=password123
TC_PORT=8080
```

(UNIX/Linux) Server Silent Installation

The following instructions are for silent installations of the server on platforms for which the server installer is provided, Linux and AIX.

To install the server in silent mode:

1. Download the installation files.
2. Create an `optionsFile.txt` and save it to the root directory. See (UNIX/Linux) Server Silent Installation Options.
3. As a user with root privileges, issue the command:

   ```
   MicroFocusDA-Server.bin -silent -options optionsFile.txt
   ```

   Where:

   `optionsFile.txt` is a file that contains the properties you have set for your system.

For examples of the `optionsFile.txt`, see (UNIX/Linux) Server Silent Installation: optionsFile.txt Examples [page 57].

(UNIX/Linux) Server Silent Installation Options

This section provides the list and description of the UNIX/Linux silent installation options, and examples of the option settings needed for each type of database.

For examples of the option settings needed for each database type, see (UNIX/Linux) Server Silent Installation: optionsFile.txt Examples [page 57].

Option	Default	Description
`-V IS_SELECTED_INSTALLATION_TYPE`	`typical`	Must be set to `typical`; do not enclose in quotes

Option	Default	Description
`-P installLocation`		Specify the directory into which to install the Deployment Automation server. The `-P` property is used by the silent installation tool as the location to place files on disk. This should be set to `/opt/microfocus/da`
`-V ServerDetailsInstallLoc`		Specify the path to the Deployment Automation profile location. The value depends on the user specified in the `-V SctTomcatOwner` option. This should be set to: `/opt/microfocus/da/<SctTomcatOwnerValue>/.microfocus/da`
`-V ServerDetailsUseExisting`	`false`	To skip the database installation and administration part of the installation, include this and set it to `true`. **Important:** Use only if there are existing Deployment Automation settings populated in the directory specified by: `SRA_USER_INSTALLDIR`.
`-V ServerDetailsSkipDb`	`false`	To skip the database creation, include this and set it to `true`.
`-V ServerDetailsPort`	`7918`	Server port
`-V ServerDetailsMutualAuth`	`false`	To use agent mutual authentication, specify this and set it to `true`. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].

Option	Default	Description
`-V SctFoundLoc`		To reuse an existing installation of Common Tomcat, specify this and set it to `true`. When this is set to `true`, you *must* also include the `SctInstallLoc` option.
`-V SctInstallLoc`		(Required) If you specify the `SctFoundLoc` option to reuse an existing installation of Common Tomcat, you must use this option to specify the location of the existing Common Tomcat.
`-V DbDetailsVendor`	`derby`	Use this to specify a database vendor other than the default: `oracle` or `sqlserver`
`-V DbDetailsUser`	`da`	Database user ID
`-V DbDetailsPwd`	`password`	Password for database user ID
`-V DbDetailsDatabaseName`		Database name (SQL Server only)
`-V DbDetailsSchema`		(Required for Oracle) Database schema
`-V dbDetailsDriverFilename`		Required for a database other than Derby; specify a database driver file. **Note:** The first "d" must be lowercase in this parameter only.
`-V DbDetailsDerbyPort`		Required for Derby, this specifies the Derby port, `11377`

Option	Default	Description
`-V DbDetailsConnection`		Database connection
`-V IS_INSTALL_MODE`		Required; installation mode must be set to `silent`
`-V AdminDetailsName`		Required installation owner's administration user name **CAUTION:** ⚠ Once created, you cannot remove the built-in administrative user through the user interface. Therefore, you should create this user using your organization's naming standards.
`-V AdminDetailsPwd`		Required user-defined password for the username set by the `AdminDetailsName` option
`-V SctTomcatOwner`		System user to own the Common Tomcat files, required only for new Common Tomcat installations
`-V SctTomcatPort`	`8080`	Common Tomcat port, required only for new installations. See the `SctFoundLoc` option.

(UNIX/Linux) Server Silent Installation: optionsFile.txt Examples

This section contains examples of how to configure the installation options in an `optionsFile.txt` file for a server silent installation on UNIX/Linux.

Examples are given for Oracle and SQL Server as those are expected to be used in a production environment.

Oracle Database `optionsFile.txt` example

```
-V IS_SELECTED_INSTALLATION_TYPE=typical
-P installLocation="/opt/microfocus/da"
-V ServerDetailsInstallLoc="/opt/microfocus/da/dmsys/.microfocus/da"
```

```
-V DbDetailsVendor="oracle"
-V DbDetailsUser="User02"
-V DbDetailsPwd="MyPassword"
-V DbDetailsSchema="da"
-V DbDetailsConnection="jdbc\:oracle\:thin\:@//serverName\:1521/sraoracle"
-V ServerDetailsMutualAuth="false"
-V IS_INSTALL_MODE="silent"
-V AdminDetailsName="admin123"
-V AdminDetailsPwd="mypassword123"
-V SctTomcatOwner="dmsys"
-V SctTomcatPort="8080"
-V ServerDetailsSkipDb=false
-V ServerDetailsUseExisting=false
-V DbDetailsVendorDerby=false
```

MS SQL Server Database `optionsFile.txt` example

```
-V IS_SELECTED_INSTALLATION_TYPE=typical
-P installLocation="/opt/microfocus/da"
-V ServerDetailsInstallLoc="/opt/microfocus/da/dmsys/.microfocus/da"
-V DbDetailsVendor="sqlserver"
-V DbDetailsUser="User03"
-V DbDetailsPwd="MyPassword"
-V DbDetailsDatabaseName="da"
-V DbDetailsConnection="jdbc\:jtds\:sqlserver\://serverName\:1433/da"
-V ServerDetailsMutualAuth="false"
-V IS_INSTALL_MODE="silent"
-V AdminDetailsName="admin123"
-V AdminDetailsPwd="mypassword123"
-V SctTomcatOwner="dmsys"
-V SctTomcatPort="8080"
-V DbDetailsVendorDerby=false
```

Uninstalling Servers

If you need to uninstall your Deployment Automation server, use the provided uninstall executables as follows.

For Windows

Run the interactive installer for the version of the product that you have installed. This will detect that you have an existing installation. Select the option to uninstall and follow the prompts to complete the uninstall process.

Run the following installer executable as a Windows local system user:

- Windows 64-bit: `MicroFocusDA-Server64.exe`

For UNIX/Linux

Navigate to the `_uninst` folder, for example `/opt/microfocus/da/_uninst`, and run the following uninstall executable as a user with root privileges. Follow the prompts to complete the uninstall process.

```
uninstaller.bin
```

Reconfiguring Server Installation Settings

If you need to reconfigure server settings for any reason, you can reconfigure them using the following procedure. For example, you may need to use this procedure if you specified the wrong connection information when you installed Deployment Automation.

To reconfigure the server settings:

1. Stop the application server where Deployment Automation is installed.
2. Navigate to the directory or folder where the web application is installed. For example:

   `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\conf`

3. Backup the contents of the following file:

   `installed.properties`

4. Edit the `installed.properties` file `installLocation` line to delete the path that follows the equal sign. You should have just the following on that line.

   `installLocation=`

   CAUTION:

   There is another `installed.properties` file in the profile directory, so make sure you are only changing the one in the user interface server directory structure.

5. Save the file.
6. Start the application server.
7. Enter the URL for the Deployment Automation user interface. For example:

   `http://serverName:8080/da`

   The configuration panel should appear instead of the login page.

8. Enter the server settings as follows:

Setting	Description

Install Location	The path to the program installation location. For example:
	(Windows)
	`C:\Users\<username>\.microfocus\da`
	(UNIX/Linux)
	`/opt/microfocus/da`

External Web URL	Specify the URL on which agents will be able to reach the server.

Setting	Description
Agent Port	Specify the port on which agents will make JMS connections to the server.
Agent Mutual Authentication	Specify whether to use agent mutual authentication.
Skip DB Install	Do not install the database schema or seed data as part of the install. Select this if installing to a second or more server in a clustered installation.
Database Vendor	Select the database vendor. For example, Derby. **Note:** For example settings for all supported database selections, see the example settings used in Silent Mode Agent Installation [page 79].
Derby Port	The port that Derby should use for database connections
Database User	The user name to use to connect to the database
Database Password	The password to use to connect to the database
Confirm Database Password	Confirm the password to be used by the Deployment Automation administrator.
Administrator User	The user name to be used by the Deployment Automation administrator
Administrator Password	The password to be used by the Deployment Automation administrator
Confirm Administrator Password	Confirm the password to be used by the Deployment Automation administrator.

9. Click **Install**.

After you have saved the configuration settings, Deployment Automation detects that the configuration file exists and does not prompt you for them when the user interface is invoked.

Agent Relay Installation

Depending on the server configuration your organization needs, you may need to install one or more agent relays. If you are doing a basic installation of Deployment Automation and are not installing agent relays, skip this section and continue on to Agent Installation [page 75].

If you do plan to use agent relays, the agent relays must be installed before you can connect agents to them. You can reconfigure agents to go through agent relays if you change the configuration later. For information on configuring the server and agent communication, see Server and Agent Communication Configuration [page 73].

The following topics lead you through an agent relay installation.

- Interactive Agent Relay Installation [page 61]
- Agent Relay Installation Options [page 62]
- Silent Mode Agent Relay Installation [page 65]
- Upgrading Agent Relays [page 72]
- Server and Agent Communication Configuration [page 73]

Interactive Agent Relay Installation

Installation Prerequisites

Before installing an agent relay, ensure that:

- A supported version of Java is installed. See Agent Minimum Requirements [page 43].
- The server with which the agent relay will connect is already installed.

Important: If you are installing on a 64-bit Linux system, be aware that the UNIX installers are 32-bit applications, and your system may not support 32-bit applications by default. Before running the installer on a 64-bit Linux system, ensure that the following are installed:

- Motif: This is usually available by default
- 32-bit compatibility libraries: C/C++ runtime

To install an agent relay using the interactive installer:

1. Download the agent relay installation package from the Support website.
2. Expand the compressed installation bundle.
3. Ensure that the system environment variable JAVA_HOME is set to the correct path for the Java files. For example: `JAVA_HOME=C:\Program Files\Micro Focus\common\jre\8.0`
4. Ensure that the system environment variable PATH is set to the correct path for the Java executable. For example: `PATH=%PATH%;C:\Program Files\Micro Focus\common\jre\8.0\bin`

5. Open a command prompt and from the expanded `agent-relay-install` directory, run the installation script.

   For Windows, run:

   ```
   install.cmd
   ```

   For UNIX/Linux platforms, run:

   ```
   install.sh
   ```

6. The installation program will prompt you for the Agent Relay Installation Options [page 62]. Accept any default values suggested by the program, displayed within brackets, by pressing **Enter**. If two options are given, such as `Y/n`, the capitalized option is the default value.

After you complete the agent relay installation, continue to Agent Installation [page 75].

Agent Relay Installation Options

When prompted by the installer, specify the options according to the information in the following table. Where Windows and UNIX/Linux options differ, operating system-specific information is included.

Agent Relay Configuration Table

Installation Option	Description

Please enter the directory where you would like to install the agent relay.	Enter the directory where you want the agent relay installed. If it does not exist, you will be prompted on whether to create it. For example:
	Windows
	`C:\Program Files\Micro Focus\agentrelay`
	Unix/Linux
	`/opt/MicroFocus/agentrelay`

Please enter your java home.	Enter the directory where Java is installed. Ensure that the `JAVA_HOME` environment variable points to this directory. For example:
	Windows
	`C:\Program Files\Micro Focus\common\jre\8.0`
	Unix/Linux
	`/opt/MicroFocus/common/jre`
	Important: On Windows, JRE 8.0 is required for an agent relay to be installed as a service. On all platforms, to restrict the security protocols, you must use version Java 8.0.

Installation Option	Description
Enter the name of this relay.	Enter the name of the agent relay. Each agent relay must have a unique name. The default name is `agent-relay`.
Enter the IP or hostname which this agent relay should use	Enter the IP address or hostname on which the agent relay will listen for HTTP requests coming from agents.
Enter the port which this agent relay should proxy HTTP requests on	Enter the port on which the agent relay will listen for HTTP requests coming from agents. The default is `20080`. **Note:** Agent relays can also communicate through HTTPS. To ensure this, you must configure the server to work through HTTPS and change the external agent link to use HTTPS. Agents will then automatically connect through HTTPS. Agent relays will automatically use HTTPS when agents request HTTPS links.
Port which this agent relay should use for communication.	Enter the port on which the agent relay will use for JMS-based communications with remote agents. The default is `7916`.
Connect the agent relay to a central server?	Specify whether you want the agent relay to connect to the Deployment Automation server as follows: • Default: If you plan to use this agent relay in agent-initiated communication, or active mode, enter `Y`. See Agent-initiated HTTP Communication Using Agent Relays [page 74]. • If you plan to use this agent relay in server-initiated communication or passive mode, enter `n`. By doing this, you will have the option to select **Server-Initiated** for the agent relay in the user interface. See Server-initiated HTTP Communication Using Agent Relays [page 74]. **Note:** If you aren't sure, take the default (`Y`). If you want to change it later, you can change this in the `agentrelay.properties` file located in the `relay_installation\conf` directory.
Enter the IP or hostname of your central server	If you selected `Y` in the option to connect the agent relay to the server, enter the IP or host name where the agent relay can contact the server.

Installation Option	Description
Enter the port which the central server uses for communication	If you selected `Y` in the option to connect the agent relay to the server, enter the port the server uses to communicate with agents. The default is `7918`.
Use mutual authentication between the agent, relay and server.	If mutual authentication is required, enter `Y`. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].
Do you want to install the Agent Relay as Windows service?	If you are installing the agent relay on Windows, you can install it as a Windows service. The default value is `N`. If you specify `Y`, see the following table entries.
Enter a unique service name for the Agent Relay. No spaces please.	(Required for running as a Windows service) Enter a unique name for the service. The default is `agentrelay`.
Enter the user account name including domain path to run the service as (for local use '.\.' before login).	(Required for running as a Windows service) Enter an account name, including the domain path under which to run the service. For a local user, insert "`.\`" before the user name. The default is `.\localsystem`
Enter the password for the '.\localsystem' user account.	(Optional for running as a Windows service) Enter a password for the Windows service user account name. The default is no password.
Do you want to start the service automatically?	(Required for running as a Windows service) Specify whether or not (`y|N`) you want the Windows service to be started automatically. The default is `N`.

Note: If you need to change the agent relay configuration, you can edit these properties in the `agentrelay.properties` file located in the `<relay_installation>\conf` directory. You can also do some agent relay management through the user interface. See Managing Agent Relays [page 316].

Silent Mode Agent Relay Installation

This section contains information about how to configure and run a silent installation of a Deployment Automation agent relay on Windows and UNIX/Linux platforms.

Important: Before you can run a silent install of an agent relay, you *must* create and save an `optionsFile.properties`.

The following topics lead you through a silent mode agent relay installation.

- (Windows) Agent Relay Silent Installation [page 65]
- (UNIX/Linux) Agent Relay Silent Installation [page 69]

(Windows) Agent Relay Silent Installation

To install an agent relay in silent mode:

1. Download the appropriate agent relay installer for your platform.
2. Create an options file and save as:

   `C:/optionsFile.properties`

   Note: Your optionsFile should be in ANSI format; the agent relay installer does not support UTF.

3. Issue the command:

   ```
   install.cmd -silent -options optionsFile.properties
   ```

   where

   `optionsFile.properties`

   is the options file you created in Step 2.

(Windows) Agent Relay Silent Install Options

This section provides the list and description of the Windows agent relay silent install options.

For an example of the option settings, see (Windows) Agent Relay Silent Install Options Example [page 68].

Installation Option	Description
AGENT_RELAY_HOME	Enter the directory where you want the agent relay installed. If it does not exist, you will be prompted on whether to create it. For example: `C:\Program Files\Micro Focus\agentrelay` **Note:** This is the only option that is required to run the agent relay silent installation. If only this property is included, the rest of the options will have the default values.
JAVA_HOME	Enter the directory where Java is installed. Ensure that the `JAVA_HOME` environment variable points to this directory. For example: `C:\Program Files\Micro Focus\common\jre\8.0`
AGENT_RELAY_NAME	Enter the name of the agent relay. Each agent relay must have a unique name. The default name is `agent-relay`.
PROXY_HOST	Enter the IP address or hostname on which the agent relay will listen for HTTP requests coming from agents.
PROXY_PORT	Enter the port on which the agent relay will listen for HTTP requests coming from agents. The default is `20080`. **Note:** Agent relays can also communicate through HTTPS. To ensure this, you must configure the server to work through HTTPS and change the external agent link to use HTTPS. Agents will then automatically connect through HTTPS. Agent relays will automatically use HTTPS when agents request HTTPS links.
AGENT_RELAY_RELAY_PORT	Enter the port on which the agent relay will use for JMS-based communications with remote agents. The default is `7916`.
AGENT_RELAY_MUTUAL_AUTH	If mutual authentication is required, enter `true`. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].

Installation Option	Description
AGENT_RELAY_CONNECT	Specify whether you want the agent relay to connect to the Deployment Automation server as follows: • Default: If you plan to use this agent relay in agent-initiated communication, or active mode, enter `true`. See Agent-initiated HTTP Communication Using Agent Relays [page 74]. • If you plan to use this agent relay in server-initiated communication or passive mode, enter `false`. By doing this, you will have the option to select **Server-Initiated** for the agent relay in the user interface. See Server-initiated HTTP Communication Using Agent Relays [page 74]. **Note:** If you aren't sure, take the default (`true`). If you want to change it later, you can change this in the `agentrelay.properties` file located in the `relay_installation\conf` directory.
AGENT_RELAY_SERVER_HOST	If you entered `true` for AGENT_RELAY_CONNECT, enter the IP or host name where the agent relay can contact the server.
AGENT_RELAY_SERVER_PORT	If you entered `true` for AGENT_RELAY_CONNECT, enter the port the server uses to communicate with agents. The default is `7918`.
INSTALL_SERVICE	If you are installing the agent relay on Windows, you can install it as a Windows service. The default value is `true`. If you specify `true`, see the following table entries.
INSTALL_SERVICE_NAME	(Required for running as a Windows service) Enter a unique name for the service. No spaces are allowed. The default is `agentrelay`.
INSTALL_SERVICE_LOGIN	(Required for running as a Windows service) Enter an account name, including the domain path under which to run the service. For a local user, insert "`.\`" before the user name. The default is `.\localsystem`
INSTALL_SERVICE_PASSWORD	(Optional for running as a Windows service) Enter a password for the Windows service user account name. The default is `nopass`.

Installation Option	Description
INSTALL_SERVICE_AUTOSTART	(Required for running as a Windows service) Specify whether or not (`true/false`) you want the Windows service to be started automatically. The default is `false`.

Note: If you need to change the agent relay configuration after the installation, you can edit these properties in the `agentrelay.properties` file located in the `<relay_installation>\conf` directory. You can also do some agent relay management through the user interface. See Managing Agent Relays [page 316].

(Windows) Agent Relay Silent Install Options Example

This section contains an example of how to configure the install options in an `optionsFile.properties` file for an agent relay silent install on Windows.

Example: Agent Relay Windows optionsFile.properties

Using this `optionsFile.properties`, the agent relay will be named `Relay123` and will be configured to connect to the server on `SDAServerHost`. Mutual Authentication mode will be activated for the agent relay, and the agent relay will run as a Windows service under the name `sda-relay123` that will start automatically when the system starts. The agent relay Windows service admin user account name will be set to: `admin01` with a password of `password`.

```
AGENT_RELAY_HOME=C:\Program Files\Micro Focus\agentrelay
JAVA_HOME=C:\Program Files\Micro Focus\common\jre\8.0
AGENT_RELAY_NAME=Relay123
PROXY_HOST=0.0.0.0
PROXY_PORT=20080
AGENT_RELAY_RELAY_PORT=7916
AGENT_RELAY_MUTUAL_AUTH=true
AGENT_RELAY_CONNECT=true
AGENT_RELAY_SERVER_HOST=DAServerHost
AGENT_RELAY_SERVER_PORT=7918
INSTALL_SERVICE=true
INSTALL_SERVICE_NAME=sda-relay123
INSTALL_SERVICE_LOGIN=admin01
INSTALL_SERVICE_PASSWORD=password
INSTALL_SERVICE_AUTOSTART=true
```

(UNIX/Linux) Agent Relay Silent Installation

The following instructions are for silent installations of agent relays.

Important: If you are installing on a 64-bit Linux system, be aware that the UNIX installers are 32-bit applications, and your system may not support 32-bit applications by default. Before running the installer on a 64-bit Linux system, ensure that the following are installed:

- Motif: This is usually available by default
- 32-bit compatibility libraries: C/C++ runtime

To install an agent relay in silent mode:

1. Download the appropriate agent relay installer for your platform.
2. Create an options file and save as:

   ```
   optionsFile.properties
   ```

   Note: Your `optionsFile` should be in ANSI format; the agent relay installer does not support UTF.

3. Issue the command:

   ```
   install.sh -silent -options optionsFile.properties
   ```

   where

   ```
   optionsFile.properties
   ```

   is the options file you created in Step 2.

(UNIX/Linux) Agent Relay Silent Install Options

This section provides the list and description of the UNIX/Linux agent relay silent install options.

For an example of the option settings, see (UNIX/Linux) Agent Relay Silent Install Options Example [page 71].

Installation Option	Description
AGENT_RELAY_HOME	Enter the directory where you want the agent relay installed. If it does not exist, you will be prompted on whether to create it. For example: `/opt/MicroFocus/agentrelay` **Note:** This is the only option that is required to run the agent relay silent installation. If only this property is included, the rest of the options will have the default values.

Installation Option	Description
JAVA_HOME	Enter the directory where Java is installed. Ensure that the `JAVA_HOME` environment variable points to this directory. For example: `/opt/MicroFocus/common/jre`
AGENT_RELAY_NAME	Enter the name of the agent relay. Each agent relay must have a unique name. The default name is `agent-relay`.
PROXY_HOST	Enter the IP address or hostname on which the agent relay will listen for HTTP requests coming from agents.
PROXY_PORT	Enter the port on which the agent relay will listen for HTTP requests coming from agents. The default is `20080`. **Note:** Agent relays can also communicate through HTTPS. To ensure this, you must configure the server to work through HTTPS and change the external agent link to use HTTPS. Agents will then automatically connect through HTTPS. Agent relays will automatically use HTTPS when agents request HTTPS links.
AGENT_RELAY_RELAY_PORT	Enter the port on which the agent relay will use for JMS-based communications with remote agents. The default is `7916`.
AGENT_RELAY_MUTUAL_AUTH	If mutual authentication is required, enter `true`. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].

Installation Option	Description
AGENT_RELAY_CONNECT	Specify whether you want the agent relay to connect to the Deployment Automation server as follows: • Default: If you plan to use this agent relay in agent-initiated communication, or active mode, enter `true`. See Agent-initiated HTTP Communication Using Agent Relays [page 74]. • If you plan to use this agent relay in server-initiated communication or passive mode, enter `false`. By doing this, you will have the option to select **Server-Initiated** for the agent relay in the user interface. See Server-initiated HTTP Communication Using Agent Relays [page 74]. **Note:** If you aren't sure, take the default (`true`). If you want to change it later, you can change this in the `agentrelay.properties` file located in the `<relay_installation>\conf` directory.
AGENT_RELAY_SERVER_HOST	If you entered `true` for AGENT_RELAY_CONNECT, enter the IP or host name where the agent relay can contact the server.
AGENT_RELAY_SERVER_PORT	If you entered `true` for AGENT_RELAY_CONNECT, enter the port the server uses to communicate with agents. The default is `7918`.
AGENTRELAY_USER	UNIX/Linux user name under which to run the agent relay.
AGENTRELAY_GROUP	UNIX/Linux group name under which to run the agent relay.

Note: If you need to change the agent relay configuration after the installation, you can edit these properties in the `agentrelay.properties` file located in the `<relay_installation>\conf` directory. You can also do some agent relay management through the user interface. See Managing Agent Relays [page 316].

(UNIX/Linux) Agent Relay Silent Install Options Example

This section contains an example of how to configure the install options in an `optionsFile.properties` file for an agent relay silent install on UNIX/Linux.

Example: Agent Relay UNIX/Linux optionsFile.properties

Using this `optionsFile.properties`, the agent relay will be named `Relay123` and will be configured to connect to the server on `SDAServerHost`. Mutual Authentication mode will be activated for the agent relay.

```
AGENT_RELAY_HOME=/opt/MicroFocus/agentrelay
JAVA_HOME=/opt/MicroFocus/common/jre
AGENT_RELAY_NAME=Relay123
PROXY_HOST=0.0.0.0
PROXY_PORT=20080
AGENT_RELAY_RELAY_PORT=7916
AGENT_RELAY_MUTUAL_AUTH=true
AGENT_RELAY_CONNECT=true
AGENT_RELAY_SERVER_HOST=DAServerHost
AGENT_RELAY_SERVER_PORT=7918
AGENTRELAY_USER=agentrelay
AGENTRELAY_GROUP=agentrelay
```

Upgrading Agent Relays

When upgrading agent relays, make sure to follow the following recommended upgrade order for agent relays and servers.

Upgrade Order for Agent Relays and Servers

1. Shut down all agent relays.
2. Shut down all servers to which the agent relays you are upgrading are pointing.
3. Upgrade all the agent relays according to the instructions in the following section.
4. If upgrading the server, upgrade the server using the server installer. See the Release Notes **Upgrades** section for information on versions from which you can upgrade directly.

   Note: Agent relays for version 5.1.5 and later that are configured for server-initiated connection are compatible only with Deployment Automation servers version 5.1.5 and later. You cannot connect to version 5.1.5 and later agent relays through network relays as was possible in earlier server versions. If you attempt to do this, the agents connected through the agent relay will not be in Online status.
5. Start the server.
6. Start the agent relays.

Agent Relay Upgrade Instructions

To upgrade your agent relays:

1. Ensure the agent relay is stopped.
2. Run the agent relay installer. See Agent Relay Installation [page 61].
3. When prompted for the destination directory, enter the path to the directory of the existing agent relay location.

4. When prompted to upgrade, enter `Y`.

After upgrading your agent relays and starting the servers and agent relays, navigate to `Management > Resources > Relays`. You should see all your upgraded agent relays listed. It may take some time before the agent relays come online.

See Managing Agent Relays [page 316] for information on configuring agent relays.

Server and Agent Communication Configuration

Servers and agents communicate either directly or through an agent relay. If an agent relay is used, the agent connects to the server through the agent relay, and the agent relay manages the requests to the server.

If your agents are dispersed across geographic locations or must communicate through firewalls, you should configure the agents to connect to the server through agent relays. You should also use agent relays to manage network traffic in implementations where you have many agents, even if they are on the same network and don't necessarily have to go through firewalls.

Once a JMS connection is established between the server and the agent, regardless of whether an agent relay is used, that connection is kept open as long as the entities involved are online. The server provides HTTP endpoints that the agents trigger to perform service operations. The most common HTTP operations are as follows. All of these operations are initiated by agents by default.

- Download plugins
- Check plugin versions
- Upload log steps
- Download files
- Upload files

You can configure agent-server communication as follows:

- **Agent-to-Server:** To configure your agents to communicate directly with the server, when installing the agents, point the agents to the server.
- **Agent-initiated with Agent Relay:** To configure your agents and servers for agent-initiated HTTP communication through agent relays, do the following:
  - When installing and configuring the agent relays, enter `Y` when prompted for whether to connect to the server. This is the default setting. When a process is running, the agent relay will connect to the server.
  - When installing the agents, point the agents to the agent relay instead of directly to the server.
- **Server-initiated with Agent Relay:** To configure your agents and servers for server-initiated HTTP communication, do the following:
  - When installing and configuring the agent relays, enter `n` when prompted for whether to connect to the server. When a process is running, the server will connect to the agent relay.

- When installing the agents, point the agents to the agent relay instead of directly to the server.

Note: You can configure agents to authenticate using PKI certificates for greater security. See Configuring PKI Certification for Agents [page 136].

The types of communication through agent relays are described in the following topics:

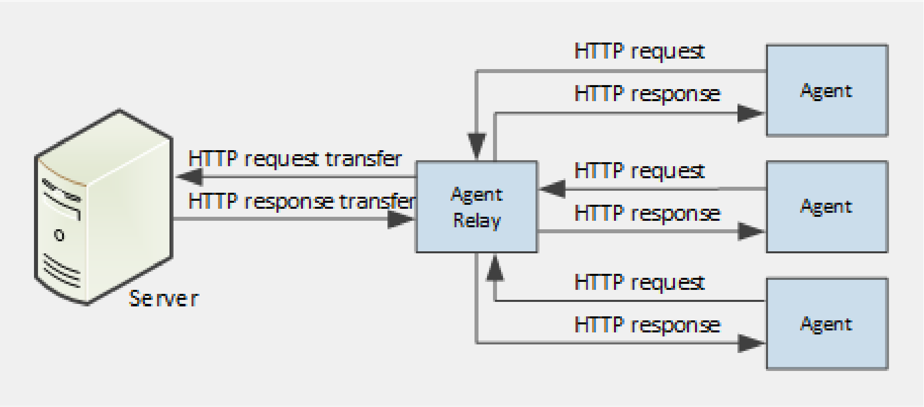
- Agent-initiated HTTP Communication Using Agent Relays [page 74]
- Server-initiated HTTP Communication Using Agent Relays [page 74]

Agent-initiated HTTP Communication Using Agent Relays

You can configure agents and agent relays so that all HTTP communication between the agent and server is initiated by the agents.

Agent-initiated HTTP communication using agent relays is shown in the following figure.

Agent-initiated HTTP Communication



Tip: The agent relay installation option that controls this is **Connect the agent relay to a central server**, which should be set to `Y` for this scenario.

Server-initiated HTTP Communication Using Agent Relays

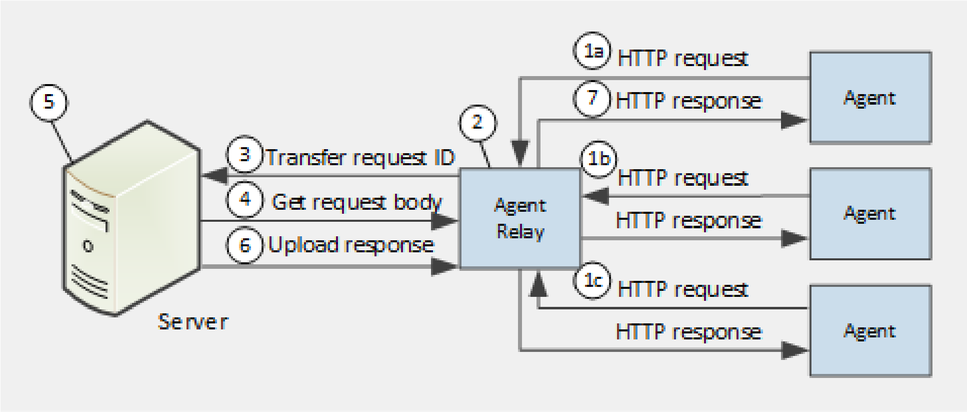
You can configure agents and agent relays so that the server initiates the upload of HTTP requests over a designated size. This enables the server to control the traffic across the network so that the volume of incoming requests from many agents does not overload the network. The agent still initiates the HTTP request as part of the process being executed, and a certain amount of information is passed through to the server on every request. However, the agent relay suspends any HTTP requests that are too large to pass through directly and enables the server to process them when it is ready.

Note:

- This type of communication is sometimes referred to as unidirectional communication, and the agent relay is referred to as passive, because the server is initiating and regulating the HTTP communication.
- Only the inbound JMS port needs to be opened and the server initiates all communications over the JMS tunnel.

Server-initiated HTTP communication using agent relays is shown in the following figure.

Server-initiated HTTP Communication



1. Agents send concurrent requests through HTTP. Requests 1a, 1b and 1c will be processed concurrently, but the figure shows only the processing of request 1a.
2. Every request coming to an agent relay from an agent will be put on hold before the response from the server is provided. All the requests coming from agents will be queued on an agent relay. The agent relay processes the requests and assigns an internal ID to each of them.
3. The server periodically polls the agent relay, and detects that the agent relay has an incoming request. The agent relay sends the request ID information to the server using JMS.
4. When the server receives the request ID, it sends an HTTP request to get the request body from the agent relay.

   Depending on the size of the initial request, this step may be omitted. For small requests, the body can be transferred using JMS during the previous step.
5. After the server receives the body of the request, it processes it as needed.
6. After processing the request, the server sends the response to the agent relay.
7. The agent relay returns the response to the agent from the agent relay using HTTP.

Tip: The agent relay installation option that controls this is **Connect the agent relay to a central server**, which should be set to `n` for this scenario.

Agent Installation

You must install at least one agent for a basic installation. You can install agents on supported operating systems using an interactive installer, a silent installer, console installation mode, or manual installation steps.

CAUTION:

Windows: the agent installer uses a standard distribution technique of the JRE that redistributes the Java runtime files in accordance with Java standards. This creates a private JRE instance that will not interfere with other Java applications that are installed. However, if you create references to the Java files that are installed by the agent, or if you interfere with the agent's private installation directory, you may corrupt other existing Java applications on the host machine.

Note: Solaris: download and install a Solaris JRE 8 before running the agent installer. To run the installer, specify the path to the JRE: `SerenaDA-agent.bin -is:javahome <path to local JRE>`

Multiple Agents on Multiple Machines

If you plan to run deployments on several machines, a separate agent with a unique name should be installed on each machine. If, for example, your testing environment consists of three machines, install an agent on each one. Follow the same procedure for each environment the application uses.

Multiple Agents on One Machine

If you plan to install multiple agents on a single machine, make sure that all names are unique: installation directory name, agent name, and service name, if running the agent as a service.

Installing Agents Remotely

For UNIX environments, you can install agents remotely through the user interface. See Installing Agents Remotely [page 311].

Upgrading Agents

Agents are upgraded through the user interface. See Upgrading Agents [page 314].

The following topics lead you through an agent installation.

- Interactive Agent Installation [page 76]
- Silent Mode Agent Installation [page 79]
- Uninstalling Agents [page 85]
- About Agent Upgrades [page 86]
- Installing and Configuring the z/OS Agent [page 86]

Interactive Agent Installation

The interactive agent installer uses a wizard that guides you through the installation process.

For Windows

Run the following installer executable as a Windows local system user:

```
DA-agent.exe
```

For UNIX/Linux

Important: If you are installing on a 64-bit Linux system, be aware that the UNIX installers are 32-bit applications, and your system may not support 32-bit applications by default. Before running the installer on a 64-bit Linux system, ensure that the following are installed:

- Motif: This is usually available by default
- 32-bit compatibility libraries: C/C++ runtime

Run the following installer executable as a user with full access rights to the installation location:

```
DA-agent.bin
```

Note:

- The interactive agent installer is available for Linux, AIX, Solaris SPARC and x86, and HPUX Itanium.
- To run the interactive installer in console mode, use:

  ```
  DA-agent.bin -console
  ```

For Solaris

Run the following installer executable as a user with full access rights to the installation location:

```
SerenaDA-agent.bin -is:javahome <path to local JRE>
```

Specify the installation information as follows.

Agent Installation Options

In the agent installation wizard panels, specify the options according to the information in the following table. Where Windows and UNIX/Linux options differ, operating system-specific information is included.

Option	Description

Install Deployment Automation Agent to	Specify the location into which the agent will be installed. For example:
	Windows:
	`C:\Program Files (x86)\Micro Focus\Deployment Automation Agent`
	UNIX/Linux:
	`/opt/Micro \ Focus/Deployment Automation Agent`

Agent Name	Specify the name of the agent. The agent name should be unique.

Option	Description
Server Agent Communication to use Mutual Authentication with SSL	Select this if mutual authentication will be used with the server. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].
Will the agent connect to an agent relay instead of directly to the server?	Select `Yes` to use an agent relay. Default: `No`
Hostname or address	If an agent relay is being used, specify the hostname of the agent relay. Default: `localhost`
Agent Relay Communication Port	If an agent relay is being used, specify the port number that will be used for JMS communication with the agent relay. Default: `7916`
HTTP Proxy Port	If an agent relay is being used, specify the proxy port number of the agent relay for HTTP communication. Default: `20080`
Hostname or address	If not using an agent relay, specify the hostname of the Deployment Automation server.
Agent Communication Port	If not using an agent relay, specify the port number that will be used for JMS communication with the Deployment Automation server. Default: `7918`
Do you want to install the Agent as Windows service?	For Windows installations, select `Yes` to create a Windows service for the agent.
Unique service name for the Agent	If installing as a Windows service, specify the name to use for the service. Default: `DA-Agent`
Start the Agent service automatically	If installing as a Windows service, select this to configure the Windows service to start automatically. **Note:** The agent must be manually started by default, so you must select this if you want it to start automatically.
User account name for the service	If installing as a Windows service, specify the log on user account name for the service. Default: `./localsystem`
User account password for the service	If installing as a Windows service, specify the log on user account password for the Windows service. This is not required if you are using the local system account.

Option	Description
Install JRE	Select this to install a new JRE with the agent. **Note:** If you are installing an HPUX agent, this option is not available. You must download and install the HPUX JRE yourself before installing the agent.
Use Existing JRE	Select this to use an existing JRE with the agent.

Please specify the JAVA_HOME for the JRE	If using an existing JRE, specify the location of a pre-existing JRE for the agent to use. For example, on Windows:
	`C:\Program Files\Micro Focus\common\jre\8.0`
	On UNIX/Linux:
	`/opt/MicroFocus/common/jre`
	On Solaris you must specify an existing JRE.

Silent Mode Agent Installation

This section contains information about how to configure and run a silent installation of a Deployment Automation agent on Windows and UNIX/Linux platforms.

Important: Before you can run a silent install of an agent, you *must* create and save an `optionsFile.txt`.

The following topics lead you through a silent mode agent installation.

- (Windows) Agent Silent Installation [page 79]
- (UNIX/Linux) Agent Silent Installation [page 82]

(Windows) Agent Silent Installation

To install an agent in silent mode:

1. Download the appropriate agent installer for your platform.
2. Create an options file and save as:

   ```
   C:/optionsFile.properties
   ```

   Note: Your optionsFile should be in ANSI format; the agent installer does not support UTF.

3. Issue the command:

   ```
   c:\agent-relay-install\install.cmd -silent -options
   optionsFile.properties
   ```

   where

`optionsFile.properties`

is the options file you created in Step 2.

(Windows) Agent Silent Installation Options

This section provides the list and description of the Windows agent silent installation options.

For examples of the option settings, see (Windows) Agent Silent Installation Options Example [page 81].

Option	Description
`-P installLocation`	(Required) The location into which the agent will be installed
`-V IS_DESTINATION`	(Required) Location into which the agent will be installed
`-V AgentName`	(Required) Name of the agent. You can also specify properties that are resolved dynamically on the server. For more information see Dynamically Registering Agents [page 314].
`-V UseRelayYes`	(Required) To use an agent relay set this to `true`, if not, set it to `false`.
`-V RelayHost`	(Required only if `-V UseRelayYes=true` is specified) The hostname of the agent relay to be used
`-V RelayPort`	(Required only if `-V UseRelayYes=true` is specified) The port number of the agent relay to be used
`-V RelayProxyPort`	(Required only if `-V UseRelayYes=true` is specified) The proxy port number of the agent relay
`-V ServerHost`	(Required if `-V UseRelayYes=false`) The hostname of the Deployment Automation server
`-V ServerPort`	(Required if `-V UseRelayYes=false`) The port number of the Deployment Automation server

Option	Description
`-V ServerMutAuth`	(Required) If mutual authentication will be used with the server, set this to `true`; otherwise, set it to `false`. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].
`-V JreNew`	(Required) To install a new JRE with the agent, set this to `true`, otherwise set it to `false`.
`-V JreInstallLoc`	(Required if `-V JreNew=false`) Use this to specify the location of a pre-existing JRE for the agent to use.
`-V ServiceYes`	(Windows Required) To create a Windows service for the agent, set this to `true`; otherwise, set it to `false`. This option is valid only for Windows systems.
Note: The following fields are only required when you specify `-V ServiceYes=True` and only apply to Windows systems.	
`-V ServiceName`	Required only when `-V ServiceYes=True` is specified. The name to use for the Windows service.
`-V ServiceStartAuto`	Required only when `-V ServiceYes=True` is specified. To configure the Windows service to start automatically, set to `true`, if not set to `false`.
`-V ServiceAccName`	Required only when `-V ServiceYes=True` is specified. To specify the log on user account name for the Windows service.
`-V ServiceAccPass`	Required only when `-V ServiceYes=true` is specified. To specify the log on user account password for the Windows service.

(Windows) Agent Silent Installation Options Example

This section contains an example of how to configure the installation options in an `optionsFile.txt` file for an agent silent installation on Windows.

Example: Agent Windows optionsFile.txt

Using this `optionsFile.txt`, the agent will be named `agent123`, be configured to connect directly to the server on `serverabc` (not through an agent relay), and will point to a new installation of a JRE. Mutual Authentication mode will be activated for the agent, and the agent will run as a Windows service under the name `SRA-Agent123` that will start automatically when the system starts. The agent Windows service admin user account name will be set to: `admin01`, with the password: `password`.

```
-P installLocation="C:\Program Files (x86)\Micro Focus\Deployment Automation Agent"
-V IS_DESTINATION="C:\Program Files (x86)\Micro Focus\Deployment Automation Agent"
-V AgentName="agent123"
-V UseRelayYes="false"
-V ServerHost="serverabc"
-V ServerPort="7918"
-V ServerMutAuth="true"
-V ServiceYes="true"
-V ServiceName="da-agent123"
-V ServiceStartAuto="true"
-V ServiceAccName="admin01"
-V ServiceAccPass="password"
-V JreNew="true"
```

(UNIX/Linux) Agent Silent Installation

The following instructions are for silent installations of agents on platforms for which the agent installer is provided. These include Linux, AIX, Solaris SPARC, and HPUX Itanium platforms.

Important: If you are installing on a 64-bit Linux system, be aware that the UNIX installers are 32-bit applications, and your system may not support 32-bit applications by default. Before running the installer on a 64-bit Linux system, ensure that the following are installed:

- Motif: This is usually available by default
- 32-bit compatibility libraries: C/C++ runtime

To install an agent in silent mode:

1. Download the appropriate agent installer for your platform.
2. Create an options file and save as:

   ```
   optionsFile.txt
   ```

   Note: Your optionsFile should be in ANSI format; the agent installer does not support UTF.

3. Issue the command:

   ```
   DA-agent.bin –silent –options optionsFile.txt
   ```

   where

   ```
   optionsFile.txt
   ```

   is the options file you created in Step 2.

(UNIX/Linux) Agent Silent Installation Options

This section provides the list and description of the UNIX/Linux silent installation options, and examples of the option settings.

For examples of the option settings, see (UNIX/Linux) Server Silent Installation: optionsFile.txt Examples [page 57].

Option	Description
`-P installLocation`	(Required) Location into which the agent will be installed.
`-V IS_DESTINATION`	(Required) Location into which the agent will be installed.
`-V AgentName`	(Required) Name of the agent. You can also specify properties that are resolved dynamically on the server. For more information see Dynamically Registering Agents [page 314].
`-V UseRelayYes`	(Required) To use an agent relay set to `true`, if not, set to `false`.
`-V RelayHost`	(Required only if `-V UseRelayYes="true"` is specified.) The hostname of the agent relay to be used.
`-V RelayPort`	(Required only if `-V UseRelayYes="true"` is specified.) The port number of the agent relay to be used.
`-V RelayProxyPort`	(Required only if `-V UseRelayYes="true"` is specified.) The proxy port number of the agent relay.
`-V ServerHost`	(Required if `-V RelayPortYes="false"`) The hostame of the Deployment Automation server.
`-V ServerPort`	(Required if `-V UseRelayYes="false"`) The port number of the Deployment Automation server.
`-V ServerMutAuth`	(Required) If mutual authentication will be used with the server set to `true`, otherwise set to `false`. For more information, see Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].

Option	Description
`-V JreNew`	(Required) To install a new JRE with the agent, set to `true`, if not then set to `false`. **Note:** If you are installing an HPUX or Solaris agent, this option must be set to `false`. You must download and install the HPUX or Solaris JRE yourself before installing the agent.
`-V JreInstallLoc`	(Required if `-V JreNew="false"`) To specify the location of a pre-existing JRE for the agent to use.
`-V ServiceYes`	(Windows Required) To create a Windows service for the agent, set to `true`, otherwise set to `false`. This option is only valid for Windows systems.

(UNIX/Linux) Agent Silent Installation Options Examples

This section contain examples of `optionsFile.txt` file configurations for an agent silent installation on a Linux or a UNIX platform.

Example 1: Agent UNIX/Linux platforms optionsFile.txt

Using this `optionsFile.txt`, during installation the agent will be configured to connect directly with the server on `serverabc` (not through an agent relay), will turn mutual authentication mode on for the agent and install a new JRE.

```
-P installLocation="/opt/Micro \ Focus/Deployment Automation Agent"
-V IS_DESTINATION="/opt/Micro \ Focus/Deployment Automation Agent"
-V AgentName="agent123"
-V UseRelayYes="false"
-V ServerHost=serverabc
-V ServerPort="7918"
-V ServerMutAuth="true"
-V JreNew="true"
```

Example 2: Agent UNIX/Linux platforms optionsFile.txt

Using this `optionsFile.txt`, during installation the agent will be configured to use an agent relay named `relayagent01` on port `7916` to connect to the server. Mutual Authentication for the agent will be turned on, and the agent will be configured to point to and use the JRE that currently exists in: `/opt/Java/`.

```
-P installLocation="/opt/Micro \ Focus/Deployment Automation Agent"
-V IS_DESTINATION="/opt/Micro \ Focus/Deployment Automation Agent"
-V AgentName="agentABC"
-V UseRelayYes="true"
-V RelayHost="relayagent01"
-V RelayPort="7916"
```

```
-V ServerMutAuth="true"
-V JreNew="false"
-V JreInstallLoc="/opt/Java/"
```

Uninstalling Agents

You can inactivate agents and delete them from servers through the web application. However, if you want to completely uninstall the agent files from an agent machine, you can uninstall as follows.

For Windows

Run the following interactive uninstaller executable in the installation path `_uninst` subdirectory as a Windows local system user. For example:

```
C:\Program Files (x86)\Micro Focus\Deployment Automation
Agent\_uninst\uninstaller.exe
```

For UNIX/Linux

The uninstall process for UNIX/Linux agents is manual.

To uninstall a UNIX/Linux agent:

1. Stop and kill all agent processes as follows:
   a. Stop the agent:
      i. Navigate to the installation directory core/bin subdirectory. For example:

         ```
         cd /opt/Micro \ Focus/Deployment \ Automation \ Agent/core/bin
         ```

      ii. Source the agent service executable using the stop parameter. For example:

         ```
         ./sraagent stop
         ```

   b. Find the process IDs and kill them. Typically there are two processes, although the previous step should have stopped them.
      i. Enter a command to list any processes of the name `Deployment Automation Agent`. For example:

         ```
         ps -ef | grep "Deployment Automation Agent"
         ```

      ii. If you find any processes running for `Deployment Automation Agent`, enter the command to kill each process using the corresponding process ID.

         ```
         kill <pid>
         ```

2. Once the agent processes are stopped and killed, remove the following directories:
   a. Installation files: `/opt/Micro \ Focus/Deployment Automation Agent`
   b. Inventory files (if found): `/tmp/microfocus/var/`
   c. Temp install files (if found): `/tmp/da_agent_install`

About Agent Upgrades

After upgrading to a new version of the server or receiving notice of available agent upgrades, you can upgrade your agents from the Deployment Automation user interface.

The agent installer does not upgrade agents. It lets you install a different version of the agent on the same machine that runs in parallel with any existing agents. When installing a second agent on the same machine, you must specify a unique port number, as the installer does not check to see if the port number is already in use by another agent.

For details on upgrading the server, see the Release Notes **Upgrades** tab. The Release Notes are posted on the Documentation Center.

For details on upgrading the agent, see Upgrading Agents [page 314].

Note: You cannot upgrade the z/OS agent from Deployment Automation.

Installing and Configuring the z/OS Agent

z/OS mainframe system requirements:

- z/OS Java Virtual Machine 86 (JVMLDM86)
- Java 8 64-bit

During the installation you will need the names of:

- The PDS/E load library where JVMLDM86 is installed.
- The UNIX directory where Java 64-bit is installed.

Note: `sudo`, `ssh`, and `su` impersonation is not available with the z/OS agent.

To install the z/OS Deployment Automation agent:

1. Download the z/OS agent archive, for example: `zos-agent-6.3.0.243.24.zip`
2. Unzip the downloaded file. This should create a folder called: `zos-agent-install`
3. Create the z/OS mainframe PDS dataset where the DA agent JCL procedure and batch job will be deployed, for example: `MF.DAAGENT.JCLLIB`
4. Edit the file `zos.properties` and set values for the following z/OS installation properties:

   z/OS DA Agent environment properties

   zos.agent.name: agent name, for example: *ZOS_DA-Agent*

   zos.agent.home.dir: home directory where the agent will be installed, for example: */MicroFocus/DA-Agent*

   zos.java.home.dir: Java home directory where Java 8 64-bit is installed.

   da.server.hostname: Micro Focus Deployment Automation server host name.

   zos.java.debug.opts: optional Java virtual machine debug options, for example: *"-Xdebug -*

Xrunjdwp:transport=dt_socket,address=localhost:10001,server=y,suspend=n -Dcom.sun.management.jmxremote"

zos.java.opts: optional Java virtual machine debug options, for example: *"-Xmx512m"*

z/OS DA Agent encoding properties

zos.agent.stdout.encoding: z/OS encoding used for plugin steps output, for example: *Cp1047*

zos.agent.console.encoding: z/OS console encoding, for example: *Cp1047*

zos.agent.file.encoding: z/OS file encoding, for example: *Cp1047*

z/OS Connection properties

zos.hostname: host name where the DA agent will be installed.

zos.ftp.port: port used for FTP on z/OS.

zos.ftp.userid: user that will be used to install the DA agent into z/OS.

zos.ftp.pwd: user password.

z/OS DA Agent Batch Job or Started Task properties

zos.da.jcllib: the mainframe PDS dataset where the batch or started task procedure and batch job will be deployed, for example: *CHGMAN.B.CMN.DAAGENT.JCLLIB*

zos.java.loadlib: the mainframe PDS dataset load library where the Java Virtual Machine module is installed, for example: *CHGMAN.B.CMN.JZOS.LOAD*

zos.taskname: the DA agent task name to use.

zos.jobname: the DA agent batch job name to use when the batch job is submitted.

zos.jobclass: the system job class assigned to the DA agent batch job. The class will determine the dispatching priority of the batch job.

zos.jobowner: the job owner of the DA agent batch job.

zos.java.use.lpa: specify whether to run JZOS in LPA (true/false).

5. Use the following command to deploy the z/OS agent to DA:

   UNIX/Linux: `deploy-agent.sh`

   Windows: `deploy-agent.bat`

6. Choose one of the following install options:

   installAgent: configure the DA agent properties based on information specified in *zos.properties.* Deploys the DA agent to z/OS.

   uninstallAgent: remove the DA agent from z/OS based on information specified in *zos.properties*.

   reinstallAgent: perform *uninstallAgent* and then *installAgent* (as described above).

7. To start the agent run this z/OS command: `/S zos.taskname`

   To stop the agent run this z/OS command: `/P zos.taskname`

   Replace `zos.taskname` with the DA agent task name (see *zos.properties*).

8. When running the agent as a started task, set a RACF STARTED profile: `RDEF STARTED procname*.* STDATA(USER(userid) TRUSTED(NO))`

9. When running with JZOS in LPA, execute the following command once per LPAR IPL: `SETPROG ADD,LPA,DSN=java.loadlib,MEMBER=JVMLDM86`

10. To run the DA Agent as a batch job execute the following TSO command:

    ```
    TSO SUBMIT 'zos.da.jcllib(zos.jobname)'
    ```

    Replace the variable `zos.da.jcllib` with the name of the mainframe PDS dataset where the DA agent procedure and batch job is installed (see *zos.properties*).

    Replace the variable `zos.jobname` with the name of the batch job (see *zos.properties*).

    Verify that the owner or submitter of the z/OS DA agent batch job has full permission to access the z/OS DA agent home directory.

Installing for a High Availability Implementation

An active-active high availability server implementation distributes the processing load of agent communication across multiple servers rather than relying on the processing of a single server. In large enterprise production environments where many agents are communicating at once, this enables faster response times.

If you are not installing multiple servers for a high availability, or active-active, Deployment Automation server implementation, you can skip this section.

For details on high availability implementations, see the following:

- Installing the First Server [page 88]
- Installing Additional Servers [page 92]
- High Availability Configurations [page 93]

Installing the First Server

Install the first server for a high availability (active-active) implementation as follows:

1. Create an Oracle or SQL Server database on a separate machine. See Preparing Your Database [page 43].

2. Install the first Deployment Automation server. This installs a profile into the designated installation folder, which by default is the following:

   ```
   C:\Users\username\.microfocus\da
   ```

   This profile will be shared with subsequent server installations. Ensure that the installation folder can be accessed by all the machines on the active-active cluster.

   See Server Installation [page 44].

3. Update the Hibernate cache properties in the installation properties file as follows:

   a. Open the Deployment Automation shared user profile `installed.properties` file. The default path is as follows:

```
C:\Users\username\.microfocus\da/conf/server/installed.properties
```

b. Disable the Hibernate query cache property as follows:

```
hibernate.cache.use_query_cache=false
```

Note: For increased I/O throughput, query caching is used by default. However, the Hibernate cache must be disabled in active-active configurations, because it doesn't work across multiple server instances.

c. Enable the Hibernate second level cache property as follows:

```
hibernate.cache.use_second_level_cache=true
```

d. Change the following line. This references the cache file you will create in the next step.

```
net.sf.ehcache.configurationResourceName=ehcache-active-active.xml
```

4. Create a cache properties file for your active-active servers as follows:

a. Copy and paste the following into a new file called `ehcache-active-active.xml`.

```
<ehcache updateCheck="false">
<diskStore path="java.io.tmpdir" />
<cacheManagerPeerProviderFactory
 class="net.sf.ehcache.distribution.RMICacheManagerPeerProviderFactory"
 properties="peerDiscovery=manual,
 rmiUrls=//<ip_server1>:<port>/deployServerCache|
 //<ip_server1>:<port>/securityServerCache|
 //<ip_server1>:<port>/vcCache|
 //<ip_server1>:<port>/org.hibernate.cache.internal.StandardQueryCache|
 //<ip_server1>:<port>/org.hibernate.cache.spi.UpdateTimestampsCache|
 //<ip_server1>:<port>/defaultCache|
 //<ip_server2>:<port>/deployServerCache|
 //<ip_server2>:<port>/securityServerCache|
 //<ip_server2>:<port>/vcCache|
 //<ip_server2>:<port>/org.hibernate.cache.internal.StandardQueryCache|
 //<ip_server2>:<port>/org.hibernate.cache.spi.UpdateTimestampsCache|
 //<ip_server2>:<port>/defaultCache" />
<cacheManagerPeerListenerFactory
 class="net.sf.ehcache.distribution.RMICacheManagerPeerListenerFactory"
 properties="port=<port>, socketTimeoutMillis=200000"/>
<cache name="deployServerCache"
 maxElementsInMemory="10000"
 eternal="false"
 timeToLiveSeconds="86400"
 overflowToDisk="true"
 maxElementsOnDisk="100000"
 diskPersistent="true"
 diskExpiryThreadIntervalSeconds="86400" >
 <cacheEventListenerFactory
 class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"/>
</cache>
<cache name="securityServerCache"
 maxElementsInMemory="10000"
 eternal="false"
 timeToLiveSeconds="86400"
 overflowToDisk="true"
 maxElementsOnDisk="100000"
 diskPersistent="true"
 diskExpiryThreadIntervalSeconds="86400" >
 <cacheEventListenerFactory
 class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"/>
</cache>
<cache name="vcCache"
 maxElementsInMemory="10000"
 eternal="false"
 timeToLiveSeconds="86400"
 overflowToDisk="true"
 maxElementsOnDisk="100000"
 diskPersistent="true"
 diskExpiryThreadIntervalSeconds="86400" >
 <cacheEventListenerFactory
 class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"/>
</cache>
<cache name="org.hibernate.cache.internal.StandardQueryCache"
 maxElementsInMemory="10000"
 eternal="false"
 timeToLiveSeconds="86400"
 overflowToDisk="true"
 maxElementsOnDisk="100000"
 diskPersistent="true"
 diskExpiryThreadIntervalSeconds="86400" >
 <cacheEventListenerFactory
 class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"/>
</cache>
<cache
 name="org.hibernate.cache.spi.UpdateTimestampsCache"
 maxEntriesLocalHeap="10000"
 eternal="true">
 <cacheEventListenerFactory
 class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"/>
</cache>
<defaultCache
 maxElementsInMemory="10000"
 eternal="false"
 timeToLiveSeconds="86400"
 overflowToDisk="true"
 maxElementsOnDisk="100000"
 diskPersistent="true"
 diskExpiryThreadIntervalSeconds="120" >
 <cacheEventListenerFactory
 class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"/>
</defaultCache>
</ehcache>
```

b. Save the file to the Deployment Automation shared user profile server directory. For example:

`C:\Users\`*`username`*`\.microfocus\da/conf/server/ehcache-active-active.xml`

c. Search for the `rmiUrls` property. In that property, modify and add lines for server IP addresses and port to match those for the servers and port you are using in your active-active configuration.

   In the following example, the first server is `10.136.40.114` and the second server is `10.136.40.250`. The port is `40001`.

```
<cacheManagerPeerProviderFactory
 class="net.sf.ehcache.distribution.RMICacheManagerPeerProviderFactory"
 properties="peerDiscovery=manual,
 rmiUrls=//10.136.40.114:40001/deployServerCache|
 //10.136.40.114:40001/securityServerCache|
 //10.136.40.114:40001/vcCache|
 //10.136.40.114:40001/org.hibernate.cache.internal.StandardQueryCache|
 //10.136.40.114:40001/org.hibernate.cache.spi.UpdateTimestampsCache|
 //10.136.40.114:40001/defaultCache|
 //10.136.40.250:40001/deployServerCache|
 //10.136.40.250:40001/securityServerCache|
 //10.136.40.250:40001/vcCache|
 //10.136.40.250:40001/org.hibernate.cache.internal.StandardQueryCache|
 //10.136.40.250:40001/org.hibernate.cache.spi.UpdateTimestampsCache|
 //10.136.40.250:40001/defaultCache" />
```

d. Save the file.

5. Update the Common Tomcat set environment file as follows:

   a. Navigate to the Common Tomcat path bin directory. For example:

      `C:\Program Files\Micro Focus\common\tomcat\8.5\bin`

   b. Edit `setenv.bat` for Windows or `setenv.sh` for Unix/Linux.

   c. Find the line `JAVA_OPTS=...` for Windows or `CATALINA_OPTS=...` for Unix/Linux and add the following property to it:

      `-Djava.rmi.server.hostname=<ip_server1>`

      where `<ip_server1>` is the IP address of this first server

   d. Restart Common Tomcat.

6. Share the directory where the Deployment Automation profile is stored. Configure proper access to its content for network clients. To share the profile location, use the following guidelines:

   - The shared network storage should not reside on any of the Deployment Automation servers; otherwise it introduces a dependency on one node always being available and invalidates High Availability principles.

   - Shared storage should be accessed using a symbolic directory link or mapped drive created on each of the Deployment Automation servers.

CAUTION:

⚠ If you use a UNC path on Windows, you cannot run Deployment Automation as a service. Instead, use a mapped drive.

- The Common Tomcat service on each of the Deployment Automation servers needs to be configured to run using a domain service account. That account must be given full access to the following:
    - The shared network location
    - The Deployment Automation and Common Tomcat installation directories on each Deployment Automation server

7. In the Deployment Automation user interface for this first server, create a network relay for each server that is part of your active-active configuration.

    a. In the Deployment Automation user interface, navigate to **Management > Resources**.

    b. In the selection box, select **Relays.**

    c. Click the **Relay Actions** button and then select **Create Network Relay**.

    d. In the **Create Network Relay** dialog, enter a **Name** for the server.

    e. In the **Host** field, enter `ip_servern`, where `ip_servern` is the IP address of server 1 (this server) through n. This is the remote hostname or IP address to which current server will connect.

    f. Enter the **JMS Port** number, such as `7918`, that the current server will use to connect to another server using the network relay.

    g. Select the **Active** check box to tell the server to start the connection with another server using the network relay.

    h. Click **Save**.

    i. Repeat this process for each server.

8. If you are connecting the agents to a single endpoint, with your Deployment Automation servers defined as a cluster with a load balancer, do the following:

    a. In the first server installed in your active-active installation, in the Deployment Automation user interface, navigate to **Administration > System.**

    b. Select **System Settings** in the selection box.

    c. Modify the **External Agent URL** and **External User URL** fields to use the IP address or the DNS name and port of the load balancer server.

Installing Additional Servers

Install additional servers (servers 2 through n) for a high availability (active-active) implementation as follows:

1. If installing on Unix/Linux, make sure you have mounted the network share with the Deployment Automation profile as follows:
   1. Open the `/etc/fstab` file for editing.
   2. Add a line for your profile share. For example:

      ```
      //serverName/Temp/sharedProfile /mnt/localProfileDir cifs _netdev,
      uid=1000,iocharset=utf8,sec=ntlm,username=user@domain.com,
      password=password,noserverino 0 0
      ```

   3. Save the file.
2. Run the installer and set the **Server Install Location** field to the shared Deployment Automation profile.
3. Select the **Use existing settings** option so that the installation will use the same database as the first server installation.
4. Update the Common Tomcat set environment file as follows:
   1. Navigate to the Common Tomcat path `bin` directory. For example:

      ```
      C:\Program Files\Micro Focus\common\tomcat\8.5\bin
      ```

   2. Edit `setenv.bat` for Windows or `setenv.sh` for Unix/Linux.
   3. Find the line `JAVA_OPTS=...` for Windows or `CATALINA_OPTS=...` for Unix/Linux and add the following property to it:

      ```
      -Djava.rmi.server.hostname=<ip_servern>
      ```

      where `<ip_servern>` is the IP address of the second or nth server
   4. Restart Common Tomcat on all servers.
5. Install agents as described in High Availability Configurations [page 93].

High Availability Configurations

For information on high availability implementations, including disaster recovery scenarios, see the following topics:

- Configuring Agents for High Availability (Recommended) [page 94]
- Configuring Agents for High Availability (Alternate) [page 94]
- Disaster Recovery with Hot Standby [page 95]
- Disaster Recovery with Cold Standby [page 96]

Related Topics

Server and Agent Communication Configuration [page 73]

Configuring Agents for High Availability (Recommended)

The preferred approach for High Availability (HA) in enterprise environments is to connect agents to a single server endpoint that is part of cluster with a load balancer.

Note: In certain circumstances, for example where high availability is required but enterprise level support is not, the alternative approach, Configuring Agents for High Availability (Alternate) [page 94], may also be taken.

To configure agents to connect to a single server endpoint:

1. Install the Deployment Automation server application on *n* servers for the active-active feature. See Installing for a High Availability Implementation [page 88]. These *n* servers are used as a cluster with a load balancer, so make sure you have set the first server's **External Agent URL** and **External User URL** system settings to point to the IP address or the DNS name and port of the load balancer cluster .

2. To make it possible for agents to communicate with each of the server nodes within the cluster, during the agent installation, the agent must point to the load balancer server. In this scenario, the load balancer will choose which of the server nodes communicates with the agent.

   To ensure each agent communicates properly through the load balancer server, do the following:

   a. When installing the agents:

      - In the agent installer **Server Details** in **Hostname or address**, use the IP address or the DNS name of the load balancer server.

      - In the agent installer **Server Details** in **Agent Communication Port**, use the port number of the load balancer server.

      See Agent Installation [page 75].

Configuring Agents for High Availability (Alternate)

Another approach for maintaining high availability is to connect agents to a series of endpoints. However enterprise-level scalability is not supported with this approach and it is difficult to maintain with a full list of all servers. Use this approach with caution. The preferred approach for high availability in enterprise environments is Configuring Agents for High Availability (Recommended) [page 94].

The following steps describe how to connect agents to a series of server endpoints with which, in the case of a server failure, the agents will attempt to communicate.

For the purpose of this example, the following procedure assumes a two-node cluster. Two example cluster nodes have the following IP addresses: `ip_node_1` and `ip_node_2` (or you can use DNS names).

To configure agents to connect to a series of endpoints:

Assuming that during installation, an agent was set to communicate with the server at `ip_node_1`, to enable the agent to also communicate with the server on the cluster node `ip_node_2`, you should configure the two servers with a network relay as follows:

1. Install the Deployment Automation server application on two servers for the active-active feature. See Installing the First Server [page 88]. These two servers are *not* used as a cluster with a load balancer.

2. In the Deployment Automation user interface, navigate to **Management > Resources** .

3. In the selection box, select **Relays**.

4. Click the **Relay Actions** button and then select **Create Network Relay**.

5. In the **Create Network Relay** dialog:

   a. Enter a **Name** for the second server.

   b. In the **Host** field, enter `ip_node2`. This is the remote hostname or IP address the server will use to connect to the network relay.

   c. Enter the **JMS Port** number that the server will use to communicate with the network relay over JMS protocol.

   d. Select the **Active** check box to tell the server to start the connection to the network relay.

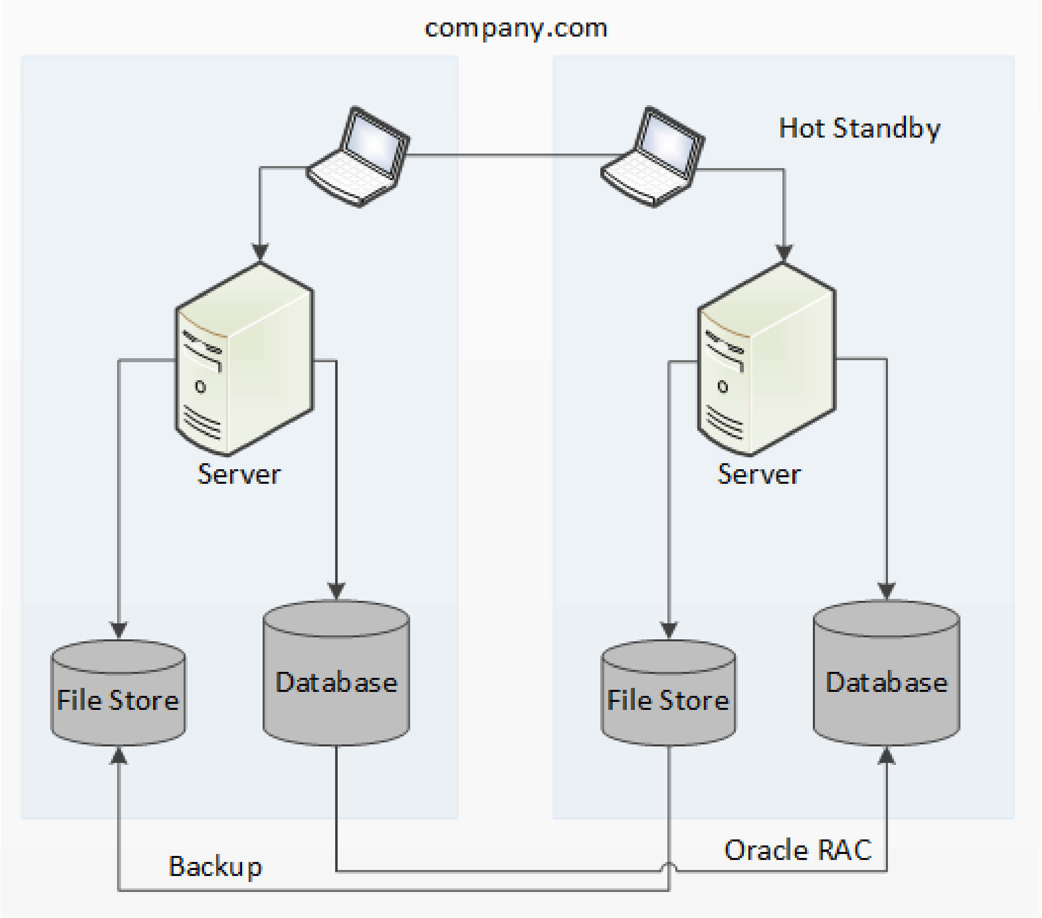
   e. Click **Save**.

6. Repeat this process for each cluster node.

Disaster Recovery with Hot Standby

For disaster recovery where high availability is required, you can use a hot standby strategy to ensure server availability in case of system failure. In this strategy, you maintain two servers and backup the file store and database as part of your regularly-scheduled system backup procedures. This enables you to immediately switch to the hot standby system should the primary system fail. A hot standby system configuration is shown in the following figure.

Disaster Recovery with a Hot Standby System



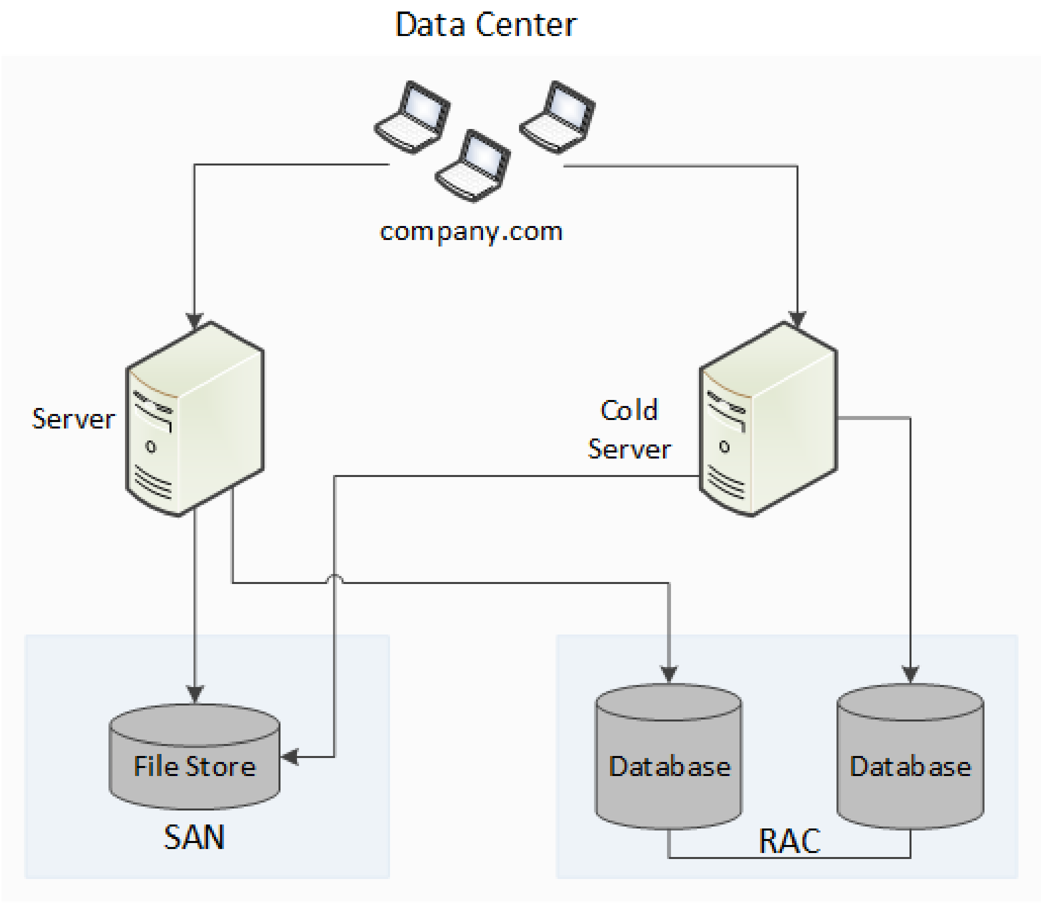
Disaster Recovery with Cold Standby

For disaster recovery where high availability is not required, you can use a cold standby strategy to ensure server availability in case of system failure. When the primary system fails, the cold standby is brought online and promoted to the primary server. Once online, the standby reestablishes connections with all agents, performs recovery, and proceeds with any queued processes. Because the most intense work is handed-off to agents, a high performance configuration should not have an agent installed on the same hardware as the main server.

When using the cold standby data center configuration, you should typically configure the data tier with network storage and a clustered database. The service tier performs best when it's on a dedicated, stable, multi-core machine with a fast connection to the data tier. A standby machine should be maintained and kept ready in case the primary server goes down.

A typical cold standby data center configuration is shown in the following figure.

Disaster Recovery with a Cold Standby System


Data Center
company.com
Server
Cold
Server
File Store
SAN
Database
Database
RAC

Chapter 5: Running Deployment Automation

The following topics describe how you start the Deployment Automation services after you have finished your installation.

Starting and Stopping the Server

The installer installs Deployment Automation under the application server and starts the service automatically.

To start or stop the server service:

For Windows

Use `Administrative Tools > Services` to start or stop **Common Tomcat**.

For UNIX/Linux

Execute the associated command script as follows:

1. Navigate to the *server_installation*`/bin` directory. For example: `/opt/MicroFocus/da/common/tomcat/8.5/bin`
2. Run `startup.sh` to start the server or `shutdown.sh` to stop it.

Starting and Stopping an Agent Relay

If you are using an agent relay and you install the agent relay as a service, the agent relay installer installs the agent relay service but does not start the service. It is set to manual start by default.

Start the agent relay service on your agent relay machine according to the following procedure.

Note: If you are using an agent relay, you must start it before starting any agents that will communicate through it.

To stop or start an agent relay service:

- In Windows, use `Administrative Tools > Services` to start or stop the agent relay of the name you gave it during the installation. The default name is `agentrelay`.

- On UNIX/Linux, to start or stop the agent relay, execute the associated command script as follows:

1. Navigate to the `agent_relay_installation/bin` directory. For example: `/opt/MicroFocus/agentrelay/bin`
2. Run `agentrelay start` to start the server in a new shell. Run `agentrelay stop` to stop it.

Starting and Stopping an Agent

The agent installer installs the agent service, but does not start the service. It is set to manual start by default.

Note: If you are using an agent relay, you must start it before starting any agents that will communicate through it.

To stop or start an agent service:

For Windows

Use `Administrative Tools > Services` to start or stop the agent, **DA-Agent**.

For UNIX/Linux

Execute the associated command script as follows:

1. Navigate to the `agent_installation/bin` directory. For example:

   `/opt/microfocus/Deployment Automation Agent/core/bin`

2. Run `agent {start|stop}` to start or stop the agent.

   Note: For information on configuring your agents to start automatically under a dedicated account, see the Community website.

Once the agent has started, in the Deployment Automation user interface, navigate to **Management > Resources** and select **Relays** in the selection box. The agent should be listed with a status of `Connected`.

Permission for Users Running Agents

Each agent needs the appropriate permission to communicate with the Deployment Automation server.

- For production environments, you should create a user account dedicated to running the agent on the machine where the agent is installed.
- For simple evaluations, the administrative user can run the agent on the machine where the server is located.

At a minimum, the user running the agent should have permission to do the following:

- **Create a cache**. By default, the cache is located in the home directory of the user running the agent. The cache can be moved or disabled.
- **Open a TCP connection**. The agent uses a TCP connection to communicate with the server's JMS port.

- **Open a HTTP(S) connection.** The agent must be able to connect to the Deployment Automation user interface to download artifacts from CodeStation.

- **Access the file system**. Many agents need read/write permission to items on the file system.

Accessing Deployment Automation

After you have completed at minimum a server installation, you can login to the user interface to start configuring Deployment Automation.

To access the Deployment Automation user interface:

1. Open a web browser and enter the following URL:

   ```
   http://serverName:port/da/
   ```

   Use the host name and port for the application server where you installed Deployment Automation. The default port is `8080` for Common Tomcat.

2. Log in to the server using the credentials (`Username` and `Password`) you set up during the server installation.

Part 3: Administration

This section contains information on configuring Deployment Automation as detailed in the following topics:

Chapter 6: Administration Overview

You configure selections that will be available to users when they work with Deployment Automation through the **Administration** page.

- **Automation**

  See Chapter 7: Automation Administration [page 107].

- **Security**

  See Chapter 8: Security Administration [page 115].

- **System**

  See Chapter 9: System Administration [page 143].

- **Reports**

  See Security Report Overview [page 139].

- **Timeline**

  See Scheduling Version Cleanup [page 157].

- **Action**

  Select from administration actions, including the following:

  - **New:** User, Group, License
  - **Replication:** Export, Import
  - View Output Log
  - Print Page
  - Help

Chapter 7: Automation Administration

The Administration Automation options enable you to load plugins, manage process locks, and configure other entities that support automation, as documented in the following topics:

- Managing Plugins [page 107]
- Managing External Source Configuration Types [page 108]
- Locks [page 108]
- Post-Processing Scripts [page 109]
- Defining Statuses [page 110]

Managing Plugins

Deployment Automation includes several standard plugins that are available when you install. Many additional plugins are available for you to download and use for your specific needs. Plugins can be installed, or loaded, at any time, directly from the Deployment Automation user interface.

For more information, see the following topics:

- Loading Plugins [page 107]
- Upgrading Plugins [page 108]

Loading Plugins

After you have downloaded a plugin, you can load it so that it becomes available to use in the Process Editor.

Note: If you download a bundled zip file that contains multiple plugins, you will need to extract the individual plugin zip files from the bundle. You should not decompress the individual plugin zip files.

To load a plugin:

1. Navigate to **Administration > Automation**.
2. Select the **Plugins** tab.
3. Click the **Load Plugin** button.
4. Click **Choose File** and select the compressed plugin (.zip) file.
5. Click **Load**.

After the plugin load process completes successfully, when you are designing processes for components, the plugin is listed in the Process Editor on the **Automation Plugins** pane. Once loaded, plugin functionality is available immediately.

Upgrading Plugins

When a new version of a plugin is available, you can upgrade the plugin to take advantage of new steps, new fields, and any other improvements.

To upgrade a plugin:

1. Load the new version of the plugin. See Loading Plugins [page 107]. All processes that use this plugin are automatically upgraded to use the new version.
2. Re-test the processes as needed.

Managing External Source Configuration Types

You can load, view information about, and delete external source configuration types in the **Automation > Source Config Type** page.

- External source configuration types enable closer interaction between Deployment Automation and the tool being accessed by the source configuration type. If **Copy to CodeStation** is not selected for these types, you can use the corresponding plugin to download artifacts directly from that tool.
- Custom source configuration types can be developed by Java programmers using the external source configuration type architecture.

For information on creating and loading custom external source configuration types, see "Creating External Source Configuration Types" in the *Deployment Automation Integration Guide*.

Note: Deployment Automation includes several built-in source configuration types that are installed with the product. These are automatically available when creating components and include external and non-external source configuration types. See Source Configuration Type Fields [page 203].

Locks

Deployment Automation uses locks to ensure that processes do not interfere with one another. Normally, once a lock is no longer needed, it is released. Sometimes a lock will not get released and its associated process will be unable to complete. The lock management feature enables you to quickly identify and resolve abnormal lock conditions.

Managing Locks

A running process with a lock, like all active processes, appears with a status of `Running`. If a locked process takes longer to complete than expected, you can cancel the process, or investigate it fully from the **Locks** page.

1. Navigate to **Administration > Automation**.
2. Select the **Locks** tab.

   The **Locks** page lists any locks with the following information:

Field	Description
Name	The name identifies the lock and is defined in the lock step properties. The default name is a concatenation of the object name + process name + resource name, with slight variations for deployment package and global processes.
Type	This indicates whether the process creating the lock is a component, application, deployment package, or global process.
Component / Application / Deployment Packages	This displays the name of the object containing the lock. Click the link to display the corresponding page.
Resource / Environment	This displays the name of the resource or environment containing the lock. Click the link to display the Resources or Environments page.
Process	This displays the name of the process containing the lock. Click the link to display the process in the process editor.

3. Resolve the lock by selecting one of the following actions:

Action	Description
View Execution Details	Displays the process log for the process containing the lock. View its details to see the name of the process step causing the lock.
Release	Releases the lock, which enables the associated process to continue processing.

If the Deployment Automation server and or agents go down while a locked process is running, Deployment Automation will automatically restore any interrupted processes along with any locks they might contain once service is restored.

Post-Processing Scripts

Deployment Automation component processes perform post-processing whenever a plugin step finishes execution. Typically, post-processing scripts ensure that expected results occurred. You can use your own script instead by instructing Deployment Automation to use your script, which must be written in JavaScript, when you define the step. See Designing Component Processes [page 241].

When a step finishes, the agent performing the step will run your script. When the agent runs the script, it first loads the server log file and finds the exit code property of the target step using regular expressions defined in the script. It then applies any actions defined in the script before processing the next step. See Output Log [page 151].

To create a post-processing script:

1. Navigate to **Administration > Automation**.

2. Select the **Post Processing Scripts** tab.

3. Click the **Create** button.

4. Enter a name for the script into the **Name** field. The name must match the name you specified when you defined the process step. See Designing Component Processes [page 241].

5. Enter or paste the Javascript script into the **Script Body** field. See the roll-over help next to the field for information about the properties and variables available for user-defined scripts.

6. Click **Save**.

Defining Statuses

You can pre-define implementation-specific inventory, component version, and snapshot status values and subsequently use them to do the following:

- Apply inventory statuses to component versions when they are successfully deployed to resources

- Apply and enforce conditions on application gates based on statuses of component versions and snapshots

To add status values, navigate to **Administration > Automation** > **Statuses**. Add values as needed for your implementation.

Each status has a required name and several optional values, as described in the following table:

Status Attributes Table

Value	Description
Name	Identifies the status; appears in user-interface. Used to create gates, and available in process steps.
Color	Hexadecimal color definition; determines the color displayed in the user interface.
Unique	Boolean value (true|false). Only one component version with this status/attribute will be deployed to the environment.
Required Component Role	Security role required by user to add this status to the component version.

Note: While you can add as many values as you like, you cannot create new status types. Only Inventory, Version, and Snapshot status types are supported.

Note: Process completion status is built in and cannot be changed. This should not be confused with these other status types.

The following topics describe the use of the statuses.

- Using Inventory Statuses [page 111]
- Using Version Statuses [page 112]
- Using Snapshot Statuses [page 112]

Using Inventory Statuses

Inventory statuses are used with component processes to apply statuses to component versions upon successful deployment to a target environment or resource. *Active* is the default setting, which indicates that a component version has been successfully deployed to its target resource.

You use inventory statuses as follows:

1. **Define the statuses**

   Navigate to **Administration > Automation > Statuses**. Select the **Inventory** tab and click **Create**. Define the inventory statuses for your implementation. For example, *UAT_Ready*, *INT_Ready*, *Prod_Ready*.

2. **Select the inventory status**

   When you create or edit a component process, select the inventory status that is to be set for this process. See Creating Component Processes [page 239].

3. **Request a process**

   When the component process executes successfully, the selected status is applied to component versions for which the process was defined. The inventory status appears on the **Inventory** tab for the component itself and in environments that ran the component process. See Running Application Processes [page 288].

Note:

You can check the inventory status of an environment as part of component and global processes using the DA Environment plugin *Verify Inventory Status* step.

See Creating Component Processes [page 239] for more information on the use of inventory statuses for your inventory management.

Using Version Statuses

Version statuses are used with environment gates to ensure that components are deployed or otherwise processed if they are in a designated status. You can select from available version statuses when you define gates for applications. There is no default setting.

You use version statuses as follows:

1. **Define the statuses**

   Navigate to **Administration > Automation > Statuses**. Select the **Version** tab and click **Create**. Define the version statuses for your implementation. For example, *INT_Ready*, *UAT_Ready*, *PROD_Ready*.

2. **Add the condition**

   See Gates [page 295].

   Note: In the **Add a new condition** selection box, both version statuses and snapshot statuses appear for selection.

3. **Set the specific component version's status**
   See Adding Statuses to Component Versions [page 234].

   Note: You can also set version status as part of component and global processes using the DA Version plugin.

4. **Request a process to deploy the component**

   The requested process will not run if any of the included component versions are not set to the status or statuses specified in the application environment's **Gates** tab. See Chapter 29: Ways of Deploying [page 367].

Using Snapshot Statuses

Snapshot statuses are used with environment gates to ensure that snapshots are deployed or otherwise processed only if they are in a designated status. You can select from available snapshot statuses when you define gates for applications. There is no default setting.

You use snapshot statuses as follows:

1. **Define the statuses**

Navigate to **Administration > Automation > Statuses**. Select the **Snapshot** tab and click **Create**. Define the snapshot statuses for your implementation. For example, *INT_Ready*, *UAT_Ready*, *PROD_Ready*.

2. **Add the condition**

   See Gates [page 295].

3. **Set the specific snapshot's status**

   See Adding Statuses to Snapshots [page 294].

4. **Request a process to deploy the snapshot**

   The requested process will not run if the snapshot is not set to the status or statuses specified in the application environment's **Gates** tab. See Chapter 29: Ways of Deploying [page 367].

Chapter 8: Security Administration

Deployment Automation provides a flexible role-based security model that maps to your organizational structure, as described by the following topics:

- Setting up Security [page 115]
- Role Configuration [page 116]
- Authorization Realms and Groups [page 121]
- Authentication Realms and Users [page 124]
- Default Permissions [page 136]
- Adding Users to Groups [page 137]
- Setting Role Membership by Product Area [page 138]
- Tokens [page 138]
- Viewing Online Users [page 139]
- Security Report Overview [page 139]

Setting up Security

Typically, you perform the following steps in order when setting up Deployment Automation your initial security. Thereafter, you will administer each security area as needed.

- **Create Roles**

  Create roles and define permissions for the various product areas. For most evaluations, the default roles should be adequate.

  See Role Configuration [page 116].

- **Create Authorization Realms and Groups**

  Authorization realms are used by authentication realms to associate users with groups and to determine user access.

  See Authorization Realms and Groups [page 121].

- **Define Default Permissions**

  Set default permissions by product area. You can set default permissions for all users or by group.

  See Default Permissions [page 136].

- **Create Authentication Realms and Add Users**

The authentication realm is used to determine a user's identity within an authorization realm. Add users to appropriate authentication realms. If more than one realm has been configured, user authentication is determined following the hierarchy of realms defined on the Authentication pane. When a user attempts to log in, all realms are polled for matching credentials.

See Authentication Realms and Users [page 124].

- **Add Users to Groups**

  Add members to groups. Users who are members of groups inherit the groups' permissions.

  See Adding Users to Groups [page 137].

- **Set Role Membership by Product Area**

  To further refine the permissions by role, you can give groups and users role membership as follows:

  - For most product areas, set these in the **Security** tab for each specific product area and item.
  - Set **System Security** and **UI Security** in the **Administration > Security** options of the same names.

  Note: Environment and component security settings can be set at the direct object level and within the applications to which they are associated.

  For component security, no matter whether you set the security on the component level or application component level, the settings are applied everywhere.

  Environment security settings are handled as follows:

  - Upon initial association with an application, the security that is set for the environment is inherited by the application environment.
  - After an environment is associated with an application, if the security is changed directly on the environment, the setting for the application environment is not affected.
  - If the security is changed for the application environment, the setting at the direct environment level is not affected.

  See Setting Role Membership by Product Area [page 138].

Role Configuration

Roles provide the building blocks for the security system. Roles have permissions that define the actions the roles can perform with product features. Typical actions include changing or executing an item, such as an application process, or modifying its security settings. Users or groups assigned to a role are automatically granted the permissions configured for it. The default roles can be edited and new roles can be created.

Deployment Automation maps key product features or areas to security roles. Each area has several permissions defined for it. When you create a role, you first specify the product area. Selecting a product area defines the set of permissions available to the new role; only permissions defined for the area are available.

Use the UI security area to quickly assign access permissions to the different areas of the Deployment Automation Web application. Use the system security area to assign usage permissions, including the ability to define security for other users.

For information on configuring roles, see the following topics:

- Default Role Types [page 117]
- Creating and Editing Roles [page 118]
- Server Roles and System Security [page 118]
- Web UI Roles and UI Security [page 120]

Default Role Types

Deployment Automation ships with several role types mapped to product areas. Each role type has a default set of roles that are given permissions relevant for that role type. You can define additional user-defined roles within each role type and give those roles permissions that you need for your implementation of Deployment Automation.

Each role type has one role, typically called `Admin` or `Administrator`, which has all permissions available for that area. Deleting a default Admin role for one role type does not affect the Admin role for another type.

Default Admin Role Permissions Table

The following table lists the default permissions for the default Admin role for each default role type except for Server and Web UI role types. For information on those role types, see Server Roles and System Security [page 118] and Web UI Roles and UI Security [page 120] respectively.

Role Type	Read	Wrt.	Sec.	Exec.	Mng. Snps.	Run Cmp. Proc.	Mng. Vers.	Clean Exec. Hist.
Agents	X	X	X					
Applications	X	X	X	X	X	X		
Component Templates	X	X	X					
Components	X	X	X	X			X	
DA Server: See Server Roles and System Security [page 118]								
Deployment Packages	X	X	X	X				
Environments	X	X	X	X				

Role Type	Read	Wrt.	Sec.	Exec.	Mng. Snps.	Run Cmp. Proc.	Mng. Vers.	Clean Exec. Hist.
Licenses	X	X	X					
Pipelines	X	X	X					
Processes	X	X	X	X				
Relays	X	X	X					
Resources	X	X	X	X				
Web UI: See Web UI Roles and UI Security [page 120]								

Creating and Editing Roles

To create and edit roles:

1. Navigate to **Administration > Security**.
2. Select the **Role Configuration** tab.
3. On the side menu select the role type for which you want to add a role.
4. To create a role, click the **Create Role** button. To edit a role, click the **Edit** icon beside the role.

   All permissions available for this role type appear.

   By default, for most product areas, Deployment Automation provides an `Admin` role with all configured permissions granted.

   For default permissions by role, see Default Role Types [page 117].

   Exceptions are for Server Roles and Web UI Roles. For information on default permissions for these, see Server Roles and System Security [page 118] and Web UI Roles and UI Security [page 120] respectively.
5. Select the permissions you want granted to this role.
6. Click **Save**.

Server Roles and System Security

Server roles define the functions users can perform with the Deployment Automation server. Available permissions are described in the following table.

Server Permissions Table

Permission	Grants Users Ability To ...
Check for Update	Check for available updates for server versions
Create Applications	Create applications
Create Component Templates	Create component templates
Create Components	Create components
Create Deployment Packages	Create deployment packages
Create Environments	Create environments
Create Pipelines	Create pipelines
Create Relays	Create relays
Create Subresources	Create sub-resources
Create/Manage Resource Roles	Create and delete resource roles
Data Collection	Select the option to allow data collection for your Deployment Automation server
Manage Custom Source Config Types	Load and delete custom source config types.
Manage Licenses	Add and remove licenses
Manage Plugins	Install new plugins
Security	Manage security configuration; users without this permission cannot access or change the security functions

To add users to system security roles:

1. Navigate to **Administration > Security**.
2. Select the **System Security** tab.

   All defined roles appear.

   Default Server Roles Table

Role	Default Permissions
Configuration Manager	Can create new applications, components, component templates, pipelines, resources, and resource roles. Has a single group pre-assigned, `Configuration Group`.
Admin	Has all configured permissions granted and a single group pre-assigned, `Admin Group` with user *admin*.

3. Click the **Add Role Members** button and select one or more roles.

4. Click **Add**.

Web UI Roles and UI Security

These roles determine which parts of the Deployment Automation user interface users can access. Each page, such as **Reports**, can be restricted.

UI Security Table

The permissions for each page or product area are described in the following table.

Permission	Description
Applications	Grants users access to the **Applications** page
Clean Execution History	Grants users access to the **Clean Execution History** action item on the **Administration** page
Components	Grants users access to the **Components** page
Configuration	Grants users access to the **Configuration** page
Dashboard	Grants users access to the **Processes** overlay
Deployment Calendar	Grants users access to the **Calendar** page
Deployment Packages	Grants users access to the **Deployment Packages** page
Environments	Grants users access to the **Environments** page
Pipelines	Grants users access to the **Pipelines** page
Reports	Grants users access to the **Reports** page
Relays	Grants users access to the **Relays** page
Resources	Grants users access to the **Resources** page

Permission	Description
Settings	Grants users access to the **System Settings** page
Work Items	Grants users access to the **Approval Requests** overlay

To add users to user interface roles:

1. Navigate to **Administration > Security**.
2. Select the **UI Security** tab.

   All defined roles appear.

   Default Web UI Roles Table

   The default permission settings for the user interface roles are described in the following table.

Role	Default Permissions
Admin	Can access all pages except **Clean Execution History**
Approver	Can access the **Reports** pages and **Approval Requests** and **Processes** overlays
Configuration Engineer	Can access all pages except **Calendar** and **Approval Requests**
Deployment Engineer	Can access the **Reports** and **Calendar** pages and **Approval Requests** and **Processes** overlays

3. Click the **Add Role Members** button and then select one or more roles.
4. Click **Add**.

Authorization Realms and Groups

Authorization realms associate users with roles and work with authentication realms to determine which users can access Deployment Automation.

For information on creating and configuring authorization realms and groups, see the following topics:

- Creating an Internal Storage Authorization Realm [page 121]
- Creating an LDAP Authorization Realm [page 122]
- Creating Groups [page 123]

Creating an Internal Storage Authorization Realm

Internal Storage type authorization realms are stored in the Deployment Automation database and use internal role management.

The default authorization realm, Internal Security, is of the Internal Storage type, and includes the following default groups:

- Admin Group
- Approve Group
- Configuration Group
- Deploy Group
- Read Only Group

If you want to create additional Internal Storage authorization realms, you can do so using the following procedure.

To create an Internal Security authorization realm:

1. Navigate to **Administration > Security**.
2. Select the **Authorization (Groups)** tab.
3. Select **Authorization Realms** in the side menu.
4. Click the **Create Authorization Realm** button.
5. In **Type** select `Internal Storage` and specify a unique name.

Creating an LDAP Authorization Realm

An LDAP authorization realm uses an external LDAP server for authorization and uses external LDAP role management.

Deployment Automation includes an internal database for storing security information and provides an integration with the Lightweight Directory Access Protocol (LDAP). LDAP is a widely-used protocol for accessing distributed directory information over IP networks. If you are implementing a production version of Deployment Automation, the LDAP integration is recommended. If you are evaluating Deployment Automation, it is not necessary to set up the LDAP integration; full security is configured and enforced by the server.

To create an LDAP authorization realm:

1. Navigate to **Administration > Security**.
2. Select the **Authorization (Groups)** tab.
3. Select **Authorization Realms** in the side menu.
4. Click the **Create Authorization Realm** button.
5. From the **Type** list select `LDAP` and specify a unique name along with the following information:

Field	Description
User Group Attribute	Name of the attribute that contains role names in the user directory entry. If user groups are defined in LDAP as an attribute of the user, the Group Attribute configuration must be used
Group Search Base	Base directory used to execute group searches, such as `ou=employees,dc=mydomain,dc=com`.
Group Search Filter	LDAP filter expression used when searching for user entries. The name will be substituted in place of 0 in the pattern, such as `uid={0}`. If this is not part of the DN pattern, wrap the value in parenthesis, such as `ud=(0)`.
Group Name	Directory name used to bind to LDAP for searches, such as `cn=Manager,dc=mycompany,dc=com`. If not specified, an anonymous connection will be made. Required if the LDAP server cannot be anonymously accessed.
Search Group Subtree	If selected, searches the subtree for the roles.

For an example, see the following:

- Limiting LDAP to a Set of Groups [page 123]

Limiting LDAP to a Set of Groups

To limit the LDAP authorization realm to a set of groups, you must use the LDAP filter as follows:

1. Navigate to **Administration > Security**.
2. Select the **Authorization (Groups)** tab.
3. Create or edit an LDAP authorization realm that is associated with your authentication realm.
4. In the Create Authorization Realm dialog box, in the **Group Search Filter** field, specify the name of the group you want to allow.

   For example:

   ```
   (&(name=automation)(member={0}))
   ```

   This filter limits the import of groups from LDAP to just the *automation* group.

Creating Groups

Groups are logical containers that serve as a mechanism to grant permissions to multiple users; members automatically share a group's permissions. Permissions are granted to groups (or all users), not individual users. Additionally, when a group is assigned a role, its members are automatically assigned the role.

To create a group:

1. Navigate to **Administration > Security**.
2. Select the **Authorization (Groups)** tab.
3. Select **Groups** in the side menu.
4. Click the **Create Group** button.
5. Provide a name for the group.
6. Select an authorization realm.

   Groups are valid only for the selected authorization realm.

Authentication Realms and Users

Authentication realms determine a user's identity within an authorization realm. For example, a user listed in the LDAP realm may have different authorizations from those in the other realms.

For information on creating and configuring authentication realms and users, see the following topics:

- Creating Authentication Realms [page 124]
- Creating Users [page 125]
- Creating LDAP Authentication Realms [page 126]
- LDAP Authentication Configuration [page 127]
- Creating PKI Certificate Authentication Realms [page 128]
- PKI Certificate Authentication Configuration [page 133]

Creating Authentication Realms

To create an authentication realm:

1. Navigate to **Administration > Security**.
2. Select the **Authentication (Users)** tab.
3. On the side panel click the '+' button (**Create Authentication Realm**)
4. Enter a unique name, an optional description, and other basic parameters:

Allowed Login Attempts	Number of attempts allowed. A value of 0 means unlimited attempts.
Authorization Realm	Requires that the authorization realm was previously created.

Type	Select from the following types: • Internal Storage • LDAP • Single Sign On • PKI Certificate

Depending on the **Type** you select, additional parameters may be available as follows:

- For **Internal Storage** authentication realms, only basic parameters are available.
- For **LDAP** authentication realms, see Creating LDAP Authentication Realms [page 126].
- For **PKI Certificate** authentication realms, see Creating PKI Certificate Authentication Realms [page 128].
- For **Single Sign On** authentication realms, used with SBM, the following additional parameters are available:
  - **User Header Name**: The header name for an authorization token that is used by the SSO provider. For example, `ALFSSOAuthNToken`.
  - **Logout Url**: (Optional) When this parameter is specified, Deployment Automation redirects the logout request to the provided URL to inform the SSO provider that the session should be terminated.

    For details on configuring SSO, see the *Deployment Automation Integration Guide*.

Important: You cannot use SSO and PKI Certificate authentication realms in the same implementation; they are incompatible. If you want to use Smart Cards with SSO, you should configure this in SBM. See the SBM documentation for details.

Creating Users

You create or import users differently depending on the type of authentication realm.

Internal Security / Internal Storage Type Users

For Internal Storage type authentication realms, such as the default Internal Security authentication realm, you create users using the **Create New User** button.

When adding a new user, the user name and password is what the individual will use when logging into Deployment Automation. The user name will also be displayed when setting up additional security.

Once the new user has been successfully added to a group, you might need to configure additional permissions. This can happen when the new user is mapped to a group that has limited permissions.

Include an email ID for the user if you want them to be eligible to receive email notifications. See Configuring Email Notifications [page 144].

LDAP Users

Once LDAP configuration is complete, when a new user logs on using their LDAP credentials, they will be listed on the **Authentication Realms** pane. You should not manage user passwords or remove users from the list. If an active user is removed from Deployment Automation, they will still be able to log in to the server as long as their LDAP credentials are valid.

Single Signon Users

SBM solutions, such as Release Control, can use Single Sign-On (SSO) to access Deployment Automation functionality. The users are created in SBM, and when Deployment Automation is accessed the first time from SBM, the corresponding user is automatically created in Deployment Automation.

PKI Certificate Users

Users authenticated through PKI Certificate authentication are automatically created with the first login using the associated CA certificate. Users authenticated with PKI Certificate are automatically added to an authentication realm of PKI Certificate type.

They are assigned to a group based on the **Default security group** set in the system settings; they are not assigned to a group if the default security group is not configured. See System Settings [page 153].

Administrators can also add users to the PKI Certificate authentication realm without providing users' client certificates. The administrator can assign the users to groups manually and otherwise configure their permissions on a user by user basis. When users log in with their certificates, they are identified by their login name and their data is updated from the properties provided in the certificate, such as user name and email ID.

Administrators map certificate properties to user properties during configuration of the PKI Certificate authentication realm. See Creating PKI Certificate Authentication Realms [page 128].

Creating LDAP Authentication Realms

If you are using LDAP, you can import users and map them to the security system. If you create an `LDAP` type of authentication realm, you need to provide information about your LDAP installation as described in the following table.

For additional configuration requirements, see LDAP Authentication Configuration [page 127].

LDAP Authentication Realm Properties Table

Field	**Description**
Context Factory	Enter the context factory class to use to connect. This may vary depending upon your Java implementation. The default for Sun Java implementations: `com.sun.jndi.ldap.LdapCtxFactory`.
LDAP URL	URL to the LDAP server beginning with `ldap://` or `ldaps://`. Separate additional servers with spaces.

Field	Description
Use DN Pattern	Enter the user directory entry pattern; the name will be substituted in place of 0 in the pattern, such as `cn={0},ou=employees,dc=yourcompany,dc=com`.
User Search Base	Enter the user base directory to search for users, such as `ou=employees,dc=mydomain,dc=com`.
User Search Filter	Enter the LDAP filter expression to use when searching for a user's directory entry. The user name will be substituted in place of `{0}` in the pattern, such as `uid={0}`. If this is an attribute and not part of the user DN pattern, wrap the value in parenthesis, such as `ud=(0)`.
Search User Subtree	Search the subtree for users.
Search Connection DN	Enter the complete directory name to use when binding to LDAP for searches, such as `cn=admin,dc=mydomain,dc=com`. If not specified, an anonymous connection will be made. Required if the LDAP server cannot be anonymously accessed.
Search Connection Password	Enter the password to use when binding to LDAP for searches.
Name Attribute	Enter the attribute name that contains the user's name, as set in LDAP.
Email Attribute	Enter the attribute name that contains the user's email address, as set in LDAP.

LDAP Authentication Configuration

For more details on configuring your implementation to use LDAP authentication, see the following topics:

- LDAP User Permissions [page 127]
- Importing LDAP Users [page 128]
- Limiting LDAP Users by Group [page 128]

LDAP User Permissions

You can import users from existing LDAP systems into Deployment Automation-managed authentication realms.

Unless they are part of an LDAP authorization realm (group), valid LDAP users can log on but will have no permissions. To provide permissions, import the LDAP users and define their permissions before they log on.

Importing LDAP Users

To import LDAP users:

1. Navigate to **Administration > Security**.
2. Select the **Authentication (Users)** tab.
3. On the side menu select an LDAP realm and click the **Import User** button.
4. Enter the name of the user.

   If you enter a search filter in the **Username** field, the filter must be enclosed in parentheses.
5. Click **Import**.

Limiting LDAP Users by Group

To limit the LDAP authentication realm users by a specific group, you must use the LDAP filter as follows:

1. Navigate to **Administration > Security**.
2. Select the **Authentication (Users)** tab.
3. Create or edit an LDAP authentication realm.
4. In the Create Authentication Realm window, in the **User Search Filter** field specify the name of the group you want to use as a restriction.

   For example:

   ```
   (&(memberOf=CN=automation,OU=LDAPTest
   Groups,DC=ldaptest,DC=net)(sAMAccountName={0}))
   ```

   The first condition here, `(memberOf=CN=automation,OU=LDAPTest Groups,DC=ldaptest,DC=net)`, limits the LDAP authentication realm to only the users belonging to the *automation* group.

Creating PKI Certificate Authentication Realms

If you are using a public key infrastructure (PKI) to create and manage digital certificates for login security, you can configure Deployment Automation to use your organization's PKI certificates for user authentication. When this authentication is configured properly, user authentication happens automatically based on a PKI certificate installed in the user's web browser.

Important: You cannot use SSO and PKI Certificate authentication realms in the same implementation; they are incompatible. If you want to use Smart Cards with SSO, you should configure this in SBM. See the SBM documentation for details.

Multiple PKI Certificate authentication realms can be set up to support multiple CA certificates.

When you create a PKI Certificate authentication realm in Deployment Automation, you must provide information about your PKI Certificate installation as described in the following table.

For additional configuration requirements, see PKI Certificate Authentication Configuration [page 133].

PKI Certificate Authentication Realm Properties Table

Field	Description
Authorization Realm	Select Internal Security; PKI Certificate authentication realm always uses Internal Security for authorization
CA Certificate File	Specify the path, including the file, where you have stored the issuer's certificate information. For example: `D:\auth\ca.crt`
Username Attribute	Select either **Subject** or **Alternative Subject** and then select from the available attributes. This attribute should map to the value in your certificate that your certificate implementation uses for username. See PKI Certificate Parsing [page 132].
Email Attribute	Select either **Subject** or **Alternative Subject** and then select from the available attributes. This attribute should map to the value in your certificate that your certificate implementation uses for email ID. See PKI Certificate Parsing [page 132].
Full Name Attribute	Select either **Subject** or **Alternative Subject** and then select from the available attributes. This attribute should map to the value in your certificate that your certificate implementation uses for full name. See PKI Certificate Parsing [page 132].
Verify Revocation	Select this if you want to check to see if the user certificate has been revoked since it was last authenticated through the PKI certificate.
Revocation Strategy	If **Verify Revocation** is selected, select the revocation strategy you want to use. Options are as follows: • **Strict:** (Default) This strategy sets the certificate revocation status to *true* if any exception happens during verification process. For example, if an OCSP service URL is inaccessible or a downloaded CRL cannot be read or parsed, the certificate is considered revoked. • **Mild:** This strategy ignores exceptions described for strict verification and continues verification using the next verification point. If all verification points are ignored or revocation status is false for all of them then the resulting revocation status is *false*.

Field	Description

Revocation Source Type	If **Verify Revocation** is selected, select the revocation source type you want to use. Options are as follows:
	• **Internal:** Select this to verify using a certificate revocation list that you have identified to the Deployment Automation server. See Configuring Internal Revocation Verification [page 134]. This verifies the revocation status against OCSP URLs and CRL lists that are specified in the certificate file you have identified in the server configuration.
	The following CRL distribution point URL formats are supported:
	http(s)
	For example:
	`http://serverName:8080/crl.pem, https://serverName:8443/crl.der`
	ldap
	For example:
	`"ldap://ldapserverName:389/ou=Users, dc=da,dc=com?certificateRevocationList;binary"`
	• **External:** (Default) Select this to verify using the certificate revocation list in the location that you specify in the **CRL Distribution Point** field.
	The following CRL distribution point URL formats are supported:
	http(s)
	For example:
	`http://serverName:8080/crl.pem, https://serverName:8443/crl.der`
	ldap
	For example:
	`"ldap://ldapserverName:389/ou=Users dc=da,dc=com?certificateRevocationList;binary"`
	file
	`"file:///D:/StrongAuth/crl.der"`
	• **Both:** Select this to verify using both internal and external certificate revocation lists.

Field	Description
OCSP Server URL	If **Revocation Source Type** is set to *External* or *Both* and you want to use an Online Certificate Service Provider (OCSP) to verify certificate revocation, enter the URL that points to the service. For example: `http://ServerName:9999` **Note:** The Deployment Automation server and OCSP server must use the same time and time zone. Otherwise, depending on your selection for **Revocation Strategy**, the following will occur: • If **Strict**, the login will fail • If **Mild**, verification using OCSP will be skipped
CRL Distribution Point	If **Revocation Source Type** is set to *External* or *Both* and you want to point to a certificate revocation list file, specify the URL that points to your file. For example: `http://ServerName:8080/crl.file`
Use Revocation Cache	If **Verify Revocation** is selected, you can select this option to cache the results from the last revocation verification to avoid performance degradation for each login to the server. The following are cached for the number of hours specified in the **Revocation Cache Expiration Period**. • **Revocation status:** The last certification revocation status is stored in the cache and are used without additional verification for subsequent requests that have this certificate in their attributes. • **Certificate Revocation Lists (CRL):** Keeps downloaded lists in memory and stores them locally in the Deployment Automation profile in the var/cache/pki directory. For example: `C:\Users\username\.microfocus\da/var/cache/pki` Multiple simultaneous downloads of CRL lists from single endpoint is not allowed; if there is already an active download of a CRL list, all other requests are postponed until the download is completed. Both caches are cleared when the authentication realm is updated.
Revocation Cache Expiration Period	If **Use Revocation Cache** is selected, specify the time period in hours after which to refresh the cache. The default is 24 hours.

PKI Certificate Parsing

Every PKI certificate has two sections to hold information related to the person for whom this certificate was issued, **Subject** and **Alternative Subject Name**.

A **Subject** section example value is as follows:

```
Subject: C=UA, L=Kiev, O=microfocus, OU=DA, CN=sample name/
emailAddress=sample@email.com/street=sample street, SN=sample/name=sample
```

An **Alternative Subject Name** section example value is as follows:

```
X509v3 Subject Alternative Name: email:email@test.com, DNS:dns-sample, IP
Address:1.2.3.4, URI:uri test, Registered ID:1.2.3.4.5.6
```

Subject Attributes

The attributes that can be selected for parsing in the **Subject** field are as follows:

C

O

T

OU

CN

L

ST

SN

EmailAddress

DC

UID

STREET

SURNAME

GIVENNAME

INITIALS

GENERATION

UnstructuredAddress

UnstructuredName

UNIQUE_IDENTIFIER

DN_QUALIFIER

PSEUDONYM

POSTAL_ADDRESS

NAME_AT_BIRTH

COUNTRY_OF_CITIZENSHIP

COUNTRY_OF_RESIDENCE

GENDER

PLACE_OF_BIRTH

DATE_OF_BIRTH

POSTAL_CODE

BUSINESS_CATEGORY

TELEPHONE_NUMBER

NAME

Alternative Subject Attributes

The attributes that can be selected for parsing in the **Alternative Subject Name** section are as follows:

otherName

email

DNS

URI

IP Address

Registered ID

PKI Certificate Authentication Configuration

For more details on configuring your implementation to use PKI Certificate authentication, see the following topics:

- Configuring the Server to Support PKI Certificates [page 133]
- Importing CA Certificates [page 134]
- Configuring Internal Revocation Verification [page 134]
- Configuring Polling for Certificate Revocation Lists [page 135]
- Configuring PKI Certification for Agents [page 136]

Configuring the Server to Support PKI Certificates

Before you can use PKI certificates for authentication, you must configure the server to expect certificate-based authentication upon requested access.

To configure the server to support use of PKI Certificates for authentication:

1. Navigate to the Common Tomcat `conf` directory. For example:

   `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\conf`

2. Edit the following file: `server.xml`

3. Copy and paste the following `Connector` entry into the file to create a new `Connector` entry. For example, in the following, the port is `8643`. You must ensure you don't use the same port number that is used in an existing `Connector` entry and

you must ensure the `clientAuth` value is set to `true`, as shown in the following example.

```
<Connector port="8643" SSLEnabled="true"
  scheme="https" secure="true" sslProtocol="TLS"
  sslEnabledProtocols="TLSv1.2,TLSv1.1,TLSv1"
  maxHttpHeaderSize="8192"
  maxThreads="150" minSpareThreads="25"
  enableLookups="false" disableUploadTimeout="true"
  acceptCount="100"
  keystoreFile="conf/sample-ssl.jks"
  keystorePass="microfocus" keyAlias="tomcat"
  truststoreFile="conf/sample-ssl.jks" truststorePass="serena"
  clientAuth="true"  />
```

This tells Common Tomcat that the server can be accessed only if the user provides a certificate in the request. If you try logging into Deployment Automation and don't have a valid certificate in your browser, you will receive the following message:

```
Certificate-based authentication failed
```

Importing CA Certificates

CA certificates from your certification centers must be imported into the Java KeyStore file as a trusted certificate before using any user certificate signed by this CA certificate.

To add CA certificates:

1. Navigate to the Common Tomcat `conf` directory. For example: `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\conf`

2. Using an appropriate editor such as the Oracle Java keytool utility, unlock and edit the `jks` file that you specified in the connector `keystoreFile` parameter. For example: `sample-ssl.jks`

3. Import the CA certificate as a trusted certificate.

After you import a CA certificate, you will be able to select it in your web browser when you enter the secure URL pointing to Deployment Automation. For example:

```
https://MyServer:8643/da
```

Configuring Internal Revocation Verification

Before you can use the *Internal* PKI certificate revocation verification option, you must configure the server to support this. If you use the *External* PKI certificate revocation, you do not need to configure this. See Creating PKI Certificate Authentication Realms [page 128].

To configure the server for internal revocation verification:

1. Navigate to the Common Tomcat `conf` directory. For example: `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\conf`

2. Edit the following file: `server.xml`

3. Edit the `Connector` entry that you added or modified in Configuring the Server to Support PKI Certificates [page 133]. Add the attribute `crlFile=”../list.crl`, where `list.crl` is a file that contains your certificate revocation list, as shown in the following example.

   ```
   <Connector port="8643" SSLEnabled="true"
     scheme="https" secure="true" sslProtocol="TLS"
     sslEnabledProtocols="TLSv1.2,TLSv1.1,TLSv1"
     maxHttpHeaderSize="8192"
     maxThreads="150" minSpareThreads="25"
     enableLookups="false" disableUploadTimeout="true"
     acceptCount="100"
     keystoreFile="conf/sample-ssl.jks"
     keystorePass="microfocus" keyAlias="tomcat"
     truststoreFile="conf/sample-ssl.jks" truststorePass="serena"
     clientAuth="true" crlFile=”../list.crl”/>
   ```

The **Revocation Source Type** of *Internal* can now be used to verify against the specified certificate revocation list.

Configuring Polling for Certificate Revocation Lists

To handle large Certificate Revocation Lists (CRL), a CRL poller service can be run as part of the server startup. This service downloads the updated CRL for all enabled PKI Certificate authentication realms once a day. Downloaded lists are stored within the Deployment Automation profile folder. For example:

`C:\Users\`*`username`*`\.microfocus\da/var/cache/pki`

The CRL poller service downloads CRL lists only if the following are true:

- A PKI Certificate authentication realm is configured to use *External* or *Both* revocation verification source types
- The **CRL Distribution Point** property is configured for the PKI Certificate authentication realm
- The CRL poller flag is set to `true` in the server configuration file

If you use the *Internal* PKI certificate revocation, you do not need to configure this. See Creating PKI Certificate Authentication Realms [page 128].

To configure the poller service:

1. Navigate to the Common Tomcat `webapps\da\WEB-INF\classes` directory. For example: `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\WEB-INF\classes`

2. Edit the following file: `da_config.xml`

3. Add or edit the `pkiConfig` entries:

   `crlPollerEnabled`: If set to `true`, the poller service runs; if set to `false`, it doesn't `crlPollerStartHour`: Set to an integer value from `0` through `23` in 24 hour time format. If this is not set, the default is `0`.

In the following example, the poller is enabled and the start hour is `0`, which means it runs at midnight.

```
<pkiConfig>
    <crlPollerEnabled>true</crlPollerEnabled>
    <crlPollerStartHour>0</crlPollerStartHour>
</pkiConfig>
```

Configuring PKI Certification for Agents

For additional security, agents can be configured to authenticate through PKI certificates. Use the following procedure to configure each agent that you want to authenticate this way.

Note: No corresponding users are created in Deployment Automation when agents authenticate through PKI certificates. For information on user authentication through PKI certificates, see Creating Users [page 125].

To configure agents to authenticate using PKI certificates:

1. Navigate to the agent's `conf` directory. For example: `C:\Program Files (x86)\Micro Focus\Deployment Automation Agent/conf/sra.keystore`
2. Using an appropriate editor such as the Oracle Java `keytool` utility, remove the default key from the agent's `sra.keystore` file.
3. Add the client certificate issued by your CA for this particular agent to the agent's `sra.keystore` file.
4. Update the `conf/installed.properties` in the agent's `conf` directory to have the correct passwords for the `sra.keystore` file and for the key itself (`locked/agent.keystore.pwd` and `locked/agent.keystore.key.pwd` respectively). Passwords can be entered unencrypted, and will be encrypted the next time the agent starts.

Default Permissions

Default permissions can be set globally for all users for a product area, or for individual user groups within an area. By default, a product area's permissions are *not* enabled for any user or group, except for the `admin` user, which has all permissions for all role types granted. Use the Default Permissions page to set default permissions for the groups you create and for those included with the product.

Users added to a group inherit the group's default permissions.

Deployment Automation includes several role types mapped to product areas. Each role type has default roles with permissions that are appropriate for those roles. Groups added to a role inherit the role's permissions. For more information, see Role Configuration [page 116].

Note: Default permissions cannot be granted for system and UI security.

The permissions that are common to most product areas are shown in the following table.

Common Permissions Table

Permission	Description
Security	Enables users to change an item's security settings. For example, a user with this permission for agents can determine which users can view, configure, and set security for them.
Write	Enables users to add, change, and delete items. For example, a user with this permission for components can create a component.
Read	Enables users to view an item, but not change it or create another of its type. For example, a user with this permission for agents can see agents within the user interface, but cannot modify them or create another unless granted additional permissions.
Execute	Enables users to run processes associated with applications, components, environments, and resources. Users must also have read permission for an item before actually executing it.

Setting Default Permissions

To set default permissions for product areas:

1. Navigate to **Administration > Security**.
2. Select the **Default Permissions** tab.
3. On the side menu, select a product area.

   Selecting an area displays the permissions available for it.
4. Select the permissions you want to grant for all users or select the permissions you want set for each group. User-defined groups are configured independently. See Authorization Realms and Groups [page 121].

Adding Users to Groups

After you have defined your users and groups, you should add your users to groups.

To add users to groups:

1. Navigate to **Administration > Security**.
2. Select the **Authorization (Groups)** tab.
3. On the side menu select **Groups**.
4. Select the group to which you want to add a user and click the **Manage Group Members** icon.
5. Select the users you want to add and click **Update**.

Setting Role Membership by Product Area

After you have defined your roles, groups, and users, you then give groups or users roles in specific product areas.

- For most product areas, you set these in the **Security** tab for each specific product area and item.
- System Security and UI Security are set in the **Administration > Security** options of the same names.
- Environment and component security settings can be set at the direct object level and within the applications to which they are associated.
    - For component security, no matter whether you set the security on the component level or application component level, the settings are applied everywhere.
    - Environment security settings are handled as follows:
        - Upon initial association with an application, the security that is set for the environment is inherited by the application environment.
        - After an environment is associated with an application, if the security is changed directly on the environment, the setting for the application environment is not affected.
        - If the security is changed for the application environment, the setting at the direct environment level is not affected.

To add groups or users to roles:

1. Navigate to the **Security** tab for the product area for which you want to define security roles.

   For example, for agents or agent resources, navigate to **Management > Resources**. Select the **Pools** tab, select a pool, and select the **Security** tab.

   Exceptions are for Server Roles and Web UI Roles. For information on adding groups and users to these roles, see Server Roles and System Security [page 118] and Web UI Roles and UI Security [page 120] respectively.

2. Select a role and click the **Add Member** button on the right.
3. Select members from the eligible groups and users and then click **Add**.

Tokens

Tokens provide authorization for agents and users. Agents use tokens when performing process steps and communicating with the server and external services. Users can use tokens instead of supplying a user name and password in certain situations. Tokens can also be used with the command line client.

You can create tokens in addition to those included with the product.

To create a token:

1. Navigate to **Administration > Security**.

2. Select **Tokens** and click **Create**.
3. From the **User** drop-down list box, select the user who will use the token.
4. Specify the expiration date and time.

Tokens can be used immediately after being created.

Viewing Online Users

You can view all users who are online and invalidate sessions if needed.

To view online users:

1. Navigate to **Administration > Security**.
2. Select **Online Users**.

A list appears of all users who are currently online. You can:

- Click a user's email ID to contact the user.
- Easily see what authentication realm each user belongs to.
- See session start time and last access time for each user.
- Click the delete icon beside a user ID to invalidate that user's session.
- Refresh the page to see an updated view of online users.

Security Report Overview

Security reports provide information about user and group roles and privileges defined with the Deployment Automation security system. Following are some of the things you can do on the **Administration > Reports** page.

- Select the report you want to run.
- Print the selected report.
- Export the selected report. This saves the report to a file in HTML format.
- View the report listing.

For information on each report, see the following topics:

- Application Security Report [page 140]
- Component Security Report [page 140]
- Environment Security Report [page 141]
- Resource Security Report [page 142]

Application Security Report

The Application Security Report provides information about user and group roles and privileges defined for Deployment Automation-managed applications. Each report row represents an individual application. When selected, the report runs automatically for all applications.

For details on configuring and running this report, see the following topics:

- Application Security Fields [page 140]

Application Security Fields

The application security fields are shown in the following table.

Application Security Fields Table

Field	Description
Application	Name of the application
Clean Execution History	Users and groups who have privileges to clean execution history
Execute	Users and groups who have application execution privileges
Manage Snapshots	Users and groups who have snapshot management privileges
Read	Users and groups who can review information about the application but not change it
Run Component Processes	Users and groups who have component process execution privileges
Security	Users and groups who can define privileges for other users and groups
Write	Users and groups who can access and edit the application

Component Security Report

The Component Security Report provides information about user and group roles and privileges defined for components. Each report row represents an individual component. When selected, the report runs automatically for all components.

For details on configuring and running this report, see the following topics:

- Component Security Fields [page 140]

Component Security Fields

The component security fields are shown in the following table.

Component Security Fields Table

Field	Description
Component	Name of the component.
Execute	Users and groups who have component process execution privileges.
Security	Users and groups who can define privileges for other users and groups.
Read	Users and groups who can review information about the component but not change it.
Write	Users and groups who can access and edit the component.

Environment Security Report

The Environment Security Report provides information about user and group roles and privileges defined for environments. Each report row represents an individual environment. When selected, the report runs automatically for all environments.

For details on configuring and running this report, see the following topics:

- Environment Security Fields [page 141]

Environment Security Fields

The environment security fields are shown in the following table.

Environment Security Fields Table

Field	Description
Application	Name of the application
Clean Execution History	Users and groups who have privileges to clean execution history
Environment	Name of the environment
Execute	Users and groups who have execution privileges for the environment
Security	Users and groups who can define privileges for other users and groups
Read	Users and groups who can review information about the environment (but not change it)
Write	Users and groups who can access and edit the environment

Resource Security Report

The *Resource Security Report* provides information about user and group roles and privileges defined for resources. Each report row represents an individual resource. When selected, the report runs automatically for all resources.

For details on configuring and running this report, see the following topics:

- Resource Security Fields [page 142]

Resource Security Fields

The resource security fields are shown in the following table.

Resource Security Fields Table

Field	Description
Resource	Name of the resource.
Execute	Users and groups who have execution privileges for the resource.
Security	Users and groups who can define privileges for other users and groups.
Read	Users and groups who can review information about the resource but not change it.
Write	Users and groups who can access and edit the resource.

Chapter 9: System Administration

You can use the Administration System options of the user interface to access, create, and/or modify features, as documented in the following topics:

- Licenses [page 143]
- Configuring Email Notifications [page 144]
- Output Log [page 151]
- Adding System Properties [page 152]
- System Settings [page 153]
- ALF Events [page 155]
- Scheduling Version Cleanup [page 157]

Licenses

You can manage user licenses by adding or deleting licenses and assigning agents to them.

- Adding a License [page 143]
- Adding Agents to a License [page 143]
- Modifying or Deleting a License [page 144]

Adding a License

To add a license:

1. Navigate to **Administration > System**.
2. Select **Licenses**.
3. Click the **Add License** button.
4. Paste the license text supplied by Micro Focus into the **License** field.
5. (Optional) Add a description.
6. Click **Save**.

To see information about a license, click **Details**.

Adding Agents to a License

Agents can be assigned to licenses automatically or manually. This section explains how to add agents manually. To automatically add agents, ensure that the Automatic License Management check box on the System Settings pane is selected.

To add an agent to a license manually:

1. Click the **Assign Agents** icon beside the license in the license list.
2. Optional. You can filter the listed agents by entering search text into the text field.
3. Select the agent or agents you want to add to the license.
4. When you are finished, click **Add**.

Modifying or Deleting a License

To modify or update an existing license:

1. Click the ID of the license you want to change.
2. Click the **Edit License** button.
3. Edit the information shown in the **License** field.
4. Click **Save**.

To delete an existing license, click the Delete link for the selected license.

Configuring Email Notifications

Deployment Automation can send email notifications whenever user-defined trigger events occur. For example, notifications can be sent when a deployment finishes or an approval is required.

To send email notifications to users, you must do the following:

- Define notification templates or use the provided templates. These define the format of notification emails for different types of notifications.
- Define the notification schemes you want to use or use the default scheme. These define the triggering events and roles for the notifications.
- Include an email ID as part of the user details for each user who will receive email notifications. See Creating Users [page 125].
- Define the email server information in system settings. See System Settings [page 153].
- If you are sending notifications for applications or agents, select the notification scheme in the specific applications or agents for which you want to send notifications.
- When the triggering event defined in the notification scheme is invoked, an email notification is sent. The trigger happens when a process request either succeeds or fails and is sent only to users who have a role in the notification scheme.

A typical use of notifications is as follows:

- When an application process is run for an environment requiring approval, an email is triggered indicating that the process request must be approved.

- When the request is approved, the approval process completes and an email is triggered indicating that the application process is ready to run.
- After the application process completes, an email is triggered to indicate the success of failure of the process.

For details on configuring email notification schemes and templates, see the following topics:

- Configuring Notification Schemes [page 145]
- Creating Notification Templates [page 147]

Configuring Notification Schemes

Prerequisites:

Deployment Automation relies on LDAP and an associated email server to send notifications. You must set up LDAP prior to configuring notifications. Notification recipients are defined with the security system's LDAP integration. See Chapter 8: Security Administration [page 115].

Deployment Automation requires an external SMTP mail server to send notifications. For information about configuring a mail server, see Chapter 9: System Administration [page 143].

Notification schemes define what events should generate notifications and who should get them. When setting up notification schemes, to determine which users will receive notifications, you select the triggering events and the role. For example, it is common for an administrator or environment owner to be notified when a work item, as part of the approval process, has been generated.

You can configure the default scheme and set up additional schemes as needed for your implementation. You should select the template to use based on the reason for the notification and the recipients of the notification.

To configure a notification scheme:

1. Navigate to **Administration > System**.
2. Select **Notification Schemes**.
3. Select and edit an existing scheme, such as *Default Notification Scheme*, or click **Create** to create a scheme.
4. Set the available notification scheme options as follows.

Option	Description
Name	Specify a name for the scheme.
Description	Specify a description for the scheme.

Option	Description
Attach Logs	Select this to include a log file attachment for each target that shows the success or failure status of each step.
Attachment maximum before archival	Specify the number of logs to attach before the logs are compressed into an archive.

5. Click **Save**.

   To edit or add notification entries, select the notification scheme.

6. Edit an existing notification entry, such as *Process Failure* or *Process Success*, or click the **Create Notification Entry** button to create an entry.

7. Set the available notification entry options as follows.

Option	Description
Type	Select an event type for which to send notifications. The event type is typically determined by the type of recipient. For example, a deployment engineer would be interested in being notified about a failed deployment (Process Failure). Available types are: • Process Success • Process Failure • Process Started • Approval Completed • Approval Failed • Agent Goes Offline • Deployment Package Success • Deployment Package Failure • Deployment Package Started

Option	Description
Target	Select notification targets: - **Application:** Notifications are sent when the event type corresponds to an application. For example, the *Process Success* event, when paired with the *Application* target, would initiate a notification when a process, such as an application deployment, is successful. See Creating Applications [page 264]. - **Environment:** Notifications are sent when the event type corresponds to an environment. For example, the *Process Success* event would initiate a notification when a successful deployment has been run in an environment, such as SIT or PROD. - **Agent:** Notifications are sent when the event type corresponds to an agent. For example, the *Agent Goes Offline* event would initiate a notification when an agent goes offline. See Assigning Notification Schemes to Agents [page 311].
Role	Select a role to be notified. Each user who is assigned the selected role will receive an email. **Note:** The role corresponds to those set in **Role Configuration**. See Creating and Editing Roles [page 118].
Template Name	Select a notification template name. This selection is limited by the **Type** and **Target** selections.

8. Click **Save**.

Creating Notification Templates

Notification templates define the format of the information that you want to send in the email notifications for given situations. Default templates include the following:

- AgentGoesOffline
- ApplicationDeploymentFailure
- ApplicationDeploymentSuccess
- ApprovalCreated
- ApprovalFailed
- DeploymentReadied
- DeploymentPackageFailure
- DeploymentPackageStarted

- DeploymentPackageSuccess
- ProcessRequestStarted
- TaskCreated

Notification Templates are XML files located in the server profile's `conf/server/notification-template` file folder.

If the default notification templates do not suit your needs, you can create new ones.

To create a notification template:

1. Create an XML file.
2. Enter a script.

   Note: Notification templates support only Velocity reports. For more information on Velocity templates, see http://velocity.apache.org/engine/.

3. Save the file in the server profile's `conf/server/notification-template` file folder. The default locations on Windows and UNIX/Linux are as follows: `C:\Users\`*username*`\.microfocus\da`\server\notification-templates `/opt/MicroFocus/da/`*username*`/.microfocus/da`/server/notification-templates
4. Restart the server.

Related Topics

- Configuring Email Notifications [page 144]

Properties for Notification Templates

Properties from the various objects in Deployment Automation can be used in your notification templates. Some of the properties that are most useful in notification templates are given in the following tables.

Common Properties

Property name	Description	Type
fn	Object with helper functions	DeployTemplateHelper
snapshot	Object containing data about the application snapshot; nullable	Snapshot
environment	Object containing data about the environment where the process was executed	Environment

Property name	Description	Type
application	Object containing data about the application that contains the executed process; nullable	Application
externalUrl	URL of the Deployment Automation server	String

Process Execution Request Properties

Property name	Description	Type
workflowTrace	Trace of request execution; nullable	WorkflowTrace
analysis	Results of the execution, available if workflowTrace is not null; nullable	FilteredAnalysis
failureAnalysis	Results of the execution, filtered by only failed steps and available if workflowTrace is not null; nullable	FilteredFailureAnalysis
request	Object with request data	ApplicationProcessRequest
applicationProcess	Object with application process data	ApplicationProcess
versions	List of component versions	List<Version>
properties	Map with process properties; see the example of its values in Examples of Keys in the Properties Map [page 150]	Map<String, String>
user	User who submitted the request	User
resourceFactory	Provides access to other Deployment Automation objects	ResourceFactory

Approval Request Properties

Property name	Description	Type
approval	Approval request data	Approval
tasks	List of approval tasks	List<ApprovalTask>
appRequest	Object containing data about the process request; is null if compRequest is available (nullable)	ApplicationProcessRequest
compRequest	Object containing data about the process request; is null if appRequest is available (nullable)	ComponentProcessRequest
requester	User who submitted the request	User
scheduledDate	Data of approval	Date
processId	Process ID	String
requestType	Type of request	String

Examples of Keys in the Properties Map

Examples of keys that can be used in the properties map, or dictionary, for process execution requests are as follows:

- applicationProcess.id
- applicationProcess.name
- deploymentPackage.id
- deploymentPackage.name
- dpRequest.id
- dpRequest.user.name
- dpRequest.user.displayName
- dpRequest.user.id
- environment.id

- environment.name
- request.id
- request.user.name
- request.user.displayName
- request.user.id
- server.url

Note: The default templates use `getName()`, but the listed keys can also be used.

Basic Velocity Examples

The following examples show how the notification template properties may be used in common Velocity statements.

1. If-then-else statement in general form:

   ```
   #if (CONDITION)
   ***Statements/HTML to output
   #else
   ***Statements/HTML to output
   #end
   ```

   Checking null values:

   ```
   #if ($snapshot)
   <tr><td>Snapshot:</td>
   <td>$snapshot.getName()</td>
   </tr>
   #end
   ```

2. Creating a new variable; setting the new value to an existing variable:

   ```
   #set ($var = "odd")
   ```

3. Iterating collections:

   ```
   #foreach ($item in $collection)
   ***Statements/HTML to output
   #end
   ```

Output Log

You can download the Deployment Automation server log from within the user interface.

To display and download the log file:

1. Navigate to **Administration > System**.
2. Select **Log**.
3. Click **Download** to save the file.

   Note: You can download the file directly from the System options by clicking the **Download** link under Output Log.

The Deployment Automation server log file default location is: `Deployment Automation_root\var\log\deployserver.out`

Adding System Properties

You can set global system properties that are then available to be referenced by any process step.

To enter system properties:

1. Navigate to **Administration > System**.
2. Select **Properties**.
3. To set a single property, click **Add Property**.
   a. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Secure	Select this if the property should be secure. Secure properties are stored encrypted and displayed obscured in the Deployment Automation user interface.
Value	Value for the property that will be used by steps in a process

4. To set multiple properties, click **Batch Edit**.
   a. In the **Properties** box, enter multiple properties and their values on separate lines as property=value.

   For example, if you had properties `host` and `port`, you would enter them as follows:

```
host=myServer
```

```
port=8080
```

Note:

The following restrictions apply to batch edit:

- The **Secure** option cannot be set.
- The value cannot contain the "=" symbol.

You can now reference the property value in processes using `${p:system/<property name>}`.

System Settings

System settings let you specify values and preferences that work best for your implementation of Deployment Automation.

To set system settings:

1. Navigate to **Administration > System**.
2. Select **System Settings**. The system setting are shown on the page in view-only mode.
3. To preview the component versions that will be archived the next time an archive file is created, click the **Preview Version Cleanup** button.
4. To update the system settings, click the **Edit** button.

   The system settings are as follows:

Setting	Description
External Agent URL	URL for agents to access the Deployment Automation web UI.
External User URL	URL for users to access the Deployment Automation web UI.
Only Groups in Security Roles	When this is enabled, individual users cannot be granted permissions to anything. All permissions must be granted to groups.
Default security group	The security group that external users will be associated with when users are auto-registered.
Automatic Version Import Check Period (seconds)	The number of seconds between polls for new versions for all components with this option turned on. Requires a server restart to take effect. It is not recommend to set this to less than 15.

Setting	Description
Mail Server Host	Hostname of the mail server for notifications. Leave this blank to disable notifications.
Mail Server Port	SMTP port for email notifications.
Secure Mail Server Connection	Whether the SMTP connection should be secured.
Mail Server Sender Address	Address to use for the "from" address for email notifications.
Mail Server Username	Username to use for sending email notifications.
Mail Server Password	Password to use for sending email notifications.
Days to Keep Versions	Number of days to keep component versions. -1 will keep indefinitely.
Number of Versions to Keep	Number of versions to keep for each component. -1 will keep all.
Archive Path	Path to write a zip file containing the artifacts of each version cleaned up. If this is blank, archives will not be written.
Minimum Password Length	The minimum length of passwords allowed to be created.
Require Complex Passwords	This will validate new passwords created through the UI or command line interface to ensure the password is complex. To pass validation, it must use 2 of the 4 character classes: upper, lower,digits, and special. It must be as long or longer than the password minimum length.
Automation License Management	Whether the server should automatically assign new agents to open license slots.

Setting	Description
Allow conflicting name for agents	The server automatically add suffixes to conflicting agent names to allow an agent to connect to the server.
Enable Property Picker	Enables property indexing and property picker suggestions.
Allow Repeat Request	Enables the repeat request button.

ALF Events

Deployment Automation can emit Application Lifecycle Framework (ALF) events to a target endpoint server. ALF is an integrative layer that is used for custom integrations from SBM to Deployment Automation.

Connection Information

To use ALF events, you must specify the following connection information.

Field	Description
SBM Link	A full URL to an SBM event generation service or a root URL to the SBM server where the event manager is hosted. The link format should be as follows: `protocol://host:port` **Note:** To import the Deployment Automation event definitions into SBM Composer, use the `SRAALFEvent.wsdl` file provided with your Deployment Automation installation.
User Name	User name used to perform authorization on the SBM server.
User Password	Password used to perform authorization on the SBM server.

ALF Event Settings

The ALF events apply to the several categories, and there are corresponding sections in the user interface that you can expand or collapse individually. You can use the following control buttons to more easily work with the ALF event options:

- Set All
- Clear All

- Expand All
- Collapse All

The ALF events categories and their options are given in the following sections.

Agent

Event	Description
Status changed	If this option is selected, ALF Events will be emitted from Deployment Automation for agent status changes.

Application

Event	Description
Approval Required	If this option is selected, ALF Events will be emitted from Deployment Automation if approval is required for application processes.
Process Started	If this option is selected, ALF Events for application process start will be emitted from Deployment Automation.
Process Finished	If this option is selected, ALF Events for application process finish will be emitted from Deployment Automation.
Process Started	If this option is selected, ALF Events for application process start will be emitted from Deployment Automation.
Manual Task Created	If this option is selected, ALF Events for manual task creation will be emitted from Deployment Automation.
Snapshot Modified	If this option is selected, ALF Events for application snapshot deletion will be emitted from Deployment Automation.
Approval Required	If this option is selected, ALF Events will be emitted when application approval is required.
Manual Task Created	If this option is selected, ALF Events will be emitted when manual tasks are created for application processes.

Component

Event	Description
Approval Required	If this option is selected, ALF Events will be emitted from Deployment Automation if approval is required for component processes.

Event	Description
Version Created	If this option is selected, ALF Events for component version creation will be emitted from Deployment Automation.
Version Deleted	If this option is selected, ALF Events for component version deletion will be emitted from Deployment Automation.
Process Started	If this option is selected, ALF Events for component process start will be emitted from Deployment Automation.
Process Finished	If this option is selected, ALF Events for component process finish will be emitted from Deployment Automation.
Manual Task Created	If this option is selected, ALF Events will be emitted when manual tasks are created for component processes.

Environment

Event	Description
Approval Required	If this option is selected, ALF Events will be emitted when environment approval is required.

Global Process

Event	Description
Process Started	If this option is selected, ALF Events will be emitted from Deployment Automation when global processes start.
Process Finished	If this option is selected, ALF Events will be emitted from Deployment Automation when global processes finish.
Manual Task Created	If this option is selected, ALF Events will be emitted when manual tasks are created for global processes.

Scheduling Version Cleanup

You can create a schedule to automatically cleanup deployment versions.

1. Navigate to **Administration > Timeline**.
2. Click **Schedule** and select **Version Clean-up**.
3. Select properties for the schedule:
   - Name

- Start date
- Start time
- An optional recurring pattern, for example, daily

4. Click **Submit.** The scheduled clean-up is now displayed on the timeline.

Chapter 10: Agent Connectivity Status Checks

Agent connectivity can be checked periodically to ensure that the status in the user interface shows the true connectivity status. The following options control aspects of the status checks. You can optionally configure these options to fine-tune your implementation.

Agent-initiated:

- Agents send regular keepalive messages to the server to keep connections open.
- The server monitors the keepalive messages and updates agents' status accordingly.
- Agent relays track keepalive message statistics from connected agents and send them regularly to the server.
- Default: enabled
- **Server-initiated:**
  - The server can check the status of all agent endpoints in a single call.
  - The status check can be set to recur periodically.
  - Status check requests and responses are stored in the database for analysis.
  - Cleanup of the stored data can be scheduled to recur periodically.
  - In an active-active environment, the server that initiated the status request is the one that performs the subsequent analysis.
  - Minimum supported agent version for the agent and server initiated status check: 6.1.2
  - Default: disabled
- **User-initiated:**
  - In the **Agent / Pools** page of the user interface, when users click the **Test** icon (📶) beside an agent, the agent status is checked and reset in the UI accordingly.
  - Connectivity is tested for response through JMS protocol and then through JMS to HTTP protocol; messages are given for each.
  - The connectivity test timeout is configurable.

For more information, see the following topics:

- Configuring the Server-Initiated Agent Status Check [page 160]
- Configuring the Agent-Initiated Agent Status Check [page 161]

- Configuring the User-Initiated Agent Status Check [page 162]

Related Topics

Managing Agents and Agent Pools [page 310]

Configuring the Server-Initiated Agent Status Check

To configure the server-initiated agent status options:

1. Navigate to the directory or folder where the Deployment Automation server configuration is stored. For example:

   ```
   ..\Users\username\.microfocus\da\conf\server
   ```

2. Edit the `da_config.xml` file `keepAliveRollCallConfig` settings as needed. Descriptions of each setting are given in the following table.

Setting	Description
enabled	Turns on or off the scheduled run of keep alive roll calls. The default value is `false`.
pollingPeriodInMinutes	Specifies the keep alive roll call polling period in minutes. This is the time wait before performing the next keep alive roll call verification. The default value is `1`.
executePeriodInMinutes	Specifies how often to initiate scheduled keep alive roll calls, in minutes. The default value is `10`.
durationInSeconds	Specifies the time in seconds to wait for responses to the roll call. The default value is `20`.
cleanupEnabled	Turns on or off the scheduled run of cleanup for keep alive roll calls. The default value is `true`.
cleanupStartHour	The hour to run the cleanup for keep alive roll calls. The default value is `1`.
cleanupDaysToKeep	The amount of days to keep roll call requests and corresponding responses. The default value is `15`.

Example Server-Initiated Settings

Following are example settings for server-initiated agent status options in the `da_config.xml` file.

```
<keepAliveRollCallConfig>
 <enabled>true</enabled>
 <pollingPeriodInMinutes>1</pollingPeriodInMinutes>
 <executePeriodInMinutes>10</executePeriodInMinutes>
 <durationInSeconds>20</durationInSeconds>
 <cleanupEnabled>true</cleanupEnabled>
 <cleanupStartHour>1</cleanupStartHour>
 <cleanupDaysToKeep>15</cleanupDaysToKeep>
</keepAliveRollCallConfig>
```

Configuring the Agent-Initiated Agent Status Check

Keepalive messages are enabled by default on agents. These messages are used by ActiveMQ to keep the connection between an agent and a server (or relay) open. You can configure a server to respond to those messages and update agent status accordingly.

To configure an agent-initiated status check:

- Navigate to the directory or folder where the Deployment Automation agent is stored. For example:

  `..\Users\username\.microfocus\da\conf\server`

- Edit the `agentStatusTracking` settings in the `da_config.xml` file:

Setting	Description
enabled	Turns the agent status tracker on or off. Default: `false`
verificationPeriodInSec	Specifies how often to initiate scheduled agent status verifications (in seconds). Default: 60
bringingOnlineTimeoutInSec	Specifies the timeout period (in seconds) to bring an agent online after it is connected. If this timeout is reached and the agent is not online, the server re-sends a connectivity test and waits for another timeout. This process is repeated until the agent goes online. Default: 150
busyWhenQueueSizeIsHigherThen	Specifies the maximum average number of agents in a connection queue. If there are more agents than specified, agent status checks do not run until the average number goes below the specified value. Default: 20

Setting	Description
busyCountPeriodInMs	Specifies the number of milliseconds for the time period when the average is calculated for `busyWhenQueueSizeIsHigherThen`. Default: 2500

Example Agent Status Tracker Settings

Example settings for agent status check options in the `da_config.xml` file.

```
<agentStatusTracker>
          <enabled>true</enabled>
          <verificationPeriodInSec>60</verificationPeriodInSec>
                  <bringingOnlineTimeoutInSec>150</bringingOnlineTimeoutInSec>
          <busyWhenQueueSizeIsHigherThen>20</busyWhenQueueSizeIsHigherThen>
          <busyCountPeriodInMs>2500</busyCountPeriodInMs>
        </agentStatusTracker>
```

Note: Agent status checks do not run when:

- A server has more connected agents than allowed (`busyWhenQueueSizeIsHigherThen`)
- When an agent actor manager or the system is busy, see Chapter 12: Configuring Server Settings [page 173].

Note: Status checks for agents connected through the agent relay only work with version 6.2.2 or higher. Versions earlier than 6.2.2 cannot pass agent keepalive information to the server and are ignored.

Configuring the User-Initiated Agent Status Check

To configure the user-initiated agent status options:

1. Navigate to the directory or folder where the Deployment Automation server configuration is stored. For example:

   `..\Users\username\.microfocus\da\conf\server`

2. Edit the `da_config.xml` file `connectivityTestConfig` settings as needed. Descriptions of each setting are given in the following table.

Setting	Description
firstLevelTestTimeoutInSeconds	Specifies the time in seconds to wait for a response to the first level (JMS) connection status request. The default value is `10`.

Setting	Description
secondLevelTestTimeoutInSeconds	Specifies the time in seconds to wait for a response to the second level (JMS to HTTP) connection status request. The default value is 60.

Example User-Initiated Settings

Following are example settings for user-initiated agent status options in the da_config.xml file.

```
<connectivityTestConfig>
 <firstLevelTestTimeoutInSeconds>10
    </firstLevelTestTimeoutInSeconds>
 <secondLevelTestTimeoutInSeconds>60
    </secondLevelTestTimeoutInSeconds>
</connectivityTestConfig>
```

Chapter 11: Secure Socket Layer (SSL) Configuration

Secure Socket Layer (SSL) technology enables clients and servers to communicate securely by encrypting all communications. Data are encrypted before being sent and decrypted by the recipient so that communications cannot be deciphered or modified by third-parties.

Deployment Automation enables the server to communicate with its agents using SSL in two modes: unauthenticated and mutual authentication. Unauthenticated mode for HTTP and mutual authentication mode for JMS are optional; you can implement one without implementing the other, or implement both.

- Configuring Unauthenticated Mode for HTTP Communication [page 165]
- Configuring Mutual Authentication Mode [page 166]

Configuring Unauthenticated Mode for HTTP Communication

In *unauthenticated mode*, communication is encrypted but users do not have to authenticate or verify their credentials. Deployment Automation automatically uses this mode for JMS-based server/agent communication. You cannot turn this off.

SSL unauthenticated mode can also be used for HTTP communication. You can implement this mode for HTTP communication during server/agent/agent relay installation, or activate it afterward.

By default Deployment Automation uses this mode for its JMS-based server/agent communication on port 7918.

Important: Because agent relays do not automatically activate SSL security, you must turn it on during relay installation or before attempting to connect to the relay. Without SSL security active, agent relays cannot communicate with the server or remote agents.

To activate unauthenticated mode for HTTP:

1. Open the `installed.properties` file which is located in the `server_installation/conf/server` directory.

   The `installed.properties` file contains the properties that were set during installation.

2. Ensure that the `install.server.web.always.secure` property is set to `Y`.

3. Ensure that the `install.server.web.ip` property is set to the port the server should use for HTTPS requests.

4. Save the file and restart the server.

Configuring Mutual Authentication Mode

In *mutual authentication mode*, communications are encrypted as usual, but users are also required to authenticate themselves by providing digital certificates. A digital *certificate* is a cryptographically signed document intended to assure others as to the identity of the certificate's owner. Deployment Automation certificates are self-signed.

When mutual authentication mode is active, Deployment Automation uses it for JMS-based server/agent communication. In this mode, the Deployment Automation server provides a digital certificate to each agent, and each agent provides one to the server. This mode can be implemented during server/agent installation, or activated afterward.

To activate this mode, the Deployment Automation server provides a digital certificate to each local agent and agent relay, and each local agent and agent relay provides one to the server.

Agent relays, in addition to swapping certificates with the server, must swap certificates with the remote agents that will use the relay. Remote agents do not have to swap certificates with the server, just with the agent relay they will use to communicate with the server.

This mode can be implemented during installation or activated afterward.

Note: When using mutual authentication mode, you must turn it on for the server, agents, and agent relays, otherwise they will not be able to connect to one another. If one party uses mutual authentication mode, they must all use it.

Important: The server and agent properties *must* be set *prior* to configuring mutual authentication and exchanging keys.

To prepare for using mutual authentication, follow the following procedures in order:

- Property Settings for Mutual Authentication [page 166]
- Adding an Alias to an Agent [page 167]
- Adding an Alias to an Agent Relay [page 167]
- Mutual Authentication: Server and Agents [page 168]
- Mutual Authentication: Server, Agent Relay, and Agents [page 169]

Property Settings for Mutual Authentication

Before you can configure mutual authentication between a server and agents, or between a server, agent relay, and agents, you *must* set the following properties.

Table 1. Property Settings Required Before Exchanging Keys table

Property	Location	Value
If not set during install: `server.mutual_auth`	In the server installation directory's `installed.properties` file; for example: `.microfocus\da\conf\server\installed.properties`	`true`

Property	Location	Value
server.jms.mutualAuth	In the server installation directory's installed.properties file; for example: .microfocus\da\conf\server\installed.properties	true
For each agent: locked/agent.mutual_auth	In the agent installation directory's installed.properties file: [agent_installdir]\conf\agent\installed.properties	true
For each agent relay: agentrelay.jms_proxy.secure	In the agent relay installation directory's agentrelay.properties file: [agentrelay_installdir]\conf\agentrelay.properties	true
For each agent relay: agentrelay.jms_proxy.mutualAuth	In the agent relay installation directory's agentrelay.properties file: [agentrelay_installdir]\conf\agentrelay.properties	true

Adding an Alias to an Agent

Before you generate a certificate for an agent, you must add an alias to the agent.

To add an alias to an agent:

Note: You must have administrative privileges to perform this procedure.

1. Make sure your agent machine JAVA_HOME environment variable points to the directory where java is installed.
2. Open a shell and navigate to the agent install location \conf directory.
3. From within the agent's [install_location]\conf directory, run the following command:

   ```
   keytool -genkeypair -dname "cn=[alias_name]" -alias [alias_name]
    -keypass changeit -keystore sra.keystore -storepass changeit
    -keyalg RSA -keysize 1024 -validity 7305
   ```

   where [alias_name] is the name you give the agent alias; we recommend using the agent's name.
4. To check the result, run the command:

   ```
   keytool -list -keystore sra.keystore
   ```

Adding an Alias to an Agent Relay

Before you generate a certificate for an agent relay, you must add an alias to the agent relay.

To add an alias to an agent relay:

Note: You must have administrative privileges to perform this procedure.

1. Make sure your agent relay machine `JAVA_HOME` environment variable points to the directory where java is installed.

2. Open a shell and navigate to the agent relay install location `\conf\jms-relay` directory.

3. From within the agent relay's `[install_location]\conf\jms-relay` directory, run the following command:

   ```
   keytool -genkeypair -dname "cn=[alias_name]" -alias [alias_name]
    -keypass changeit -keystore agentrelay.keystore -storepass changeit
    -keyalg RSA -keysize 1024 -validity 7305
   ```

   where `[alias_name]` is the name you give the agent relay alias; we recommend using the agent relay's name.

4. To check the result, run the command:

   ```
   keytool -list -keystore agentrelay.keystore
   ```

Mutual Authentication: Server and Agents

Make sure your server and agents are not running before you start this configuration.

To configure mutual authentication between a server and agent:

1. Open a shell and navigate to the server install location `\conf` directory. For example:

   ```
   .microfocus\da\conf
   ```

2. Export the server key as a certificate by running:

   ```
   keytool -export -keystore server.keystore -storepass changeit
    -alias server -file server.crt
   ```

   Note: When you install a server, the alias server to be used for certificate key generation is assigned to the server.

   You should see the message:

   ```
   Certificate stored in file server.crt
   ```

3. Copy the exported `server.crt` (certificate file) to the agent `[install_location]\[agent_name]\conf` directory.

   where `[agent_name]` is the unique name you gave the agent during install.

4. From within the agent's `[install_location]\[agent_name]\conf` directory, import the `server.crt` file by running:

   ```
   keytool -import -keystore sra.keystore -storepass changeit -alias server
    -file server.crt -keypass changeit -noprompt
   ```

   You should see the message:

   ```
   Certificate was added to keystore
   ```

5. From within the agent's `[install_location]\[agent_name]\conf` directory, export the agent key as a certificate by running:

   ```
   keytool -export -keystore sra.keystore -storepass changeit
    -alias [agent_alias] -file [agent_name].crt
   ```

   The certificate is stored in the `[agent_name].crt` file.

   Important: Before you export an agent key, you *must* first add an alias to the agent (see Adding an Alias to an Agent [page 167]).

6. Copy the exported `[agent_name].crt` (certificate file) to the server installation location `\conf` directory.

7. From within the server's install location `.microfocus\da\conf` directory, import the `[agent_name].crt` file by running:

   ```
   keytool -import -keystore server.keystore -storepass changeit
    -alias [agent_alias] -file [agent_alias].crt -keypass changeit -noprompt
   ```

   You should see the message:

   ```
   Certificate was added to keystore
   ```

8. For additional agents, repeat from step 5.

9. Start the server and agents.

Mutual Authentication: Server, Agent Relay, and Agents

These instructions are for configuring mutual authentication for a server, agent relays, and agents that communicate with the server through agent relays. Make sure your server and agents are not running before you start this configuration.

To configure mutual authentication between a server, agent relay, and agents:

1. Open a shell and navigate to the server installation location `\conf` directory. For example:

   ```
   .microfocus\da\conf
   ```

2. Export the server key as a certificate by running:

```
keytool -export -keystore server.keystore -storepass changeit
 -alias server -file server.crt
```

You should see the message:

```
Certificate stored in file server.crt
```

3. Copy the exported `server.crt` (certificate file) to the agent relay `[install_location]\[agentrelay_name]\conf\jms-relay` directory.

   where `[agent-relay_name]` is the unique name you gave the agent relay during install.

4. From within the agent relay directory:

   ```
   [install_location]\[agent-relay_name]\conf\jms-relay
   ```

   import the `server.crt` file by running:

   ```
   keytool -import -keystore agentrelay.keystore -storepass changeit
    -alias server -file server.crt -keypass changeit -noprompt
   ```

   You should see the message:

   ```
   Certificate was added to keystore
   ```

   Important: Before you can export an agent relay key, you must first add an alias to the agent relay. See Adding an Alias to an Agent Relay [page 167].

5. From within the agent relay directory:

   ```
   [install_location]\[agent-relay_name]\conf\jms-relay
   ```

   export the agent relay key as a certificate by running:

   ```
   keytool -export -keystore agentrelay.keystore -storepass changeit
    -alias [agent-relay_alias] -file [agent-relay_name].crt
   ```

   You should see the message:

   ```
    Certificate is stored in file [agent-relay_name].crt
   ```

6. Copy the exported `[agent-relay_name].crt` (certificate file) to the server installation location `.microfocus\da\conf` directory.

7. From within the server's install location `.microfocus\da\conf` directory, import the `[agent-relay_name].crt` file by running:

   ```
   keytool -import -keystore server.keystore -storepass changeit
    -alias [agent-relay_alias] -file [agent-relay_name].crt
    -keypass changeit -noprompt
   ```

You should see the message:

```
Certificate was added to keystore
```

8. For an agent that is configured to connect to the agent relay, copy the exported `[agent-relay_name].crt` (certificate file) to the agent `[install_location]\[agent_name]\conf` directory.

   where `[agent_name]` is the unique name you gave the agent during install.

9. From within the agent's `[install_location]\[agent_name]\conf` directory, import the `[agent-relay_name].crt` file by running:

   ```
   keytool -import -keystore sra.keystore -storepass changeit
    -alias [agent-relay_alias] -file [agent-relay].crt
    -keypass changeit -noprompt
   ```

   You should see the message:

   ```
   Certificate was added to keystore
   ```

10. From within the agent's `[install_location]\[agent_name]\conf` directory, export the agent key as a certificate by running:

    ```
    keytool -export -keystore sra.keystore -storepass changeit
     -alias [agent_alias] -file [agent_name].crt
    ```

    You should see the message:

    ```
     Certificate is stored in file [agent_name].crt
    ```

    Important: Before you can export an agent's certificate key, you must first add an alias to the agent. See Adding an Alias to an Agent [page 167].

11. Copy the exported `[agent_name].crt` (certificate file) to the agent relay `[install_location]\[agent-relay_name]\conf\jms-relay\` directory.

12. From within the agent relay `[install_location]\[agent-relay_name]\conf\jms-relay` directory, import the `[agent_name].crt` file by running:

    ```
    keytool -import -keystore agentrelay.keystore -storepass changeit
     -alias [agent_alias] -file [agent_name].crt -keypass changeit -noprompt
    ```

    You should see the message:

    ```
    Certificate was added to keystore
    ```

13. To configure another agent that communicates with the server through this agent relay, repeat from step 10.

    Attention: For each agent, make sure you change the name of the `-alias` argument `[agent_alias]`, and the `-file` argument `[agent_name]`.

14. Restart the server, agent relay, and agents.

Chapter 12: Configuring Server Settings

Several default settings for options in Deployment Automation are controlled in the server configuration file. This section gives information on settings you may want to change in your implementation. For details on reconfiguring settings you designated during the installation of Deployment Automation, see Reconfiguring Server Installation Settings [page 59].

Tip: If there is a particular behavior or timeout value you want to change that isn't documented here, try searching the Knowledgebase.

For details on configuring server settings, see the following topics:

- Configuring Component Version Import [page 173]
- Configuring the Auto-select for Export Option [page 174]
- Configuring the Default for Only Changed Versions Option [page 175]
- Configuring Agent Connectivity Settings [page 175]

Configuring Component Version Import

Component versions are imported into Deployment Automation from various source configuration tools. This topic describes the way this works and tells how to configure it. For details on importing component versions, see Component Versions [page 231].

When a component version import is requested, the request is added to the queue and a message is displayed in the user interface that the request was successful. Requests run asynchronously on the server in the order they were placed in the queue. Incoming tasks are queued up until the queue is full.

CAUTION:

A successful request does not necessarily mean that the component version import will be successful.

The **Versions** page is automatically refreshed every 15 seconds following submission of a version import. You can also manually refresh the page using the **Refresh Page** button.

The maximum integration task count parameter controls the capacity of the queue of submitted versions for each component. The default value is 1. This parameter affects both automatic and manual version import for all source configuration types.

To change the capacity of the component version import queue:

1. Navigate to the Deployment Automation profile directory. For example:

   ```
   ..\Users\myUser\.microfocus\da\conf\server
   ```

2. Edit the `da_config.xml` file.

3. Update the `<maxIntegrationTaskCount>` parameter as follows, where # is a value 1 or above:

   ```
   <componentIntegrationTaskConfig>
   <maxIntegrationTaskCount>#</maxIntegrationTaskCount>
   </componentIntegrationTaskConfig>
   ```

   CAUTION:

   When changing this number, be aware of the following:

   - Do not set it to 0. If you do so, it will prevent versions from being imported.
   - If you set this to a number higher than 1, you should ensure that your system is capable of handling the number of version imports allowed.

Configuring the Auto-select for Export Option

When you create or edit an environment, you can select the **Auto-select for Export** option, which causes the environment to appear in the list of environments to export in the replication export wizard if you later export using replication export. By default, the **Auto-select for Export** option is selected. If you prefer the option to be de-selected by default, you can configure this using the following procedure.

The **Auto-select for Export** option can be selected or de-selected in the environment details regardless of the setting of this flag. This simply controls the default setting. See Creating Environments [page 298].

Tip: If you want most or all environments to be exported during replication export, you should leave this flag set to true, as that will make the selection faster when you go through the replication export wizard. However, to export only a few environments with any extra choices being explicitly added to the selection list in the export wizard, you should set this flag to false. For example, to select only pre-production and production environments for replication, and not select others such as development and testing environments, setting this to false may be the better choice for you.

To configure the Auto-select for Export option default setting:

1. Navigate to the directory the Deployment Automation server `da\WEB-INF\classes` directory. For example: `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\WEB-INF\classes`
2. Edit the following file: `da_config.xml`
3. Change the environment export flag default value, `<envAuto-select for ExportFlagDefValue>`, to false as follows:

   ```
   <importExportConfig>
        <ignoreImportErrors>false</ignoreImportErrors>
        <exportInactive>false</exportInactive>
        <envAuto-select for ExportFlagDefValue>false</envAuto-select for ExportFlagD
   </importExportConfig>
   ```

4. Save your changes.
5. Restart the Deployment Automation server.

The **Auto-select for Export** option for environments will now be de-selected by default.

Configuring the Default for Only Changed Versions Option

When you create an application process, you can select the **Default for Only Changed Versions** option, which subsequently controls the setting of the **Only Changed Versions** option in the Run Application Process wizard. By default, the **Default for Only Changed Versions** option is selected. If you prefer the option to be de-selected by default, you can configure this using the following procedure.

The **Default for Only Changed Versions** option can be selected or de-selected in the application process details regardless of the setting of this flag. This simply controls the default setting. See Creating Application Processes [page 275].

To configure the Default for Only Changed Versions option default setting:

1. Navigate to the directory the Deployment Automation server `da\WEB-INF\classes` directory. For example: `C:\Program Files\Micro Focus\common\tomcat\8.5\webapps\da\WEB-INF\classes`
2. Edit the following file: `da_config.xml`
3. Change the `systemConfig` `applicationProcessDefaultForOnlyChangedVersions` value to `false` as follows:

```
<systemConfig>
  <applicationProcessDefaultForOnlyChangedVersions>false
  </applicationProcessDefaultForOnlyChangedVersions>
</systemConfig>
```

4. Save your changes.
5. Restart the Deployment Automation server.

The **Default for Only Changed Versions** option in application process details will now be de-selected by default.

Configuring Agent Connectivity Settings

When agents are connected to a server, every request goes to an actor pool. All requests are spread evenly between actors in a pool. You can configure:

- How many actors are available for processing agent requests.
- When a server should stop checking agent connectivity status due to a large number of requests, see Chapter 10: Agent Connectivity Status Checks [page 159].

To configure agent connectivity settings:

1. Navigate to the directory where the Deployment Automation server configuration is stored, for example: `..\Users\username\.microfocus\da\conf\server`

2. Edit the `agentActorManager` settings in `da_config.xml`:

Setting	Description
threadPoolSizeMultiplicationCoefficient	Specifies a multiplication coefficient for an actor pool. The number of logical CPUs is multiplied by this coefficient to calculate the number of actors in the actor pool. For example, if the coefficient is 0.75 and the server has 4 logical CPUs, the actor pool size is 3. Default: 0.75
busyWhenQueueSizeIsHigherThen	Specifies the maximum average number of connectivity tasks that can be processed during a verification period. If there are more connectivity tasks then specified, the agent manager is considered to be busy and agent status checks are not performed. For details see Chapter 10: Agent Connectivity Status Checks [page 159]. Default: 100
busyWhenConnectedMoreThen	Specifies the maximum average number of connectivity tasks that can be processed during a verification period. If there are more connectivity tasks then specified, the agent manager is considered to be busy and agent status checks are not performed. For details see Chapter 10: Agent Connectivity Status Checks [page 159]. Default: 500

Example Agent Connectivity Settings

Example settings for agent connectivity options in `da_config.xml`.

```
<agentActorManager>
                <threadPoolSizeMultiplicationCoefficient>0.75
</threadPoolSizeMultiplicationCoefficient>
                <busyWhenQueueSizeIsHigherThen>100</busyWhenQueueSizeIsHigherThen
                <busyWhenConnectedMoreThen>500</busyWhenConnectedMoreThen>
        </agentActorManager>
```

Chapter 13: Artifact Repository

The following topics explain the architecture of the artifact repository in Deployment Automation.

Artifact Repository Overview

Artifact versioning in Deployment Automation is handled by CodeStation. You can direct Deployment Automation to introduce artifacts into CodeStation from the file system or from external source control tools that you identify when you select the **Source Config Type** for a component.

CodeStation artifacts represent deployable items such as files, images, databases, configuration materials, or anything else associated with a software project. By default, these are stored in the `var` subdirectory in the Deployment Automation server installation directory.

Deployment Automation's secure and tamper-proof artifact repository ensures that deployed components are identical to those tested in pre-production environments. Without the repository, artifacts would have to be pulled from network shares or some other system, increasing both security risks and the potential for error.

The artifact repository uses content addressable storage to maximize efficiency while minimizing disk use. The repository tracks file versions and maintains a complete history for all components. Maximizing efficiency is important, since the artifact repository stores files that are much larger than source files.

Association of files with Components is built into the system. Without any configuration, each Component gets its own area of the repository for its files. There is no chance of confusion or mix-up of files to Components. And, each Component Package is mapped to a specific set of files and versions corresponding to the Component.

The artifact repository comes with a client application that provides remote access to the repository. Using the client, the user can add/modify files, create Packages, retrieve files, as well as view the history of changes.

The client application provides a file transfer capability that can be used to deliver files to target servers during deployments. This built-in transfer mechanism verifies the integrity of all transferred files against their expected cryptographic signatures, thus guaranteeing that files have not been corrupted during transmission or tampered with during storage.

In addition to the client application, the artifact repository exposes REST-based web services. These services are used to build integrations between build systems such as AnthillPro and Deployment Automation. Such integrations automatically place the artifacts produced by the build process in the artifact repository, thus making the artifacts available for deployment.

In an enterprise environment, the default installation might not be ideal. See Relocating the Repository [page 178] for a discussion about enterprise options.

Relocating the Repository

For enterprise implementations, due to storage requirements, you may need to relocate your CodeStation files so that they are stored on a different machine than your Deployment Automation server. A good formula for determining CodeStation storage requirements is: `average artifact size * number of versions imported per day * average number of days before cleanup`.

File storage is organized as follows on a default Deployment Automation server installation:

- Execution log files are located in `.microfocus\da\logs` folder
- CodeStation respository files are located in `.microfocus\da\var\repository`
- Plugin files are located in `.microfocus\da\var\plugins`
- This folder `.microfocus\da\var\db` contains Derby database files if Derby is used.

Relocate the CodeStation files as follows:

1. Stop the server before moving the folder location.
2. To relocate log files, you must relocate the `\logs` folder.
3. You can either relocate the whole `var` folder, or if you want to relocate only the repository files, you can relocate just the `\var\repository` folder.
4. Use the following to configure your network storage in an optimal way:
   - This data should be stored on robust network storage that is regularly synchronized with an off-site disaster recovery facility.
   - The Deployment Automation server should have a fast network connection to storage (agents do not need access to the storage location).
   - In Unix environments, you can use *symbolic links* from the `var` subdirectory to network storage.
   - On Windows platforms there are several options for redirecting logs and artifacts, including `mklink` (supported in Windows 7 and later).
   - Distributed teams should also take advantage of location-specific CodeStation proxies to improve performance and lower WAN usage.

Chapter 14: Managing Execution History

Typically you should keep your execution history, since that is an audit trail of your system. However, you may want to clean up execution history that is no longer needed in your system. The **Execution History Cleanup** action enables you to do this.

For details, see the following:

- Pre-Cleanup [page 179]
- Rules of Execution History Cleanup [page 180]
- Running Execution History Cleanup [page 181]
- Execution History Cleanup Audit [page 181]
- Post-Cleanup [page 181]

Pre-Cleanup

Before running the execution history cleanup, consider the following:

- While running, execution history cleanup can effect Deployment Automation performance and could result in process execution delays. Therefore, the cleanup should be planned ahead of time and run at a low or no activity time.
- To avoid causing confusion when execution history data has been deleted, you should notify users to let them know when the cleanup is being run.
- Backup your Deployment Automation database in case of system failure while the cleanup is running.

Before running the execution history cleanup, you must ensure the proper permissions are set as follows:

- In **Administration > Security > Role Configuration > Web UI**, ensure that the **Clean Execution History** permission is selected for the role who will run the cleanup, such as Admin, for each of the following product areas:
  - Web UI
  - Applications
  - Environments
- Ensure that the users or groups who will run the cleanup have membership in the role.

See Role Configuration [page 116].

Note: In **Administration > Security > Default Permissions**, you can set **Clean Execution History** permission for **All Users Permissions** and **Groups Permissions**. This will cause this permission to be set whenever new users or groups are created. However, since the cleanup should be run only by select users in a controlled situation, you should not typically add this permission as a default.

Rules of Execution History Cleanup

Whoever runs the execution history cleanup must have the **Clean Execution History** permission through their role. This includes the following product areas:

- **Web UI**: to run the **Execution History Cleanup** action
- **Applications**: to select specific applications in the **Execution History Cleanup** wizard
- **Environments**: to clean up process requests for specific environments

Note: The **Execution History Cleanup** action is available by default only for the root administrative user defined in your installation.

An execution history cleanup request will not delete any history in the following situations:

- A process request that is not started yet
- A process request that is currently running
- A process request that is waiting for approval
- An application process request that has at least one component process request that is part of the resource inventory
- A standalone component process request that is part of the resource inventory
- An application process request that is part of the latest desired inventory

A cleanup request is initiated for each process request to be cleaned up. If a cleanup request cannot be completed for any reason, the data is rolled back to its original state for just that one cleanup request. For each process request that is eligible to be deleted, the following information is deleted:

- Logs
- Attachments
- Approvals
- Workflow traces
- Tasks
- Basic process requests

- Deployment process requests for application process requests (not deleted if the deployment process request has more than one application process request)
- Pipeline execution requests
- Process request property contexts
- Process request property sheets

Running Execution History Cleanup

A wizard leads you through the cleanup of your execution history.

To cleanup execution history:

1. Navigate to **Administration > System**.
2. Click **Run Execution History Cleanup**.
3. Select the applications that you want to cleanup and click **Next**.
4. Enter or select the start and end dates and times for the execution history data to be cleaned.
5. Click **Submit**.

Detailed results of the execution history cleanup are shown. Select given tabs to see more information on any failed application or component requests. p>

Note: By default, a sweep job is automatically run at set intervals to cancel or kill cleanup requests that have hung. The settings for the sweep can be configured, including whether or not to run the job at all. For details, refer to Knowledgebase item S141764.

Execution History Cleanup Audit

After running execution history cleanup, audit information is available for the cleanup requests.

To access the audit:

- Navigate to **Administration > System** and select **Execution History Cleanup**.

In this page you can do the following:

- View a summary list of **Execution History Cleanup** request executions.
- Click the eye icon beside a request to view more detail.
- Click the **Run Execution History Cleanup** button in the upper right of the pane to initiate another cleanup.

Post-Cleanup

Since the execution history cleanup action is not something to be run without prior planning and communication to users, it is recommended that after running the cleanup that you consider deselecting the **Clean Execution History** permissions that you selected in Pre-Cleanup [page 179]. This will prevent any users from running an unplanned cleanup.

Chapter 15: Data Collection and Check for Update

Allow Data Collection

Micro Focus can optionally collect statistics on your product usage of Deployment Automation. Users are prompted for permission to collect this data at their first login to a new or upgraded installation of Deployment Automation. If they agree there or in their user profile preferences, Micro Focus collects data on the use of various Deployment Automation objects. This option can be selected only by users with the **Data collection** permission.

Allow Check for Update

Micro Focus can optionally detect and suggest server, agent, and plugin upgrades. Users are prompted for permission to allow check for update at their first login to a new or upgraded installation of Deployment Automation. If they agree there or in their user profile preferences, they are alerted of available updates. This option can be selected only by users with the **Check for upgrade** permission.

For information on setting these permissions, see Server Roles and System Security [page 118].

See the following topics for more information about the data collected.

- Data Collected [page 183]
- Data Collection FAQ [page 185]
- Upgrade Eligibility Information Collected [page 186]
- Suggestions for Upgrade [page 187]
- Checking for Updates Manually [page 189]

Data Collected

The following statistics are collected for each user who agrees to and is allowed to have data collection.

Information is encrypted before it is sent, and the receiving Micro Focus server will reject any unencrypted data.

Name	Description
Server Version	The version of the Deployment Automation server software in use by this user

Name	Description
Server URL	The external URL of the Deployment Automation server in use by this user
Server MAC address	The MAC address of the Deployment Automation server in use by this user
IP address	The IP address of the Deployment Automation server in use by this user
Server ID	The UUID specific to this Deployment Automation server
User ID	The UUID of the user who authorized the data collection
Number of logins	The number of times each user whose ID is provided has logged into the system
Start date	The start date and time, to the second, of the period for which data is collected
End date	The end date and time, to the second, of the period for which data is collected
Number of comp process requests	The number of component process requests made on this Deployment Automation server that this user is authorized to view
Number of app process requests	The number of application process requests made on this Deployment Automation server that this user is authorized to view
Number of global process requests	The number of global process requests on this Deployment Automation server that this user is authorized to view
Number of agents	The number of agents registered to this Deployment Automation server that this user is authorized to view
Min Agent Version	The minimum version of any agent on this Deployment Automation server for which this user is authorized
Max Agent Version	The maximum version of any agent on this Deployment Automation server for which this user is authorized

Name	Description
Components	IDs of components that this user is authorized to view and names of the plugins used in these components
Community	This is set to true if the server uses only a community license and has no commercial licenses installed. Otherwise it is set to false.

Data Collection FAQ

Following are some frequently asked questions on the data collected.

1. **How does Micro Focus collect information?**

   If authorized through the data collection option, your Deployment Automation server automatically collects Deployment Automation usage data and sends that information to the Micro Focus data collection server. It also suggests server and agent upgrades for which your implementation is eligible. See Suggestions for Upgrade [page 187].

2. **What information does it collect?**

   For information on user statistics collected, see Data Collected [page 183]. For information on global server and agent upgrade eligibility, see Upgrade Eligibility Information Collected [page 186].

3. **Who can see the data that is collected?**

   This server is stored on a Micro Focus data collection server and is visible only to authorized Micro Focus employees.

4. **Will it collect information if I say No?**

   Micro Focus only collects statistics for users who authorize it to do so by clicking **Yes** when prompted at login, or by selecting the **Allow Data Collection** check box in their user profile **Preferences**. However, if any user agrees to data collection, Micro Focus will detect server and agent data for upgrade eligibility suggestions.

5. **How often will it collect information?**

   Data is collected for each user's usage and sent to the data collection server once a week.

6. **Will it slow down my server?**

   Since the data collection happens only once a week and sends only a limited amount of data, you should notice no performance degradation on your server as a result of this.

7. **Can I turn it off if I said Yes?**

   If you click **Yes**, then change your mind later, you can turn it off at any time through **Preferences** in your user profile.

8. **Can I turn off the reminder?**

If you click **Ask again later**, you will be reminded weekly. To turn off the reminder, you can simply answer **No** the next time you are reminded.

9. **Will it save my personal information?**

   No, the data is sent anonymously and contains encoded information. Although each user must authorize the sending of the data on his or her behalf, the data sent doesn't contain any identifying information that could be used to tie the sessions back to a particular user ID or IP address.

10. **Why does it collect my IP address?**

    The IP address of your Deployment Automation server may be used by Micro Focus for statistical roll-up of information over time to determine numbers of servers being used as single servers or active-active configurations.

11. **Is it secure?**

    Yes, the connection sends data to a secure Micro Focus server. It collects information about the use of Deployment Automation only for the particular users who authorized it. It does not collect or send usage information for individual Deployment Automation users other than the ones who have authorized it.

12. **What will Micro Focus use it for?**

    Micro Focus will use the information to help improve Deployment Automation. By understanding how our users work with Deployment Automation, we can prioritize the most-used areas of the product for enhancements and maintenance.

13. **Will Micro Focus sell the data?**

    Micro Focus will not sell the data. It will be used only as stated.

Upgrade Eligibility Information Collected

The following upgrade eligibility information is collected globally for the server and all agents and plugins associated with it.

Information is encrypted before it is sent, and the receiving Micro Focus server will reject any unencrypted data.

Name	Description
Server URL	The external URL of the Deployment Automation server in use by this user
Server ID	The UUID specific to this Deployment Automation server
Server Version	The version of the Deployment Automation server software in use by this user
Community	This is set to true if the server uses only a community license and has no commercial licenses installed. Otherwise it is set to false.

Name	Description
Java Version	Version of Java being used by the Deployment Automation server
Platform	Operating system on which the Deployment Automation server is running, including name, version, and architecture
Database	Database used by the Deployment Automation server, including name and version
Agent	All agent information for each agent associated with the server, including the following: • Agent Version • Agent ID • Operating System Name, Architecture, and Version • Java Version
Plugin	All plugin information for each agent associated with the server, including the following: • Plugin Name • Plugin Version

Suggestions for Upgrade

If given permission, Micro Focus informs you when your server and agents are eligible for upgrade.

When upgrade eligibility is detected, in the dashboard at the top of the Deployment Automation page you will see a message indicator 🔔. Click this to see the upgrade suggestions for your server, agent, and plugin versions. Suggestions are shown in an overlay in separate sections as follows:

- Server Upgrade Messages [page 187]
- Agent Upgrade Messages [page 188]
- Plugin Upgrade Messages [page 188]

Server Upgrade Messages

Server upgrade message options are as follows:

- **Download new version**: If you click this, you will be redirected to a download page where you can download and upgrade the server.

- **Ask again later**: Click this to temporarily dismiss the messages. You will receive the messages again a week later.
- **Never for this version**: If you click this, you will not be notified again about the suggested version. However, when a new version becomes available, you will be notified.
- **More info**: Click this to be directed to the documentation for this option.

Messages indicate how critical the suggested upgrade is and include color coding as follows:

- **critical (red)**
- **important (yellow)**
- **recommended (green)**
- **optional (grey)**

Agent Upgrade Messages

Agent upgrade options are as follows:

- **Upgrade**: If you click this, you will be redirected to the Deployment Automation **Agents / Pools** page for where you can upgrade the agents.
- **Ask again later**: Click this to temporarily dismiss the messages. You will receive the messages again a week later.
- **Never for this version**: If you click this, you will not be notified again about the suggested version. However, when a new version becomes available, you will be notified.
- **More info**: Click this to be directed to the documentation for this option.

Messages indicate how critical the suggested upgrade is and include color coding as follows:

- **critical (red)**: At least one agent requires a critical upgrade
- **important (yellow)**: At least one agent requires an important upgrade and no agents require critical upgrades
- **recommended (green)**: At least one agent requires a recommended upgrade and no agents require critical or important upgrades
- **optional (grey)**: At least one agent requires an optional upgrade and no other upgrade types are required

The **Management > Resources > Agents / Pools** page shows upgrade status for agents if you have agreed to Check for Upgrade and have the permission. Beside each agent with eligible upgrades, the upgrade status is shown.

For the procedure to upgrade agents, see Upgrading Agents [page 314].

Plugin Upgrade Messages

Plugin upgrade options are as follows:

- **Download**: If you click this, you will be redirected to the Deployment Automation **Plugins** page for where you can upgrade the plugins.

- **Ask again later**: Click this to temporarily dismiss the messages. You will receive the messages again a week later.

- **Never for this version**: If you click this, you will not be notified again about the suggested version. However, when a new version becomes available, you will be notified.

- **More info**: Click this to be directed to the documentation for this option.

The **Administration > Automation > Plugins** page shows **Download** links for plugins eligible for upgrade if you have agreed to Check for Upgrade and have the permission. After you have upgraded the plugins, the **Download** links no longer appear.

For the procedure to upgrade agents, see Upgrading Agents [page 314].

Checking for Updates Manually

If your system is eligible for upgrade and your server is behind a firewall, your server can't automatically check for updates. However, you can still request update information through the web application.

If the checking period elapses and you don't have any update messages, you'll receive a warning in your alerts. This prompts you to request the update information manually.

To check for updates manually:

1. In your user profile, select **About**.

2. Click **Check For Updates**.

Chapter 16: Troubleshooting

Following are some known issues to review when troubleshooting Deployment Automation.

For more help with troubleshooting, see the Knowledgebase and the Community website.

Conflict with WebDAV in IIS

If you are running Deployment Automation under JBOSS and IIS, you must ensure that WebDAV is removed from IIS on the server that hosts Deployment Automation. WebDAV prevents Deployment Automation from using critical web service DELETE and PUT operations. For example, you cannot even apply a license in Deployment Automation.

To remove WebDAV:

1. Select the website in IIS.
2. Edit the Handler Mappings and remove WebDAV.
3. Navigate back to the website Home page.
4. Edit the Modules and remove the WebDAVModule.

Installation Failure Due to Interruption

Your installation may fail due to system interruption the first time you attempt to install. In this case, the installation runs very quickly and on the **Install Wizard Completed** dialog box, a message similar to the following appears:

```
The installation wizard was interrupted before Deployment Automation could
be completely installed.

Your system has not been modified. To complete installation at another time,
please run setup again.
```

To complete the failed installation:

1. Run the installer again to remove the installation.
2. Run the installer again to install.

This time, the installation should take a longer time to copy the files and you'll receive a message with installation details on the **Install Wizard Completed** dialog box.

Import Failure or Issues After Import

Generally export files are backward compatible, but you may encounter an issue when importing export files from prior versions of Deployment Automation.

Following are some examples of the types of errors you may encounter and ways to resolve them.

Error Type	Resolution
You receive an error when you try to import environments that were exported from a prior version of Deployment Automation.	You should upgrade the primary Deployment Automation server and then export the objects again. You should then be able to import them into a target server of the same version.
You cannot import an application from a `json` file because it was exported while using a different version of the plugin than the one you have. For example, you receive an error similar to "Plugin DA Versioned File Storage Version 20 does not exist."	You can use the resolution in the prior entry, or you can edit the exported application `json` file using a text editor to change some of the parameters and try the import again. To resolve the error given in the example, you must change all occurrences of `pluginName` and `pluginVersion` in the `json` file to use to the updated name and version for the plugin you have loaded. In our example, `pluginName` should be changed to *DA Versioned File Storage* and `pluginVersion` should be changed to *20*.
Your import appears to work, but when you try to deploy, you receive a generic run error. This means the structure of the application may have been imported incorrectly. For example, if you can deploy at the component level but not the application level, there is a structure mismatch in the imported files.	You can typically fix this by removing and re-adding the component to the application.

SSO Login Failure After a Tomcat Upgrade

If you have configured Single Sign-On (SSO) with one version of Common Tomcat and have upgraded Deployment Automation to a version that uses a different version, you must configure Common Tomcat for SSO again, including setting the parameters in the `gatekeeper-core-config.xml` file. Otherwise, the SSO login will fail.

You must set these parameters by copying over the corresponding strings from earlier version of the `gatekeeper-core-config.xml` file. Copying and replacing the entire file from the earlier Common Tomcat installation does not work.

For instructions on setting up SSO, refer to the *Deployment Automation Integration Guide*.

Process Failures or False Successes

Some common reasons for process failures and false successes are covered here.

Issue	Resolution
The process runs quickly and the status is Success, but the process didn't run and the output log says that the component already exists in that environment.	If you want to run the process in that environment regardless of whether the component already exists, deselect the **Only Changed Version** check box when you run the process.
The process runs as a global process but when you try to run it as a component process, it fails.	You may not have mapped resources for the component within the application environment. This is required for component processes. See Mapping Application Components to Resources [page 273].
When using a plugin, the status of the process doesn't show the same status as that in the integrating product.	Click the Refresh button to see the latest status. In some cases, additional action may be necessary. See the documentation for your specific plugin.

Navigation Tips

New or occasional users may find the following tips helpful as you navigate the user interface.

Navigation Point	Resolution
When you're in the process editor, you click other options but the process editor still shows on the screen.	You must click the **x** in the upper right of the page beside the other controls to close the process editor before you can continue navigation to other options. See Designing Component Processes [page 241].
When you click **View Details** for an application environment, the page won't open.	If the click is released too slowly, the page sometimes doesn't open. If a quick single-click won't work, try double-clicking. See Application Environments [page 269].
You clicked a link after you ran a process request, and now can't get back to the page that shows the process results.	For information on viewing process request information, see Viewing an Application Process Request [page 373] and Viewing a Component Process Request [page 374].

Navigation Point	Resolution
You can't find the option to map a component to a resource.	You map components to resources for each application environment to which the component is to be deployed. See Mapping Application Components to Resources [page 273].
You can't find the option to map an agent to an environment.	You map agents to environments at the global level in the environment **Resources** tab. See Mapping Agents and other Resources to Environments [page 299].
You can't find the page that shows the environments graphically so that you can configure them.	The environments are shown graphically in the application they are associated with. See Adding Environments to Applications [page 270].
You can't find a way to promote or rerun a process in an environment that is part of a pipeline.	If your pipeline is configured for auto promote and a process fails in one of the environments, you must run the process again for the pipeline, starting with the first environment. See Chapter 29: Ways of Deploying [page 367].
You can't find the replication import and export options.	Select any Administration page, such as System. Expand the **Action** drop-down in the upper right of the page. There you will see the **Replication** options. See Chapter 27: Replication [page 351]. **Note:** The replication options are not available from the drop-down selection on the left side of the page.

Part 4: Management

This section contains information on using Deployment Automation.

Included are the following topics:

Chapter 17: Management Overview

The Management page is where you create and configure the primary elements of Deployment Automation, design your processes, and initiate deployments. The following overview of the **Management** page will help get you started managing your Deployment Automation processes. Following are some of the selections on this page.

- **Applications**

  See Chapter 19: Managing Applications [page 263].

- **Components**

  See Chapter 18: Managing Components [page 199].

- **Environments**

  See Chapter 20: Managing Environments [page 297].

- **Deployment Packages**

  See Chapter 23: Managing Deployment Packages [page 325].

- **Pipelines**

  See Chapter 21: Managing Pipelines [page 303].

- **Resources**

  See Chapter 22: Managing Resources [page 307].

- **Global Processes**

  See Chapter 24: Managing Global Processes [page 335].

- **Configuration**

  See Chapter 26: Configuring Within the Application Hierarchy [page 349].

- **Action**

  Select from management actions, including the following:

  - **New:** Application, Component, Environment, Deployment Package, Global Process, Pipeline, Resource, Template (Component)

  - **Import:** Application, Component, Environment, Global Process, Pipeline, Template (Component)

Chapter 18: Managing Components

Components typically map to a functional part of a real world application, and represent deployable items, or artifacts, such as files, images, databases, and configuration materials.

Components have user-defined *component processes* that operate on those items, usually by deploying them. Components have *versions*, which are used to ensure that proper component instances get deployed.

When you create a component, you identify the source and define how the artifacts will be brought into Deployment Automation. The artifacts can come from a number of sources, such as file systems, build servers such as Jenkins, configuration management systems such as Dimensions CM, source control systems such as Subversion and PVCS, and Maven repositories. Each component represents artifacts from a single source.

Upon automatic or manual version import, Deployment Automation imports the component's artifacts into the Deployment Automation artifact repository, CodeStation. This stores *component versions*.

You create and configure components in the user interface as detailed in the following topics:

Components Page Overview

You typically manage components in the **Components** page of the user interface. Following are some of the things you can do from this page.

- Filter the component list by components or component templates.
- Filter by the components created by you or by all users.

- Toggle the **Inactive** button to include or exclude inactive components.
- Click the **Component Actions** button to create or import components.
- Click the **Component Template Actions** button to create or import component templates.
- Click the name to open the page for a listed component or component template.
- Click an icon beside a specific component or component template in the list to do the following:
  - Edit
  - Copy
  - Inactivate/Activate
  - Export
  - Delete

Creating Components

You can create a new component or import an existing one. See Importing/Exporting Components [page 229] for information about importing components.

To create a component:

1. Ensure that any artifacts you want to import into Deployment Automation are ready and are in the location from which you want to import them, such as your file system or a source configuration tool.
2. Navigate to **Management > Components**.
3. Click **Create**.
4. Enter the information needed to create the component as follows:

Option	Description
Name	Identifies the component; appears in many UI features. Required.
Description	The optional description can be used to convey additional information about the component. If the component is used by more than one application, for example, entering "Used in applications A and B" can help identify how the component is used.

Option	Description
Template	A component template enables you to reuse component definitions; components based on templates inherit the template's source configuration, properties, and process. Any previously created templates are listed. A component can have a single template associated with it. The default value is *None*. If you select a template, the **Template Version** field is displayed, which is used to select a template version. By controlling the version, you can roll out template changes as required. The default value is *Latest Version* which means the component will automatically use the newest version, by creation date. See Creating Component Templates [page 257]. **Note:** If you select a template that already has a source config type, the dialog box will change to reflect values defined for the template. Several fields, including the **Source Config Type** field, will become populated and locked.
Source Config Type	Defines the location from which to load the component's artifacts into Deployment Automation. All artifacts for a component must have the same source configuration type. Selecting a value displays additional fields associated with the selection. For more information see Source Configuration Type Fields [page 203]. **Note:** If you selected a template, this field is locked and its value is inherited from the template.
Import Versions Automatically	If selected, the source location is periodically polled for new versions; any found are automatically imported. • Polling Period The time period in seconds between each poll for new versions. The polling interval is controlled by the Automatic Version Import Check Period (seconds) in **Administration > System > System Settings**. The default value is 15 seconds. • Quiet Period The time period in minutes in which no polling is done. The quiet period starts after the last polling period that detected changes. There is no default quiet period; you must set it for each source type. If left deselected, you can manually create versions by using the Versions pane. By default, the check box is deselected. See Importing Versions Automatically [page 232] for more information.

Option	Description
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. This option cannot be deselected if the selected source configuration type cannot be used to populate versions directly. If the option to deselect is available, see your source configuration type in Source Configuration Type Fields [page 203] for more details.
Default Version Type	Required. Defines how versions are imported into CodeStation. **Full** means the version is comprehensive and contains all artifacts; **Incremental** means the version contains a subset of the component's artifacts. The default value is Full.
Inherit Cleanup Settings	Determines how many component versions are kept in CodeStation, and how long they are kept. If selected, the component will use the values specified in **Administration > System > System Settings**. If deselected, additional fields appear.
Days to Keep Versions	If Inherit Cleanup Settings is not selected, this field appears, which enables you to define custom values. This is initially set to *-1*, keep indefinitely.
Number of Versions to Keep	If Inherit Cleanup Settings is not selected, this field appears, which enables you to define custom values. This is initially set to *-1*, keep indefinitely.
Run process after creating a new version	If selected, you may specify an application process and environment. The given application process will be run after a new version of the component is created.

5. To create additional components click **Save and New**. To start configuring the component you just created, click **Build Out**.

Saved components are listed in the **Components** page. If you have selected **Import Versions Automatically**, the initial versions will appear under the component's **Versions** after the polling period plus any quiet period have elapsed. If not, you must import the initial versions manually. See Component Versions [page 231].

Note: You can also create components from a selected application, with the option to create and then add the component in one dialog.

Source Configuration Type Fields

The **Source Config Type** field enables you to select a built-in or custom source configuration type. This enables you to load or point to artifacts stored in external tools or locations into Deployment Automation as component versions.

Loading artifacts into Deployment Automation enables you to track your artifacts as component versions as they are deployed into application environments.

Note: The source configuration types are not the same as plugins. Plugins are special integrations used in processes. For details on using plugins, see Component Processes [page 238] and the *Deployment Automation Plugins Guide*.

Depending on the **Source Config Type** you select, various tool-specific fields appear. Point to the question mark (**?**) after a field to view a description of the field.

Select `None` if you don't want to put artifacts into Deployment Automation. You might decide to do this if you are using another product for the deployment activities, such as Dimensions CM, and using Deployment Automation only for further configuration and processing post-deployment. If you use `None`, you may not have visibility into artifacts' associated component versions within Deployment Automation application environments, and would need to rely on the other product for that information.

The source configuration tools that you can select are shown in the following table.

Source Config Type	Description
AnthillPro	Select this to load artifacts that are stored in AnthillPro into Deployment Automation as component versions.
Artifactory - Folder based	Select this to load artifacts that are stored in file folders managed by Artifactory into Deployment Automation as component versions.
Artifactory - Maven / NuGet	Select this to load artifacts that are stored in Maven or NuGet repositories that are managed by Artifactory into Deployment Automation as component versions.
ClearCaseUCM	Select this to load artifacts that are stored in ClearCase UCM into Deployment Automation as component versions.
Dimensions	Select this to load artifacts that are stored in Dimensions CM into Deployment Automation as component versions.
File System (Basic)	Select this to load artifacts into Deployment Automation from directories in your file system. This imports all files in the subdirectories and creates a component version either on a designated name or based on a version name pattern. Automatic import is not supported with this option.

Source Config Type	Description
File System (Versioned)	Select this to load artifacts into Deployment Automation from directories in your file system, creating a component version for each subdirectory in the base path.
Git	Select this to load artifacts that are stored in Git into Deployment Automation as component versions.
Jenkins	Select this to load artifacts that are stored in Jenkins into Deployment Automation as component versions. This does not display additional fields, but rather indicates that the Jenkins plugin for Deployment Automation is configured and activated.
Luntbuild	Select this to load artifacts that are stored in Luntbuild into Deployment Automation as component versions.
Maven	Select this to load artifacts that are stored in Maven into Deployment Automation as component versions.
Nexus	Select this to load artifacts that are stored in Nexus into Deployment Automation as component versions.
PVCS	Select this to load artifacts that are stored in PVCS into Deployment Automation as component versions.
Perforce	Select this to load artifacts that are stored in a Perforce versioning engine into Deployment Automation as component versions.
StarTeam	Select this to load artifacts that are stored in Borland StarTeam into Deployment Automation as component versions.
Subversion	Select this to load artifacts that are stored in Subversion into Deployment Automation as component versions.
TFS	Select this to load artifacts that are stored in Microsoft Team Foundation Server (TFS) into Deployment Automation as component versions. Use this if using XAML build automation.
TFS vNext	Select this to load artifacts that are stored in Microsoft Team Foundation Server (TFS) 2015 and above into Deployment Automation as component versions. If you are using XAML build automation, use the TFS source configuration type.
TFS_SCM	Select this to load artifacts that are stored in TFS_SCM into Deployment Automation as component versions.
TeamCity	Select this to load artifacts that are stored in JetBrains TeamCity into Deployment Automation as component versions.

Source Config Type	**Description**
TeamForge	Select this to load artifacts that are stored in CollabNet TeamForge into Deployment Automation as component versions.

For information on some of the source configuration types' fields, see the following topics.

- File System (Basic) [page 205]
- File System (Versioned) [page 206]
- AnthillPro [page 207]
- Artifactory [page 208]
- Dimensions [page 210]
- Git [page 212]
- Jenkins [page 213]
- Maven [page 214]
- Nexus Sonatype [page 215]
- PVCS [page 217]
- Subversion [page 218]
- TeamCity [page 220]
- TFS [page 220]
- TFS vNext [page 222]
- TFS_SCM [page 224]
- TeamForge [page 225]

File System (Basic)

The File System (Basic) source configuration type is implemented as an external type.

When you select File System (Basic) for Source Config Type, Deployment Automation imports all contents of a designated directory from the file system as follows:

- At import, imports everything in the given base path, including all files at the base path level, its subdirectories, and all files in all of its subdirectories.
- If you specify the Version Name Pattern and Next Version Number, Deployment Automation auto-increments version numbers and assigns them as the component version names. With this option, you can use Import Versions Automatically or import manually.

- If you do not specify a version name pattern, you must manually import the component version, and you will be prompted to enter a version name at the time of manual import. The Import Versions Automatically option, if selected, is ignored.

File System (Basic) Source Fields Table

Field	Description
Base Path	Base path for artifact storage.
Always Use Name Pattern	When selected, the Version Name Pattern will be used for new version names at the time of import. If this is not selected, the user is prompted for the version name at the time of import.
Version Name Pattern	An optional field to create a template for automatic version names. Use `${version}` to reference the **Next Version Number** field.
Next Version Number	An integer used for the next version created. This number is automatically incremented for every version created.
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.
Use Latest Component Properties for Version Download	If this is selected, the current component properties are used to download this component's version artifacts in **Versions > Artifacts**. You should deselect this if the component's storage location or connection details have changed since the version artifacts you want to download were imported, so that the component properties that were set when the version was created are used instead. See Maintaining Versions Externally [page 233].
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].

File System (Versioned)

The File System (Versioned) source configuration type is implemented as an external type.

When you select File System (Versioned) for Source Config Type, Deployment Automation imports the subdirectories of a designated directory from the file system as follows:

- At import, imports only the subdirectories and contents of subdirectories; it does not include the contents directly in the base path.
- The first level subdirectory in the base path becomes a component version and the version name is the name of the subdirectory.
- It does not prompt for version name during manual import; version naming is always automatic and is always the first subdirectory name.

File System (Versioned) Source Fields Table

Field	Description
Base Path	Path to directory containing artifacts. The content of each subdirectory within the base directory is considered a distinct component version. The subdirectory with the most recent time-stamp is considered the "latest version"; however, is a good practice to name the subdirectories according to your version scheme, such as v1.0, v1.1, and so on.
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.
Use Latest Component Properties for Version Download	If this is selected, the current component properties are used to download this component's version artifacts in **Versions > Artifacts**. You should deselect this if the component's storage location or connection details have changed since the version artifacts you want to download were imported, so that the component properties that were set when the version was created are used instead. See Maintaining Versions Externally [page 233].
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].

AnthillPro

IBM AnthillPro combines continuous integration with automated deployments and test orchestration to create a continuous delivery pipeline.

To use AnthillPro as an artifact source, select AnthillPro from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

AnthillPro Fields Table

Field	Description
AnthillPro URL	AnthillPro Repository URL
User	AnthillPro username
Password	Password for the specified username
Project	Name of the project in Anthill
Workflow	Name of the workflow in Anthill from which to import builds
Status	Only builds with this status will be imported
Artifact Sets	A new-line-separated list of artifact sets to resolve from Anthill when Copy to CodeStation is enabled

Artifactory

JFrog Artifactory is a universal artifact repository manager. You can use Artifactory with artifacts stored in folders or with Maven or Nuget as an artifact source.

To use Artifactory as an artifact source, select Artifactory in **Source Config Type** and then configure the type-specific fields. For information about creating components, see Creating Components [page 200].

Use the **Copy to CodeStation** option as follows:

- If you deselect **Copy to CodeStation** for a component using this source configuration type:
  - Empty versions are created in Deployment Automation upon import. To download Artifactory-managed versions in processes, you must use the Artifactory plugin. See the JFrog Artifactory section in the *Deployment Automation Plugins Guide*.
  - You must use the Artifactory plugin to download artifacts as part of your component processes. You cannot use the DA VFS plugin to download them.
- If you select **Copy to CodeStation** for a component, you must use the DA VFS plugin to download artifacts.

  See the *Deployment Automation Plugins Guide* for plugin details.

Note: If you are using Apache Maven without Artifactory as your source configuration type, see Maven [page 214].

Artifactory - Folder Based Fields Table

Prerequisite:

To use this option, you must do the following:

1. On the Artifactory server, navigate to **Admin > General Configuration > Folder Download Settings**.
2. Select **Enable Folder Download**.

Following are the fields for Artifactory with artifacts stored in folders as an artifact source.

Field	Description
Server URL	The base URL for the Artifactory server, such as `http://serverName:8081/artifactory`
Username	Username for authentication to the Artifactory server
Password	Password for authentication to the Artifactory server
Repository	Name of the repository
Module	Module path to the artifacts in the specified repository
Latest Artifact Count	The maximum number of latest artifact versions to be imported
Version Name Pattern	If specified, only versions with names matching this regular expression will be imported.
Preserve Execute Permissions	If this is selected, file execute permissions are saved with the files.

Artifactory - Maven / NuGet Fields Table

Following are the fields for Artifactory with Maven or Nuget as an artifact source.

Field	Description
Server URL	The base URL for the Artifactory server, such as `http://serverName:8081/artifactory`
Username	Username for authentication to the Artifactory server
Password	Password for authentication to the Artifactory server

Field	Description
Repository	Name of Maven or NuGet repository
Package Type	Select the type of package to be used as the repository, Maven or NuGet.
Group ID	For Maven only, this is the group ID of the artifact, such as `com.mycompany` or `com/mycompany`. For NuGet, leave this field blank. **Tip:** Group ID is the `<groupid>` parameter in the Maven dependency within the Artifactory artifact.
Artifact ID or nuget.id	Enter the name of the artifact to be imported. For Maven it should be the artifact ID, such as `mypackage-test`. For NuGet it should be the value of the artifact nuget.id property, such as EntityFramework. **Tip:** Artifact ID is the `<artifactid>` parameter in the Maven dependency within the Artifactory artifact.
File Name Pattern	If specified, only files with names matching this regular expression will be imported.
Latest Artifact Count	The maximum number of latest artifact versions to be imported
Version Name Pattern	If specified, only versions with names matching this regular expression will be imported. For a snapshot repository, specify the snapshot version, such as `1.0-SNAPSHOT`.
Preserve Execute Permissions	If this is selected, file execute permissions are saved with the files.

Dimensions

The Dimensions CM source configuration type is implemented as an external type.

Dimensions CM is a process management and change control system that helps you organize, manage, and protect your software development projects.

To use Dimensions CM as an artifact source, select `Dimensions` from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

Tip: Deployment Automation always takes the latest baseline during automatic import. Therefore, if there are three baselines A, B, and C, with C being the latest, only C would be imported. If you add baseline D, we will import D, but A and B will not be imported unless you manually import them.

Note: The Dimensions CM Source Config Type integration is not the same as the Dimensions CM plugin. Plugins are special integrations used in processes. For details on using the Dimensions CM plugin, see the *Deployment Automation Integration Guide*.

Dimensions Fields Table

Field	Description
Username	Dimensions CM user name. For information about user impersonation, see User Impersonation [page 251].
Password	Password associated with the Dimensions CM user name
DB Name	Name of the Dimensions CM database. For example: `cm_typical`
DB Conn	Name of the Dimensions CM connection to be used. For example: `dim14` A connection/session is required to send or receive commands to/from the database.
Server	Server managing the Dimensions CM database. For example: `my-dim-server`
Product Spec	Location of the Dimensions CM managed artifacts. For example: `qlarius`

Field	Description

Includes	Specify patterns to match files to include in the upload. You can match exact file paths and file names, or you can specify wildcards. The default value is `**/*`, for all directories and files.
	You can specify exact file path and file name combinations with one entry per line. For example,
	`Maestro/RLM.jobs`
	`Maestro/RLM.sched`
	You can specify wildcards to match directories and files where the wildcard `**` indicates every directory and the wildcard `*` indicates every file. For example, the pattern `dist/**/*` would retrieve the entire file tree under the `dist` directory.

Excludes	Specify patterns to match files to exclude from the upload. You can match exact file paths and file names, or you can specify wildcards. See the examples in the previous table entry.
Use Latest Component Properties for Version Download	If this is selected, the current component properties are used to download this component's version artifacts in **Versions > Artifacts**. You should deselect this if the component's storage location or connection details have changed since the version artifacts you want to download were imported, so that the component properties that were set when the version was created are used instead. See Maintaining Versions Externally [page 233].
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].

Git

The Git source configuration type is implemented as an external type.

Git is a free and open source distributed version control system.

To use Git as an artifact source, select `Git` from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

Git Fields Table

Field	Description
Repository URL	Enter the base URL for the Git repository.
Branch	Enter the Git branch from which to get artifacts, such as `master`.
Watch for tags	If selected, Deployment Automation watches for new tags instead of new versions.
Includes	The patterns to match files to upload. The wildcard ** indicates every directory and * indicates every file. For example, `dist/**/*` retrieves the entire file tree under the `dist` directory.
Excludes	The patterns to match files to exclude from the upload.
GIT Path	Enter the Git client executable, using the full path if necessary. For example: `git`
Preserve Execute Permissions	If selected, file execute permissions are saved with the files.
Use Latest Component Properties for Version Download	If this is selected, the current component properties are used to download this component's version artifacts in **Versions > Artifacts**. You should deselect this if the component's storage location or connection details have changed since the version artifacts you want to download were imported, so that the component properties that were set when the version was created are used instead. See Maintaining Versions Externally [page 233].
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].

Jenkins

Jenkins is an extendable open source continuous integration server.

To use Jenkins as an artifact source, you must install the "Serena Deploy" plugin on your Jenkins system. All of the configuration of the plugin is done in Jenkins. For details, see the Community website and search for "Jenkins / Hudson HPI".

When the Jenkins / Hudson HPI plugin is configured, a Jenkins post-build step pushes the built artifacts into the Deployment Automation CodeStation repository and either deploys them immediately, if the **Run process after creating a new version option** is selected, or deploys later when a deployment process associated with the component is executed.

When you select Jenkins in the **Source Config Type** field, no additional fields appear. Selecting Jenkins in the Source Config Type is optional, as it simply indicates that you are using the plugin that is configured and running in Jenkins.

For information about creating components, see Creating Components [page 200].

Important: The Jenkins Source Config Type integration is not a Deployment Automation plugin; it is a Jenkins plugin.

Maven

Apache Maven is a software project management and comprehension tool.

To use Maven as an artifact source, select Maven in **Source Config Type** and then configure the type-specific fields. For information about creating components, see Creating Components [page 200].

If you deselect **Copy to CodeStation** for a component using this source configuration type:

- Empty versions are created in Deployment Automation upon import.
- You must use the Apache Maven plugin to download artifacts as part of your component processes. You cannot use the DA VFS plugin to download them.

If you select **Copy to CodeStation** for a component, you must use the DA VFS plugin to download artifacts.

See the *Deployment Automation Plugins Guide* for plugin details.

Note: If you are using Maven through Artifactory as your source configuration type, see Artifactory [page 208].

Maven Fields Table

Field	Description
Repository URL	Base URL for the Maven repository
User	Maven user name
Password	Password associated with the user name
Group ID	Maven group ID

Field	Description
Artifact ID	Maven artifact ID
Maven Qualifier	Specify the name to append to the filename after the version. Leave blank if not applicable.
File Extension	Extension of the file to be pulled from Maven for new versions
Latest Build Count	Number of most recent builds in Maven to load into Deployment Automation
Version Name Pattern	If set, only versions with names matching this pattern will be imported. For each group matching the first capturing group, the desired number of latest builds will be loaded into Deployment Automation. This property is only necessary when tracking 'older' branches that contain new builds. Example: To return n versions of each 1.0.*, 1.1.*, 1.2.*, ..., use the following regex: (1.[0-9]+).*
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.

Nexus Sonatype

The Nexus Sonatype source type component supports granular access and filtering for artifacts stored in Nexus repositories.

To use Nexus as an artifact source, select Nexus in **Source Config Type** and then configure the type-specific fields. For information about creating components, see Creating Components [page 200].

Nexus Fields Table

Field	Description
Server URL	Base URL and port for the Nexus server, for example: `http://localhost:8081`
User Name	Nexus user name
Password	Password for the Nexus user
Repository	The name of the Nexus repository
Nexus Component	Specify a Nexus component name if you only want to import versions of that component. Use this field when you have multiple components in the same repository, for example, for NPM repositories.

Field	Description
Component Group	The group ID of the component to which the assets belong. In Maven repositories the ID follows Java package naming rules, for example, `org.apache.maven`.
Import folders as versions	Applies to repository types that do not have specific versions, for example, 'raw'. Folders with subfiles are imported as separate versions.
Import all new versions	Selected: all missing versions are imported. Not selected: only the latest version is imported.
Skip empty versions	Versions without content are not imported.

Include files	Specify patterns to match files to include in the upload. You can match exact file paths and file names, or you can specify wildcards. The default value is `**/*`, for all directories and files.
	You can specify exact file path and file name combinations with one entry per line. For example,
	`Maestro/RLM.jobs`
	`Maestro/RLM.sched`
	You can specify wildcards to match directories and files where the wildcard `**` indicates every directory and the wildcard `*` indicates every file. For example, the pattern `dist/**/*` would retrieve the entire file tree under the `dist` directory.

Exclude files	Specify patterns to match files to exclude from the upload. You can match exact file paths and file names, or you can specify wildcards. See the examples in the row above.
Preserve execute permissions	File execute permissions are preserved.
Import versions automatically	Configured version source are polled periodically for new versions.

Field	Description
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].
Use latest component properties for versions download	When component version artifacts are downloaded, the properties set for the associated component are used instead of the properties set when the version is created. Component version properties are always preserved when a version is created.
Default Version Type	The default version type to be created for imported versions.
Inherit cleanup settings	This environment will inherit version cleanup configurations from the settings of each component or the system.

PVCS

PVCS Version Manager is a version control management tool.

To use PVCS as an artifact source, select `PVCS` from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

PVCS Fields Table

Field	Description
PCLI Path	Path to the PVCS CLI tool.
Database Path	Path to the PVCS database.
Base Path	Base path of the repository.
Project Path	Path to the project.

Field	Description
Archive Path	The location of the archive relative to the database path.
Includes	The patterns to match files to upload. The wild card ** indicates every directory and the wildcard * indicates every file. For example, the pattern `dist/**/*` would retrieve the entire file tree under the `dist` directory.
Excludes	The patterns to exclude files from the upload.
User	PVCS username. For information about user impersonation, see User Impersonation [page 251]
Password	Password associated with the PVCS username.
Always Use Name Pattern	If this option is selected, the Version Name Pattern specified will always be used. Not selected by default.
Version Name Pattern	An optional template for automatic version names. Use `${version}` to reference the Next Version Number field.
Next Version Number	An integer used for the next version created. The number is automatically incremented for each version created.
Preserve Execute Permissions	To save file execute permissions with the files, select this check box. Not selected by default.

Subversion

The Subversion source configuration type is implemented as an external type.

Subversion (SVN) is an open source version control system.

To use Subversion as an artifact source, select Subversion from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

Note: The Subversion Source Config Type integration is not the same as the Subversion plugin. Plugins are special integrations used in processes. For details on using the Subversion plugin, see the *Deployment Automation Plugins Guide*.

Subversion Fields Table

Field	Description
Repository URL	Base URL for the Subversion repository
Watch for tags	Watch for new tags instead of new versions.
Includes	The patterns to match files to include in the upload. The wildcard `**` indicates every directory and the wildcard `*` indicates every file. For example, the pattern `dist/**/*` would retrieve the entire file tree under the `dist` directory.
Excludes	The patterns to match files to exclude from the upload.
User	Subversion user name
Password	Password associated with the user name
SVN Path	Specify the path to the Subversion client executable. Enter the full path if necessary.
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.
Use Latest Component Properties for Version Download	If this is selected, the current component properties are used to download this component's version artifacts in **Versions > Artifacts**. You should deselect this if the component's storage location or connection details have changed since the version artifacts you want to download were imported, so that the component properties that were set when the version was created are used instead. See Maintaining Versions Externally [page 233].
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].

TeamCity

TeamCity is a Java-based build management and continuous integration server from JetBrains.

To use TeamCity as an artifact source, select TeamCity from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

TeamCity Fields Table

Field	Description
Repository URL	Base URL for the TeamCity repository.
User	TeamCity user name.
Password	Password associated with the user name.
Project	Name of the TeamCity project that contains the build configuration.
Build Configuration	Name of the build configuration in the TeamCity project that contains the builds to use in the component.
Build Type	ID of the TeamCity build type.
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.

TFS

Microsoft Visual Studio Team Foundation Server (TFS) is the collaboration platform at the core of Microsoft's application lifecycle management (ALM) solution.

For more information on using TFS, see the Microsoft documentation.

Select TFS from the **Source Config Type** drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

Note: This brings your built artifacts from TFS into Deployment Automation. If you want to bring TFS source controlled artifacts into Deployment Automation, see TFS_SCM [page 224].

Prerequisites

Before you can use TFS source configuration type, ensure the following prerequisites are met.

- An executable named `tftoolxxxx.exe` (where `xxxx` is the version of TFS, and can be 2008, 2010, 2012, 2013, 2015, or 2017) must be installed on the Deployment Automation server.

The Deployment Automation server installer in Deployment Automation version 5.1.5 and later automatically installs the supported `tftool` executables. These are placed in the Deployment Automation server `tools\tfs` directory in folders that match each `exe` name. For example, the `tftool2013.exe` is installed here by default:

```
C:\Program Files\Micro Focus\Deployment Automation Server
\tools\tfs\tftool2013\tftool2013.exe
```

- The Deployment Automation server and the TFS software must be installed on the same machine, because there are certain TFS DLL files required by the tftool executable.
- The installation folder is specified in the `tfsToolLocation` property in the server installation's `conf\server\installed.properties` file, so there is no need to include this in the Path variable.

  Note: If the `tftool` executable is not found in the in designated folder, an error message similar to the following will appear:

  "You must provide a valid TFS tool; `C:\Program Files\Micro Focus\Deployment Automation Server \tools\tfs\tftool2013\tftool2013.exe` was not found."
- If you upgraded or reinstalled and selected the Use Existing Settings option on the General Program Settings panel during installation, the `tfsToolLocation` property may not be set in the server installation's `conf\server\installed.properties` file. In that case, you'll need to add the property to the file manually and set it to the directory path where your tftool executables reside. For example:

  tfsToolLocation=`C:\Program Files\Micro Focus\Deployment Automation Server \tools\tfs`

TFS Fields Table

Field	Description
TFS URL	Specify the TFS server URL. For example: `http://tfs2013:8080/tfs/mycollection`
TFS Version	Select the version of the TFS libraries that you want to use. These are listed from those installed on the Deployment Automation server. See the Prerequisites for details. **Note:** Use 2015 or 2017 with the TFS source configuration type only if you are using XAML builds. Otherwise, see TFS vNext [page 222].
TFS Username	Specify a username if you need to use different credentials than the ones used for the Deployment Automation process.

Field	Description
TFS Password	Specify a password if you specified a username.
TFS Team Project	The name of the TFS team project where the build definition resides
Build Definition	The name of the build definition to monitor for new builds
Build Status	Limit builds to a given status.
Build Quality	Limit builds to a given build quality.
Directory Offset	Directory offset within the root of the build artifacts
File Includes	Specify a new-line-separated list of files to include. Leave blank or use `**/*` to include all.
File Excludes	Specify a new-line-separated list of files to exclude.

TFS vNext

Microsoft Visual Studio Team Foundation Server (TFS) is the collaboration platform at the core of Microsoft's application lifecycle management (ALM) solution. For more information on using TFS, see the Microsoft documentation.

The TFS vNext source configuration type is implemented as an external type and is to be used with TFS 2015 and 2017. If you are using earlier versions of TFS, see TFS [page 220].

Select TFS vNext from the **Source Config Type** drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

Tip:

When you import a version for a component that uses this source configuration type, you are prompted for build number. There you should enter the build that has the artifacts you want to import. If you leave it blank, the latest build's artifacts are imported.

Note: This brings your built artifacts from TFS into Deployment Automation. If you want to bring TFS source controlled artifacts into Deployment Automation, see TFS_SCM [page 224].

Important: If TFS build artifacts are published to a network share, you must ensure that the Deployment Automation application server, Common Tomcat, is running under a user account that has access to that network share. See Server Installation [page 44].

TFS vNext Fields Table

Field	Description
TFS URL	Specify the TFS server URL. For example: `http://tfs2017:8080/tfs/DefaultCollection`
TFS Username	A username that can access the TFS API
TFS Access Token	If using access tokens, specify a valid TFS access token for a TFS user with build (read) permissions. If using basic authentication, specify the user password. **Note:** The TFS vNext source configuration type is implemented using the REST API. Access tokens can be created to access the REST API in TFS 2017. This access token must have build (read) permission.
TFS Team Project	The name of the TFS team project where the build definition resides
Build Definition	Specify the name of the build definition to monitor for new builds
Build Result	Select one or more results to limit the builds that are eligible for version import.

Field	Description
Use Latest Component Properties for Version Download	If this is selected, the current component properties are used to download this component's version artifacts in **Versions > Artifacts**. You should deselect this if the component's storage location or connection details have changed since the version artifacts you want to download were imported, so that the component properties that were set when the version was created are used instead. See Maintaining Versions Externally [page 233].
Copy to CodeStation	This option, selected by default, creates a tamper-proof copy of the artifacts and stores them in the embedded artifact management system, CodeStation. For optimal use of Deployment Automation, the check box should be selected. If you deselect **Copy to Codestation** for this source configuration type, pointers to versions are saved in Deployment Automation rather than the versions themselves. This enables you to track the versions, but limits some functionality in Deployment Automation. See Maintaining Versions Externally [page 233].

TFS_SCM

TFS_SCM is a Microsoft extension for TFS, Source Control Explorer.

To use TFS_SCM as an artifact source, select TFS_SCM from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

For more information on using TFS_SCM, see https://docs.microsoft.com/en-us/vsts/tfvc/use-source-control-explorer-manage-files-under-version-control.

Note: This brings your source controlled artifacts from TFS into Deployment Automation. If you want to bring TFS build artifacts into Deployment Automation, see TFS [page 220].

TFS_SCM Fields Table

Field	Description
TFS URL	Specify the TFS Repository URL.
User	Specify a username if you need to use different credentials than the ones used for the Deployment Automation process.
Password	Specify a password if you specified a username.
tf.exe Path	`tf.exe` client executable. Enter the full path if necessary.

Field	Description
Workspace	Enter the name of the TFS workspace to use. It will be created if it does not exist.
Server Folder	The TFS server folder to load
Local Folder	The local folder of the workspace
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.

TeamForge

CollabNet TeamForge is an integrated Web-based platform that enables teams to work collaboratively through the various stages of the application life cycle.

To use TeamForge as an artifact source, select TeamForge from the Source Config Type drop-down list box and then configure the type-specific fields described here. For information about creating components, see Creating Components [page 200].

For more information on using TeamForge, see http://www.collab.net/products/teamforge.

Note: The TeamForge Source Config Type integration is not the same as the TeamForge plugin. Plugins are special integrations used in processes. For details on using the TeamForge plugin, see the *Deployment Automation Plugins Guide*.

TeamForge Fields Table

Field	Description
TeamForge URL	Base URL of the TeamForge server.
Username	Username for authenticating with TeamForge.
Password	Password for authenticating with TeamForge.
Project Name	TeamForge project name from which to import component versions.
Package Name	TeamForge package name from which to import component versions.
Preserve Execute Permissions	When enabled, file execute permissions are saved with the files.

Component Properties

Component properties include not only properties to be resolved for the component, but also settings that can be overridden for each environment the component is mapped to and for each version of the component.

For more information on the use of properties in Deployment Automation, see Chapter 25: Properties [page 341].

The types of properties available to add to components are as follows:

- **Properties**

  These user-defined component properties can be used in scripts and plugins. Properties inherited from templates cannot be modified on the component level.

- **Environment Property Definitions**

  These user-defined environment properties are available to environments that use the component. Values can be set at the individual environment level.

- **Version Property Definitions**

  These user-defined version properties are available to this component's versions. Values can be set at the individual version level.

For information on working with component properties, see the following topics:

- Adding Component Properties [page 226]
- Adding Component Environment Properties [page 227]
- Adding Component Version Properties [page 228]

Adding Component Properties

Add properties to a component for information that may have values specific to that component. A component property value would override the value for the same-named property at the application level.

To add a property to the selected component:

1. Navigate to **Management > Components > [select component]**.
2. Select **Properties** and then click the **Properties** tab.
3. To set a single property, click **Add Property**.
   a. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description

Field	Description
Secure	Select this if the property should be secure. Secure properties are stored encrypted and displayed obscured in the Deployment Automation user interface.
Value	Value for the property that will be used by steps in a process

4. To set multiple properties, click **Batch Edit**.

   a. In the **Properties** box, enter multiple properties and their values on separate lines as property=value.

   For example, if you had properties `host` and `port`, you would enter them as follows:

   ```
   host=myServer

   port=8080
   ```

   Note:

   The following restrictions apply to batch edit:

   - The **Secure** option cannot be set.
   - The value cannot contain the "=" symbol.

5. Click **Save**.

To use the property in a process, reference it when you configure a step that uses it. Use the following syntax: `${p:component/<property name>}`

Adding Component Environment Properties

Component environment properties are user-defined properties that you can define on the component and then give unique values for each of the environments mapped to the component. This lets you run the same process step in multiple environments that have different configurations; Deployment Automation resolves the property values as the component is deployed to each environment. For example, you may have different web server instances on each environment, and can add a component environment property and enter a different value for each environment, DEV, INT, UAT, and PROD.

To add component environment properties:

1. Navigate to **Management > Components > [select component]**.
2. Select **Properties** and then click the **Environment PropDefs** tab.
3. Click the **Add Property** button.
4. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

5. Click **Save**.

You can now set values for the property in each of this component's application environments. See Setting Component Environment Property Values [page 273].

Adding Component Version Properties

Component version properties are user-defined properties that you can define on the component and then give unique property values for each version of the component.

To add component version properties:

1. Navigate to **Management > Components > [select component]**.
2. Select **Properties** and then click the **Version PropDefs** tab.
3. Click the **Add Property** button.
4. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

5. Click **Save**.

You can now set values for the property in each of this component's versions. See Setting Component Version Property Values [page 235].

Importing/Exporting Components

Components can be exported and then imported into another server. This can be useful if you have multiple Deployment Automation servers, such as test and production servers, where you create and validate the components on the test server and then export them and import them into the production server.

- Exporting Components [page 230]
- Importing Components [page 230]

Related Topics

Chapter 27: Replication [page 351]

Exporting Components

Exporting a component creates a JSON file that contains the component's source configuration information, properties, and processes. For information about JSON, see www.json.org.

To export a component:

1. Navigate to **Management > Components**.
2. In the column beside a component, click **Export**.

You can load the file into a text editor, or save it. If you save it, a file is created with the same name as the selected component, such as `helloWorld.json`.

Importing Components

When you import a component, you can create a new component or upgrade an existing one. Additionally, if the imported component was created from a template, you can use it or create a new one.

Note: If the imported component has the **Import Versions Automatically** property set to true, the new component will automatically import component versions as long as the artifacts are accessible to the importing server.

To import a component

1. Navigate to **Management > Components**.
2. Click **Import**.
3. If you want to upgrade an existing component, select **Upgrade Component**.

   If the component's name in the JSON file (not the name of the file itself) matches an existing component, the component's properties are updated with the new values, and new items, such as processes, are added. If the name of the component is not found, the command has no effect.

   Note: The component's name is the first property in the JSON file; for example,

   ```
   "name": "helloWorld",
   ```

4. If the imported component was originally created from a template, use the **Component Template Upgrade Type** options to specify how you want to use the template. For these options, the template must be on the importing server. If the imported component was not created from a template, these options are ignored.

   - To use the imported component's template, select **Use Existing Template**. The new component will be an exact copy of the imported one and contain a pointer to the imported component's template. This option is especially useful if you are importing a lot of components based on the same template.

     If you are upgrading, the component will also point to the imported template.

   - To create a new template, select **Create Template**. The new component will be an exact copy of the imported one and contain a pointer to the newly created template (which is based on the imported component's template).

If you are upgrading a component, a new template is also created used.

- When you want to create a fresh installation and ensure a template is *not* on the importing server, select **Fail if Template Exists**. If you are creating a component, it will create both a new component and template unless the template already exists, in which case the component is not imported.

  If you are upgrading a component, the upgrade will fail if the imported component's template already exists.

- To ensure the template is on the importing server, select **Fail if Template Does Not Exist**. If you are creating a component, it will create both a new component and template unless the template does not exist, in which case the component is not imported.

  If you are upgrading a component, the upgrade will fail if the imported component's template does not exist on the importing server.

- To upgrade the template, select **Upgrade if Exists**. This option creates a new component and upgrades the template on the importing server. If the template does not exist, a new one is created.

5. Click **Choose File** and select the JSON file containing the component.

6. Click **Import**.

Component Versions

Deployment Automation tracks all artifact changes, which enables you to roll back components or deploy multiple versions of the same component.

For information on working with component versions, see the following topics:

- Importing Component Versions [page 231]
- Adding Statuses to Component Versions [page 234]
- Setting Component Version Property Values [page 235]
- Deleting Component Versions [page 235]
- Inactivating Component Versions [page 235]
- Component Version Cleanup [page 236]
- Downloading Version Artifacts [page 237]
- Displaying the Number of Active Component Versions [page 238]

Importing Component Versions

After defining a component's source and processes, you import its artifacts into the Deployment Automation artifact repository, CodeStation.

By default, a complete copy of an artifact's content is imported into CodeStation. The original artifacts are untouched. This provides tamper-proof storage and the ability to review and validate artifacts with the Deployment Automation user interface.

If you use a source configuration type that supports maintaining versions externally, you can limit CodeStation to using references to the artifacts instead of actually copying them.

Each time a component is imported, including the first time, it is versioned. Versions can be assigned automatically by Deployment Automation, applied manually, or come from a build server. Every time a component's artifacts are modified and re-imported, a new version of the component is created.

A component may have several unique versions in CodeStation. A version can be full or incremental. A full version contains all component artifacts; an incremental version contains only the artifacts modified since the previous version was created.

Note: Typically, you can use the DA VFS plugin to download and deploy the artifacts as part of your component and global processes. See the *Deployment Automation Plugins Guide* for plugin details.

For administrative information, see Configuring Component Version Import [page 173].

For information on how to import, see the following topics:

- Importing Versions Automatically [page 232]
- Importing Versions Manually [page 233]
- Maintaining Versions Externally [page 233]
- Viewing the Version Import Queue [page 234]

Importing Versions Automatically

When the **Import Versions Automatically** check box is selected for a component, the source location is periodically polled for new versions; any found are automatically imported. The default polling period is 15 seconds, which can be changed in System Settings. See Import Versions Automatically [page 0].

If you have multiple versions of files that you are trying to import at once into Deployment Automation, such as Dimensions CM baselines, only the latest version is imported through automatic import. If you want versions other than the latest to be imported, you must manually import the earlier versions.

CAUTION:

If you set this value too low, it is possible for the loading of the previous polling to be interrupted, causing partial import of the component version. You should not set this to less than the default of 15 seconds, but you may set it higher as needed.

You can also set a quiet period. Specified in minutes, the quiet period gives time for any changes detected in a particular polling period to be imported before the next polling period starts. This prevents any changes done during processing to be missed in the next polling period.

Importing Versions Manually

If the **Import Versions Automatically** check box is not selected for a component, you must manually import the files and assign the versions.

To import versions manually:

1. Navigate to **Versions** for the component you want to use, **Management > Components > [select component]**.
2. Select the **Versions** tab and click **Import**. Any new version of the component artifacts that is detected in the source configuration location for the component is imported

Maintaining Versions Externally

If you deselect **Copy to CodeStation** for a component, this tells Deployment Automation to maintain the full versions for that component only in the integrating source configuration tool.

If you do this, the version management for the component is limited as follows:

- In the **Versions > Artifacts** page, you cannot download individual version artifacts; to download them, you must download them all using the **Download All** button.
- Version information is obtained only during the version import. If anyone modifies the artifact version in the source configuration tool, metadata stored about that version is not updated in CodeStation.
- If you change connection parameters to external systems, this may affect the ability to download existing versions. For example, if credentials to a Dimensions CM server have changed or a file share location has been moved, the metadata stored for existing versions will not have the updated information.

  To mitigate this, if you want to download an earlier version, edit the component details and deselect **Use Latest Component Properties for Version Download**. By selecting or deselecting this setting before downloading, you can download versions from the current or original location, based on the component property settings. If you have changed the repository or connection parameters more than once between imports, you may not be able to download some versions.

  CAUTION:

  ⚠ If you modify the source folder, you will not be able to download it again. The download procedure now strictly verifies that the content matches what was originally imported, including file set and size of the files.

To ensure better performance, versions that are not stored in CodeStation are cached in local storage in the server profile. By default, versions are kept in local storage for 24 hours, and then deleted. When a new download request is initiated, the server checks local storage first and provides the content from there if it is found. Otherwise, the server downloads the version and puts it in local storage. If needed, you can change the cache time as follows:

1. Navigate to your Deployment Automation server profile directory. For example:

   ```
   C:\Users\username\.microfocus\da\conf\server
   ```

or

`/opt/MicroFocus/da/`*`username`*`/.microfocus/da`/conf/server

2. Modify the `da_config.xml` to set the `scheduledTimeInHRS` property to the time you want the local storage cache to be retained. For example:

```
<versionCacheCleanUpConfig>
    <scheduledTimeInHRS>12</scheduledTimeInHRS>
</versionCacheCleanUpConfig>
```

If you decide to select **Copy to CodeStation** after previously deselecting, Deployment Automation begins copying the full version into CodeStation. The inventory maintains its integrity.

Viewing the Version Import Queue

As versions are being imported for various components, you can view the version import requests that are running or queued up to run.

To view the queue of component version imports:

1. In the upper right of the Deployment Automation user interface, click the **Version Import View** icon. The number on the icon indicates the number of import requests currently in the queue.
2. In the resulting **Version Import View** overlay, you can:
   - See the list of current import requests.
   - See information about each request such as import type, time scheduled, who did the import, and status.
   - Click the component name link to open the page for that component.

Adding Statuses to Component Versions

Component version statuses are user-managed values that can be added to component versions. Once a status is added to a version, the value can be used in component processes or application gates. See Gates [page 295].

Before you can add a status, statuses must be defined. See Defining Statuses [page 110] for information on defining version statuses.

To add statuses to component versions:

1. Navigate to **Management > Components > [select component]**.
2. Select the **Versions** tab, select a version, and click **Statuses**.
3. Click **Add**.
4. Select the statuses you want to add.
5. Click **Add**.

You can also set statuses in processes using the **Add Status to Version** plugin step.

Setting Component Version Property Values

On this page you set values for the version properties defined for a component. If you have not added version properties for the component, none will appear here.

You cannot add the properties themselves on this page; the version properties must be added at the component level. See Adding Component Version Properties [page 228].

To create component version properties:

1. Navigate to **Management > Components > [select component]**.
2. Select the **Properties** tab.
3. Select the version and click **Edit**.
4. Enter values for the version properties that were previously defined for the component. See Adding Component Version Properties [page 228].
5. Click **Save**.

To use the property in a process, reference it when you configure a step that uses it. Use the following syntax: `{p:component/<property name>}`.

Deleting Component Versions

You can delete any component version.

To delete component versions:

1. Navigate to **Management > Components > [select component]**.
2. Select the **Versions** tab.
3. In the list beside the version you want to delete, click the **Delete** icon.
4. To confirm that you want to delete the version, click **Yes**.

This removes associated version metadata from the repository; original artifacts are unaffected.

Inactivating Component Versions

You can inactivate any component version. Unlike deleted versions, inactive component versions remain in the repository but cannot be deployed.

To inactivate component versions:

1. Navigate to **Management > Components > [select component]**.
2. Select the **Versions** tab.
3. In the list beside the version you want to inactivate, click the **Inactivate** icon.
4. To confirm that you want to inactivate the version, click **Yes**.

To make an inactive version active, use the **Show Inactive** filter to show inactive versions and then in the list beside the version you want to activate, click the **Activate** icon.

Component Version Cleanup

Component versions are automatically cleaned up, or deleted, based on rules that you define in the administration timeline. By default, the cleanup runs every 24 hours. You can create your own cleanup schedules, see Scheduling Version Cleanup [page 157].

The following parameters are used to determine whether versions should be kept.

- **Days to Keep Versions –** number of days to keep versions after they are created.

  Example: A version was created on the 1st of July. If **Days to Keep Versions** is set to 30 days, this version will be kept at least until the 30th of July.

- **Number of Versions to Keep –** number of versions for the current component, ordered in reverse order, to keep. After a new version is created and cleanup is started, the oldest version will be candidate for removing.

There are multiple levels at which the parameters may be set. They are evaluated in ascending order of priority.

1. **System level.** Settings for those parameters can be found in System Settings. Allowed values for both of the parameters are "-1" or "integer greater then 0".

2. **Component level.** Settings for those parameters can be found when you edit a component. To enable them you should clear the **Inherit Cleanup Settings** checkbox. Allowed values for both of the parameters are "-1", "0", or "integer greater then 0".

3. **Environment level.** Settings for those parameters can be found when you edit an environment at the application level. To enable them you should uncheck the **Inherit Cleanup Settings** checkbox. Allowed values for both of the parameters are "-1", "0", or "integer greater then 0".

Environment and component level settings are deselected by default for each component and environment.

The cleanup procedure starts with retrieving all components from the database and looping through their versions in descending order. Every version is evaluated as follows:

1. What values of **Days to Keep Versions** and **Number of Versions to Keep** should be used for this version?

   a. Initial parameter values are set to be equal to system level values.

   b. If the component overrides at least one of these values with a non-zero value, use both of them from the component level instead.

   c. Get a list of environments that ever had this version used in a process request.

   d. Iterate them sequentially. If an environment has cleanup parameters overridden, use the following criteria:

      i. For **Days to Keep Versions** parameter, extend, if needed, the amount of days to the environment level value.

   ii. For **Number of Versions to Keep**, save the environment level value for later use.

2. Sequentially check this version against the set of rules. If a rule indicates the version should be kept, the version is not deleted.

   a. If either **Days to Keep Versions** and **Number of Versions to Keep** is set to **"-1",** keep this version.

   b. If both **Days to Keep Versions** and **Number of Versions to Keep** is set to **"0",** keep this version.

   c. If this version fits the "days to keep version" rule, in that the specified amount of days has not yet passed, keep this version.

   d. If this version fits the "number of versions to keep" rule, in that the version index is less then specified number of versions to keep, keep this version.

   e. If this version has active inventory, keep this version.

   f. If this version fits the "number of versions to keep" on some environment as determined in the previous pass, in that the version index is less than the specified number of versions to keep for this environment, keep this version.

Related Topics

System Settings [page 153]

Downloading Version Artifacts

You can download version artifacts in the Component Versions UI or in component or global processes.

To download version artifacts in the UI:

1. Navigate to **Management > Components > [select component]**.

2. Select the **Versions** tab.

3. Select the version for which you want to download artifacts.

4. On the **Artifacts** tab, do one of the following:

   - Click **Download** beside a version to download a specific artifact.

   - Click **Download All** above the version list to download all associated artifacts in a `zip` file.

For information on downloading artifacts in processes, see the DA VFS plugin information in the *Deployment Automation Plugins Guide*.

Note: If your source configuration type requires a different plugin for artifact download, that information is included in the source configuration type documentation. See Source Configuration Type Fields [page 203].

Displaying the Number of Active Component Versions

You can add a counter that displays the number of active component versions. The counter excludes all archived and inactive versions.

1. Navigate to the directory or folder where the Deployment Automation server configuration is stored. For example:

   ```
   ..\Users\username\.microfocus\da\conf\server
   ```

2. Open `da_config.xml`.

3. Add:

   ```
   <versionCounterConfig>

   <includeArchived>true</includeArchive>

   <includeInactive>true</includeInactive>

   </versionCounterConfig>
   ```

Component History

Deployment Automation retains information about modifications to components so that you can keep the history of change for your Deployment Automation components and the history of processes run against the components.

To view component change and deployment history:

1. Navigate to **Management > Components > [select component] > History**.

2. Select the **Change History** or **Deployment History** tab.

Component Processes

Component processes define how the component should be deployed, installed or interacted with. A process typically contains a number of plugin steps that are inter-dependent, and may include complex logic for actions to perform if a step of the process fails.

For information on creating and designing component processes, see the following topics:

- Creating Component Processes [page 239]
- Component Process Details [page 241]
- Designing Component Processes [page 241]
- Adding Process Steps [page 242]
- Connecting Process Steps [page 242]
- Component Process Step Details [page 244]
- Adding Component Process Properties [page 254]
- Plugin Post Processes [page 255]

- Running Component Processes [page 255]

Creating Component Processes

Each component has at least one process defined for it and can have several. These typically define the configuration information you want to do on the target resources for each component. Frequently-used processes can be saved in component templates and applied to other components.

A component process is created in two steps:

1. Create the component process to configure basic information, such as name
2. Design the process using the process editor

To create a component process:

1. Navigate to **Management > Components > [select component]** .
2. Select the **Processes** tab.
3. Click **Create**.
4. Complete the fields.

Field	Description
Name	Identifies the process; appears in many UI elements. Required.
Description	The optional description can be used to convey additional information about the process.

Field	Description
Process Type	Required. Defines the process type. Available values are: • **Deployment:** deploys a component version to the target resource and updates the inventory after a successful execution. • **Configuration Deployment:** configuration-only deployment with no component version or artifacts; applies the configuration (using properties, or configuration templates) to the target agent and updates the resource configuration inventory afterwards. • **Uninstall:** standard uninstall that removes a component version from a target resource and the resource's inventory. • **Operational (With Version):** operational process that does not add or remove any artifacts or configuration; runs arbitrary steps given a component version. Useful when you want to start or stop some service for a previously-deployed component version. • **Operational (No Version Needed):** same as the previous type, but does not require a component version.
Inventory Status	Required. Status applied to component versions after being successfully executed by this process. `Active` is a built-in status that indicates the component version is deployed to its target resource. An example of a user-defined status is `Staged`, that could indicate that the component version is in a pre-deployment location. The status appears on the Inventory panes for the component itself and environments that ran the process. See Defining Statuses [page 110].
Default Working Directory	Required. Defines the location used by the agent running the process, for temporary file storage. The default value resolves to *agent_directory*\work\component_name_directory. The default properties work for most components; you might need to change it if a component process cannot be run at the agent's location.
Required Component Role	Restricts who can run the process. The available options are derived from the Deployment Automation security system. The default value is `None`, meaning anyone can run the process. For information about security roles, see Chapter 8: Security Administration [page 115].

5. To create additional components processes click **Save and New**.

Component Process Details

After you create a process, beside the process in the process list, click the **Details** button (👁) to open the process details page. From this page you can view and edit the process details, add properties, and view its changes.

1. Click the **Design** button to add and edit process steps.
2. Edit the process details.
3. Select the **Properties** tab to access process properties; select the **Changes** tab to view change history for the process.
4. Click the **Add Properties** button to add process properties.
5. Click the associated icon to edit or delete a listed property.

Designing Component Processes

You can design component processes to tell Deployment Automation what to do when you deploy components.

You design processes using the process editor. In the process editor you drag process steps from a menu of steps onto the design space and configure them immediately. As you place additional steps, you visually define their relationships with one another.

Component Process Design

1. Select from the following in the process designer tools palette:
   - Zoom In
   - Zoom Out
   - Actual Size
   - Print
   - Poster Print
   - Save
   - Cancel
   - Show Outline Box: Use this to move the whole process design around in the design area. For example:
     - Drag its corner inward and move it over parts of the process design to zoom in.
     - Drag its corner outward again to zoom out.
   - Show Connection Options: Select this to show the options for all connections.
2. In **Tools**, select process steps.
3. Use the Search box to find a step by name.
4. Select and drag plugin steps to the design area to add them to your process.

5. **Properties** are automatically shown when you select a step.
6. Connect process steps and edit or delete existing steps or connections.

   Tip: Select a connection to explicitly show its options if the **Show Connection Options** is not selected. Connections are color-coded for the process path: Success (green), Failure (red), and Always (gray).

Adding Process Steps

Component processes may use only one plugin step or may have many steps and relationships. When you initially implement Deployment Automation, you will typically use these process steps to replace existing deployment scripts and manual processes.

To add a step:

1. In **Tools**, drag the step you want to use from available plugin steps onto the design space.
2. Release the step over the design space. You can place the step anywhere on the design space for now and reposition it later.
3. Configure the step's properties in **Properties**.

You can add connections immediately after placing a step, or place multiple steps before defining connections. You can save or cancel at any time.

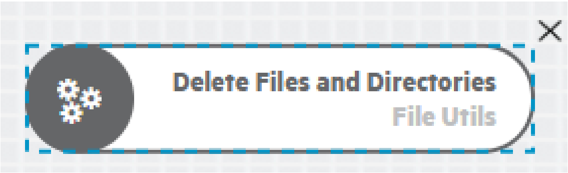
Connecting Process Steps

Connections control process flow. The originating step will process before the target step. Creating a connection between steps is a simple process; drag a connection from the originating step to the target step. Connections are formed one at a time between two steps, the originating step and the target step.

To create a connection:

1. After you have added your steps, the graphical design tools available when steps are selected include a delete option and a connection handle, as shown in the following figure. The **Start** step cannot be deleted or edited.

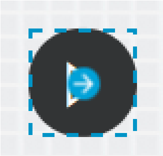
   Graphical Design Tools

2. Point to the step that you want to use as the connection's origin.

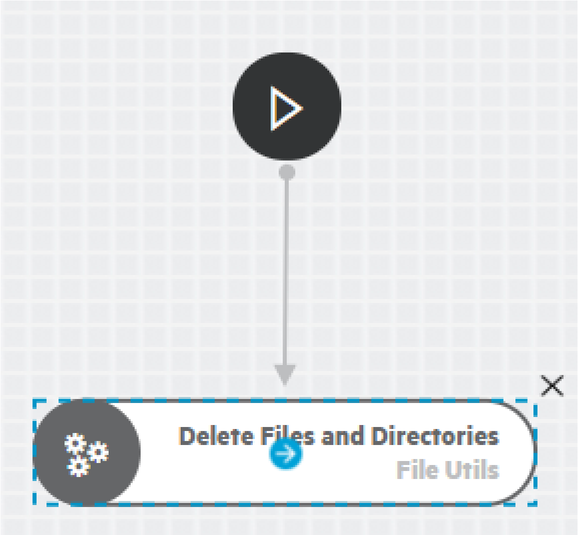
   The cursor changes to the connection handle.

Connection Handle



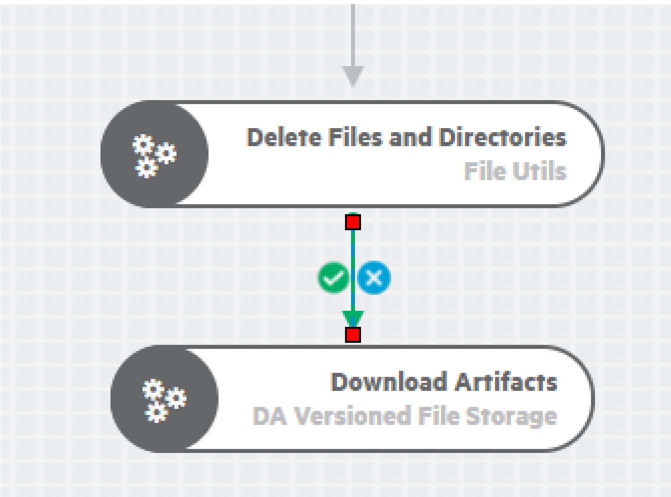
3. Hold and drag the connection tool over the target step.

   Connecting the Steps

4. Release the connection tool over the target step to complete the connection.

   Completed Connection

5. In the **Properties** panel select a connection type, for example, Success. See Using Conditions and Switch Steps in Processes [page 249].

6. To add a conditional transition, in the **Properties** panel select **Conditional** and enter properties. You can use arithmetical (+, -, %, /, *), and logical operators (==,

!=, >, <). The transition is successful only if the condition is resolved to true. You cannot add conditional properties to Switch transitions and the transition from the Start step.

> **Note:** The Spring Expression Language (SpEL) is used to evaluate properties. Strings are delimited by single quotes. To put a single quote itself in a string, use two single quote characters, for example: ''Hello World''. For more information about SpeL see the help: https://docs.spring.io/spring/docs/4.3.10.RELEASE/spring-framework-reference/html/expressions.html

7. Continue with additional steps as needed.

8. After configuring the steps and their properties, to save the process design, click the **Save** icon ( ); if you don't want to save your changes, click the **Discard Changes** icon ( ).

   If saved, the step is on the design space and ready to be connected to other steps.

Component Process Step Details

After you create a component process, select it to design the process steps. The component process steps are described in the following topics:

- Plugins [page 244]
- Component Process Utility Steps [page 245]
- Using Expression Language in Processes [page 249]
- Using Conditions and Switch Steps in Processes [page 249]
- User Impersonation [page 251]

Plugins

Plugins provide functionality in discrete steps to be used in component and global processes for configuration of or deployment into target environments.

Plugins provide the selection of process steps for component and global processes. These include basic Deployment Automation and file operation functions as well as many integrations with third-party tools. Deployment Automation ships with plugins for many common deployment process needs.

Plugins break down a tool's functions into steps such as detecting if a server is stopped, stopping a server, and starting a server. When you use plugins to create a component process, you typically use steps from several plugins. For example, you might create a process using a plugin for a source control tool that deploys a component to a middleware server, and another plugin to configure a step that removes the component from the server.

Deployment Automation provides plugins for:

- Executing standard utility processes, such as inventory management and workflow control and native Deployment Automation operations
- Executing common processes, such as downloading and uploading artifacts, and retrieving environment information

- Interacting with a variety of tools, such as middleware tools, databases, and servers

For a reference of provided plugins, see the *Deployment Automation Plugins Guide*. For plugins written by others, check the Community website.

Related Topics

Loading Plugins [page 107]

Upgrading Plugins [page 108]

Component Process Utility Steps

Several utility steps are provided for component processes.

Process Sleep

The Process Sleep step enables you to enter time in seconds for a process to wait until continuing with the next steps in the process. The following table describes the property for this step.

Property	Description
Sleep Time	Default: 30

Acquire Lock and Release Lock

The **Acquire Lock** and **Release Lock** steps enable you to lock and release a process for a designated resource.

Locks are typically used to prevent specific processes from modifying the same resource at the same time, such as when a database is being updated.

In an Acquire Lock step, the process tries to acquire a lock with the specified name. If there is no other lock with that exact name, the process acquires the lock with that name and continues. If another process is using a lock with that exact name, the process waits until the lock is released by the other process. Any processes that access the resource without a lock or with a lock of a different name do not wait on this lock.

Locks are released automatically when the process completes, but you can also release the lock using the Release Lock step if you need to release it explicitly before continuing later steps in a process. Administrators can also release locks. See Managing Locks [page 108].

The following table describes the property for these steps.

Property	Description
Lock Name	Specify the lock name. The default setting is as follows: `${p:component.name}-${p:componentProcess.name}-${p:resource.name}` **Tip:** The default lock name setting is set using a default variable. Therefore, to make the lock names exactly the same for different objects and processes, you can enter a static string for the name or use variables that will resolve to the same in both processes. For example, if you want to lock a resource against concurrent use for two processes in different components that are mapped to the same resource, you could use `${p:resource.name}` as the lock name for both.

Set Status

The **Set Status** step enables you to explicitly set the status of a process. The following table describes the property for this step.

Property	Description
Status	Values: Success or Failed

Set Property

The **Set Property** step enables you to set a value of a property. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.

Property	Description
Property Name	Enter a name for the property.
Property Value	Enter a value for the property.

Switch

The **Switch** step enables you to define a conditional step. The outbound connections for this step are set to check for values to determine the ongoing path of the process. See Using Conditions and Switch Steps in Processes [page 249]. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.
Property Name	Enter a name for the property to be checked for values in the outgoing connections.

Join

The Join step enables you to join steps in a process. There are no properties for this step.

For the Join step to continue processing on to its outboard connections, all its inbound connection paths must be valid. To be valid, all that should evaluate to success do so, all that should evaluate to failure do so, all that should evaluate to any value do so, and all that should evaluate to a specific switch value do so.

You can do an implicit join without the Join step by having multiple inbound connections for another step. Regardless of whether it is an implicit or explicit join, all incoming connections must be valid for the joining step to be executed.

One reason you should choose an explicit Join step is when your process has multiple steps that must all complete to a certain outcome and then should continue on to multiple outbound steps without additional processing between them.

Manual Task

The Manual Task step enables you to interrupt a process until some manual intervention is performed according to a pre-configured component task. Component tasks must be set for an component before you can select them for this step. See Component Tasks [page 256].

The following table describes the properties for this step.

Field	Description
Name	Enter a descriptive name for the step.
Task Definition	Used to select a previously-defined user-defined task. See Application Tasks [page 289].

Field	Description
Environment Role	Select the role expected to respond. The user mapped to this role must respond to the generated work item before the process can continue.
Component Role	Select the role expected to respond. The user mapped to this role must respond to the generated work item before the process can continue.
Resource Role	Select the role expected to respond. The user mapped to this role must respond to the generated work item before the process can continue.

All roles are required, so all affected users must respond before the process can continue.

Add Inventory Status and Remove Inventory Status

The **Add Inventory Status** and **Remove Inventory Status** steps enable you to add an inventory status as part of your process. See Using Inventory Statuses [page 111].

The following table describes the property for these steps.

Property	Description
Status	Select the status to add or remove.

Run Component Process

The **Run Component Process** step enables you to run a component process as part of the current process. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.
Component Process	Select the component process name.

Run Global Process

The **Run Global Process** step enables you to run a global process as part of the current process. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.
Process	Select the global process name.
Resource Name	Enter the resource name. The default value is as follows: `${p:resource.name}`

Using Expression Language in Processes

You can use Java Unified Expression Language (EL) in your processes in the following Utility Steps:

- Set Property
- Switch
- Run Component Process

For example, if in Set Property you used the expression `${p:propA}+1` in the Property Value field for a property called `propB`, if the value of `propA` was 5, the value of `propB` would be set to 6.

Using Conditions and Switch Steps in Processes

Conditions and switch steps can be used in processes to enable dynamic decision making within the processes.

Conditions

Each connection, except that from the Start step, has a conditional flag. The conditional flag enables you to set a condition on a connection. The condition refers to the processing status of the originating step. Possible flag conditions are:

Success

The process completed successfully. When set, the connection is green.

Failure

The process did not finish successfully. When set, the connection is red.

Always

Accept either status. When set, the connection is gray.

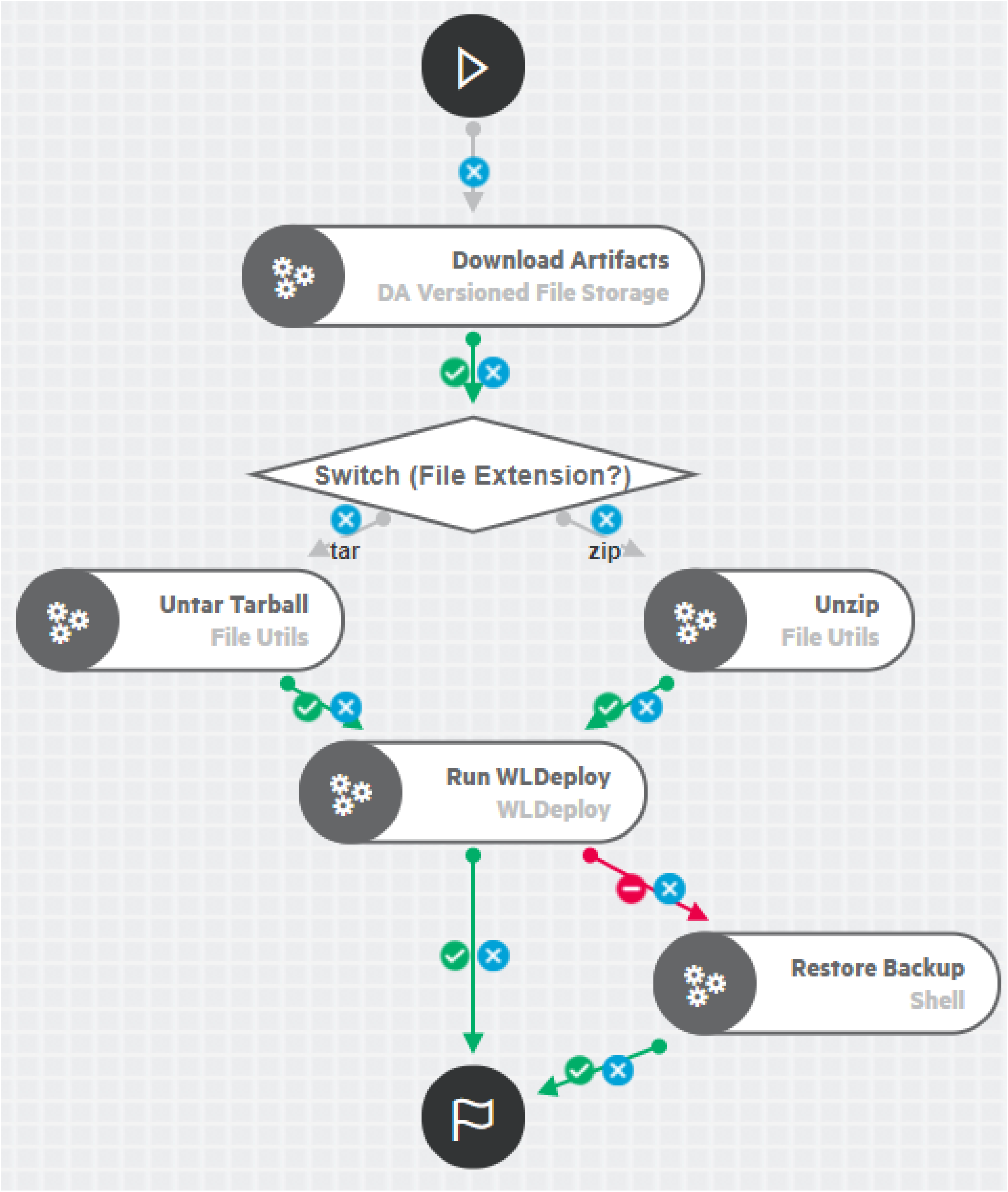
By default, all connections have the flag selected, meaning the originating step must successfully end processing before the target step starts processing.

To change a flag's value, select the connection and cycle through possible values by clicking the flag. You can also select the value in **Properties**.

Process with Switch Step

A *switch step* is a utility step that enables you to branch a process based on the value of a property.

The following figure illustrates a switch step. In this example, the switch property is *version.name*. The connections from the switch step represent process branches depending on the value of *version.name*. In this example, regardless of which branch is taken after the switch, the process converges to the Run WLDeploy step.



Note:

Run WLDeploy has success and fail conditions. See the *Deployment Automation Plugins Guide* for more information.

Switch steps are processed as follows:

- If a step has multiple connections that eventually reach the same target step, determining whether the target will execute depends on the value of the intervening flags. If all of the intervening connections have success flags, the target will only process if all the steps are successful. If the intervening connections consist of an assortment of success and fail flags, the target will process the first time one of these connections is used.
- Regular expression matching is used to evaluate the switch step values. If a value is not a properly-formed regular expression, the step will fail. For example, if your value is a directory name, such as `C:\temp\store`, you must escape the `:` and `\` to make it a valid regular expression as follows:

  ```
  C\:\\temp\\store
  ```

- For a process to succeed, execution must reach a Finish step. If it does not end with Finish, the process will fail every time.

Note: By default, Deployment Automation resolves expressions in Set and Switch steps, but you can configure the server so that it doesn't. For details, refer to Knowledgebase item S141763.

User Impersonation

Deployment Automation can use user impersonation when an agent must execute a command for which it might not otherwise have permission, or when a specific user must be employed for a given process.

For example, to run a database update script, an agent might need to be the "oracle" user; but to update the application, the agent might need to be the "websphere" user. By using impersonation, the same agent can run the script and update the application, which enables you to combine these steps into a single process.

- On UNIX/Linux systems, the `ssh`, `sudo`, or `su` commands are used to impersonate users. See User Impersonation on UNIX/Linux [page 251].
- On Windows, Deployment Automation provides a utility program to handle impersonation. See Impersonation on Windows Systems [page 253].

User Impersonation on UNIX/Linux

For agents running on UNIX/Linux platforms, when you configure a process step you can provide the agents with the user impersonation capability.

You can select `ssh`, `sudo`, or `su` options.

Note: `ssh`, `sudo`, and `su` are short for secure shell, super user "doer", and super user respectively.

When a process step runs that has user impersonation configured, the `ssh`, `sudo`, or `su` command runs the step as the impersonated user.

Process steps can be considered individual shells; the `ssh`, `sudo`, or `su` command enables a user to start a shell as another user.

Note: Each process step that needs user impersonation must be configured independently.

To configure impersonation:

1. In the Process Editor, in the process step's **Properties**, select **Use Impersonation**.
2. Enter the following information:

Field	Description
User	The username to authenticate as during impersonation.
Password	The password to use during impersonation. • For *nix impersonation types: ▪ SSH: The password of user to impersonate. ▪ Sudo: The password of user that the agent is running under (if required). ▪ Su: The password should be blank. • For Windows: The password of user to impersonate.

Field	Description

*nix Impersonation Type	Select `ssh`, `sudo`, or `su`.
	If you select `ssh`, SSH authentication will be used to perform impersonation. You must install, configure, and start the SSH daemon on your agent machines if you want to use this option.
	If you select `su`, it can be used only on agents running under the root account.
	If you select `sudo`, before it can be used, you must provide impersonation privileges as follows:
	• Password Required. Impersonation privileges *must* be defined in the `/etc/sudoers` file along with grant privileges to run scripts from the agent `.temp` folder. For example:
	`User1<>ALL=(User2)/home/User1/agent/var/temp/*`
	Grants *User1* the privilege to impersonate *User2* to run plugin steps as *User2*.
	`Defaults:X!requiretty`
	`X ALL=(Y)`
	where *X* and *Y* are user names, and user *X* can run any command as user *Y*.
	• No Password Required. Impersonation privileges *must* be defined in the `/etc/sudoers` file. For example:
	`Defaults:X!requiretty`
	`X ALL=(Y) NOPASSWD: ALL`
	where *X* and *Y* are user names, and user *X* can run any command as user *Y* without supplying a password.
	See the UNIX/Linux documentation for more information on configuring `ssh`, `sudo`, and `su`.

Impersonation on Windows Systems

For agents running on Windows platforms, Deployment Automation provides a program that handles impersonation.

You implement impersonation for Windows-based agents the same way you do for UNIX- or Linux-based agents. When you configure a process step, you specify the credentials that will be used to login on the agent when the step is processed. This is a different user than the user under which the agent normally runs.

To run process steps on a Windows agent, the user must:

- have a user name and password stored on the target agent computer

- be part of the Administrators group
- have, at a minimum, the following privileges:

```
SE_INCREASE_QUOTA_NAME (adjust memory quotas for a process)
SE_ASSIGNPRIMARYTOKEN_NAME (replace a process-level token)
SE_RESTORE_NAME (Restore files and directories)
SE_BACKUP_NAME (Back up files and directories)
SE_TCB_NAME (Act as part of the operating system; Required for Windows Vista and later
```

In addition, they must have at least one of the following logon permissions.

```
SE_INTERACTIVE_LOGON_NAME (Log on locally)
SE_SERVICE_LOGON_NAME (Log on as a service)
SE_BATCH_LOGON_NAME (Log on as a batch job)
```

Adding Component Process Properties

Add properties to a component process for information that may have values specific to the scope of that process.

To define a component process property:

1. Navigate to **Management > Components > [select component] > Processes**.
2. In the processes list, next to the process for which you want to define a property, click the **Details** icon (👁).
3. On the **Properties** tab, click the **Add Property** button.
4. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.

Field	Description
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

5. Click **Save**.

You can use this property in the same process where it is defined by referencing it in one of the steps. Use the following syntax: `${p:<property name>}`

Plugin Post Processes

When a plugin step finishes processing, its default post-processing element is executed. The post-processing element is defined in the plugin's XML definition.

You can override the default behavior by entering your own script into the step's **Post Processing Script** field. A post-processing script can contain any valid JavaScript code. Although not required, it's recommended that scripts be wrapped in a `CDATA` element.

See the *Deployment Automation Plugins Guide* for more information.

Running Component Processes

You can run a component process directly or initiate it as part of an application or deployment package process. This topic tells how to run it directly.

No matter how you run a component process, you must identify the application you want to use for the process, because within the applications you map specific environment resources to components. For more about deployment, see Chapter 28: Deployment Overview [page 365].

To run a component process directly:

1. Navigate to **Management > Applications > [select application]**.

2. Select the **Components** tab.

3. Click **Run Component Process** for the component where you want to run a process.

4. Select an environment to deploy to.

5. Select a resource to use.

6. Select the process to run.

7. Select other options.

8. Click **Submit**.

When a process starts, the **Request** page is displayed so that you can review the process request progress. After the request completes, you can review the results on that and other related pages. See Chapter 30: Analyzing Deployments [page 371].

Note: If gate conditions are defined for a target environment, the component version is deployed into that environment only if its status meets those conditions. See Gates [page 295].

Component Tasks

A component task is used to interrupt a component process until some manual intervention is performed. A task-interrupted process will remain suspended until the targeted user or users respond. Typically, component tasks are removed after the process has been tested or automated.

Like approvals, triggered component tasks alert targeted users. Alerts are associated with component-, environment-, or resource-defined user roles. Unlike approvals, component tasks can be incorporated within a component process.

The person alerted for the task can indicate that it is complete by approving the step from the request's log in **Approval Requests**. See Approving Requests [page 291].

Create and use tasks as follows:

1. Create component tasks in components.

2. Insert a manual task step in a component process.

   See Manual Task [page 247].

Creating Component Tasks

To create a component task:

1. Navigate to **Components > [select component] > Tasks**.

2. Click the **Create** button.

3. Name the task then select a template from the **Template Name** field.

   The individual tasks map to the notification scheme used by the application (see Creating Notification Templates [page 147]). If a scheme is not specified, the default scheme is used. The available task notification template is:

   - TaskCreated

4. Click **Save**.

Creating Component Templates

Component templates enable you to save and reuse component processes and properties and create new components from them; template-based components inherit the template properties and processes.

To create a template:

1. Navigate to **Management > Components**.
2. Click the **Component Template** tab and select **Create**.
3. Enter the template's name in the **Name** field.
4. Enter a description in the **Description** field.

   The optional description can be used to convey additional information about the template.
5. Select the source location of your artifacts in **Source Config Type**, and then complete the additional fields specific to that source configuration type. See Source Configuration Type Fields [page 203].
6. To continue creating component templates and stay in the list, click **Save**. To start configuring the component template you just created, click **Build Out**.

You create a process for the template in the same way processes are created for components. For information about creating component processes, see Component Processes [page 238].

Importing/Exporting Component Templates

Component templates can be exported and then imported into another server. This can be useful if you have multiple Deployment Automation servers, such as test and production servers, where you create and validate the component templates on the test server and then export them and import them into the production server.

- Exporting Component Templates [page 257]
- Importing Component Templates [page 258]

Related Topics

Chapter 27: Replication [page 351]

Exporting Component Templates

Exporting a template creates a JSON file that contains the template's configuration information, properties, and processes.

To export a template:

1. Navigate to **Management > Components**.
2. Click the **Component Templates** tab.

3. In the column beside a component template, click the **Export** icon. You can load the file into a text editor or save it. If you save it, a file is created with the same name as the selected component, such as `helloWorldTemplate.json`.

Importing Component Templates

When you import a component template, you can create an entirely new template or upgrade an existing one.

To import a component template:

1. Navigate to **Management > Components**.
2. Select the **Component Templates** tab and click **Import**.
3. If you want to upgrade an existing template, select **Upgrade Template**.

   If the template's name in the JSON file (not the name of the file itself) matches an existing template, the template will be upgraded. If the name is not found, the command has no effect.

   Note: The template's name is the first parameter in the JSON file. For example:

   ```
   "name": "helloWorldTemplate"
   ```

4. In the **Choose File** field navigate to the JSON file containing the template.
5. Click **Import**.

Adding Component Template Properties

Component template properties ensure that every component created from a template has the same properties. The three types of available properties are described in the following table.

To add component template properties:

1. Navigate to **Management > Components > Component Templates.**
2. Select a template and select the **Properties** tab.
3. Select one of these component template property tabs and click **Add Property**:

- **Properties**

  These user-defined properties can be used in scripts and plugins. Properties inherited from templates cannot be modified on the component level. If you change the value, the change will be reflected in components created from the template, including those previously created.

- **Component Property Definitions**

  Every component will have this property; it will appear on the Create Component dialog box for every component created from this template. A value defined here can be changed by created components. Each property must have a type.

- **Environment Property Definitions**

Every environment that uses a component created by this template will have this property. The property will appear on the environment's Component Mappings at **Applications > [select application] > Environments > [select environment] > View Details > Component Mapping**. A value defined here can be changed by environment. Each property must have a type.

- **Version Property Definitions**

Type

You must select a type for each component and environment property. The available types are as follows:

- *`Text`*

  Enables users to enter text characters.

- *`Text Area`*

  Enables users to enter an arbitrary amount of text, limited to 4064 characters.

- *`Check Box`*

  Displays a check box. If selected, a value of `true` will be used; otherwise the property is not set.

- *`Select`*

  Requires a list of one or more values that will be displayed in a drop-down list box. Enables a single selection.

- `Multi Select`

  Requires a list of one or more values that will be displayed in a drop-down list box. Enables multiple selections.

- `Secure`

  Used for passwords. This is similar to *`Text`* except values are redacted.

Using Component Templates

When you create a component based on a template, the component inherits any processes and properties from the template.

To create a template-based component:

1. Navigate to **Management > Components > Create**.

   The Create Component dialog is used to configure the component.

2. From the **Template** list select a component template. Properties defined in the template are predefined. If a source was selected in the template, the source is set here and the Source Config Type field is locked. For information about using this dialog, see Creating Components [page 200].

3. After configuring editable properties, save the component.

Templates used to create components are listed in the Templates view.

Components created from templates are listed in the Components view.

Configuration Templates

You can use configuration templates to save and reuse configuration data in processes. Typically, the data is for server configurations, for example on Tomcat servers, but the data can be for any purpose.

Preparing to Use Configuration Templates

Before you can use configuration templates, you must do the following preparation steps.

To prepare for using configuration templates:

1. Load the `DAConfigManagement` plugin.
2. Create a configuration template for your component.
3. In a component process, use the *Resource Tools > Configuration Management > Install Template* step to download the template from the Deployment Automation server and do token replacement based on environment or other properties.

Note: Configuration templates are not the same as component templates. For information on component templates, see Creating Component Templates [page 257].

For information on working with configuration templates, see the following topics:

- Creating Configuration Templates [page 260]

Creating Configuration Templates

After you have prepared to use configuration templates, you can create them.

To create a configuration template:

1. Navigate to the **Management > Components > [select component] > Templates**.
2. Click **Create**.
3. Enter a name in the **Name** field.
4. In the **Template** field, enter or paste the template text. The amount of text is based on the database used by Deployment Automation. In general, there is no limit to the amount of text used for a configuration template. For an example, see the Community website.
5. Click **Save**.

To edit a configuration template, click the **Edit** icon beside it in the list.

Deleting and Deactivating Components

Components can be deactivated or deleted. To delete or deactivate a component, use the desired action on the Components page for the intended component.

When a component is deactivated, it remains in the database and CodeStation and can be activated later. To activate a component, first click the **Show Inactive Components** check box, then use the **Activate** action for the component.

When a component is deleted, it is removed, along with all versions, from the database and CodeStation and cannot be activated at a later time. The original artifacts are not affected; only the CodeStation copies are deleted.

Note: Components cannot be deleted if they are used by an application. To delete a component used by an application, first remove the component from the application.

Chapter 19: Managing Applications

Applications bring together components with their deployment targets and orchestrate multi-component deployments.

Applications have *processes* that are used to automate deployments and rollbacks, and application *snapshots* are used to manage versions of applications.

Note: Before configuring an application, you should ensure that at least one agent has been installed and mapped to a target environment. See Chapter 22: Managing Resources [page 307].

For details on managing applications, see the following topics:

Related Topics

Chapter 20: Managing Environments [page 297]

Chapter 22: Managing Resources [page 307]

Chapter 18: Managing Components [page 199]

A Typical Sequence [page 24]

Applications Page Overview

You typically manage applications in the Applications page of the user interface. Following are some of the things you can do from this page.

- Filter the applications created by you or by all users.

- Filter by Success or Failure.
- Toggle the **Inactive** button to include or exclude inactive applications.
- Click the **Applications Actions** button to create or import applications.
- Click the name to open the page for a listed application.
- Click an icon beside an application to do the following:
  - Edit
  - Inactivate/Activate
  - Export
  - Copy

Creating Applications

You can create a new application or import an existing one. See Importing/Exporting Applications [page 267] for information about importing applications.

To create an application:

1. Navigate to **Management > Applications**.
2. Click **Create**.
3. Enter the information needed to create the application as follows:

Field	Description
Name and Description	Enter a name and description. These typically relate to the real-world application you plan on deploying.
Notification Scheme	Select a notification scheme if you want to send email notifications based on triggering events, such as when an application deployment fails or succeeds or when a request needs approval. See Configuring Email Notifications [page 144].
Pipeline	Select the pipeline you want to use for this application's process requests. See Chapter 21: Managing Pipelines [page 303].
Enforce Complete Snapshots	If selected, the application requires every component to be versioned.

4. To create additional applications click **Save and New**. To start configuring the application you just created, click **Build Out**.

Application Configuration

After you create an application, select it in the **Applications** page to configure it. Following are some of the things you can do from this page.

- **Activity:** View a graphical roll-up of activity for the selected application, including deployment summary, next scheduled deployment, and deployments to environments by process.

- **Details:** View and edit the application details, including designating the pipeline. From this page you can also add application properties.

- **Components:** After defining the application, identify the components it will manage. Associating a component makes its processes and properties available to the application. An application can have any number of components associated with it.

- **Environments:** Add environments for the application to use and map the environments' resources to components. Mapping means assigning an agent to manage the component. Each component can be mapped to the same agent, a different one, or some combination. An application can have more than one environment associated with it.

- **Processes:** Create application processes using the process editor, with a toolkit of process steps specifically for application processes. Previously defined component processes can be incorporated into the process.

- **Tasks:** Configure manual tasks that you will later incorporate into an application process.

- **Snapshots:** Capture information about the application for a given environment, including the component versions, process versions, and properties.

- **History:** View application deployment and change history.

- **Security:** Set role membership and default permissions for this application.

- **Run:** Run application and component processes.

- **Action**: Select from actions for this application, including the following:
  - **New:** Property, Component, Process, Task, Snapshot
  - **Add:** Component, Environment
  - Export
  - Print Page
  - Help

Adding Components to Applications

Add at least one component to the application. Applications bring the different components (their versions and processes) together so they can be deployed as a single unit.

To add components to an application:

1. Navigate to **Management > Applications > [select application] > Components**.
2. Click **Add Component**.
3. Select one or more components and click **Add**.

Adding Application Properties

Add properties to an application for information that may have values specific to that application. An application property value would override the value for the same-named property at the system level.

To add a property to the selected application:

1. Navigate to **Management > Application > [select application]**.
2. Select the **Properties** tab.
3. To set a single property, click **Add Property**.
   a. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Secure	Select this if the property should be secure. Secure properties are stored encrypted and displayed obscured in the Deployment Automation user interface.
Value	Value for the property that will be used by steps in a process

4. To set multiple properties, click **Batch Edit**.
   a. In the **Properties** box, enter multiple properties and their values on separate lines as property=value.

   For example, if you had properties `host` and `port`, you would enter them as follows:

```
host=myServer

port=8080
```

Note:

The following restrictions apply to batch edit:

- The **Secure** option cannot be set.
- The value cannot contain the "=" symbol.

5. Click **Save**.

To use the application property in a process, reference it when you configure a step that uses it. Use the following syntax: `${p:application/<property name>}`

Copying Applications

You can create a new application based on an existing application and run the same operations as the original. Application snapshots and component versions are not copied. The copied application contains copies of:

- Environments
- Processes
- Task definitions
- Property sheets
- Security and application settings

By default, the new application references the original components and processes. Modifications made to the original components affect the behavior of the copied application.

You can optionally create a copy of an application that does not reference existing components. Modifications made to the original components do not affect the copied application.

To copy an application:

1. Navigate to **Management > Applications**.
2. Select the application's row and click **Copy**.
3. Enter a name and description for the new application.
4. To specify new names for the components that will be copied and not reference the original components, select **Deep copy components**. Enter a new name for each component.
5. Click **Save**.

Importing/Exporting Applications

Applications can be exported and then imported into another server. This can be useful if you have multiple Deployment Automation servers, such as test and production servers, where you create and validate the applications on the test server and then export them and import them into the production server.

- Exporting Applications [page 268]
- Importing Applications [page 268]

Related Topics

Chapter 27: Replication [page 351]

Exporting Applications

Exporting an application creates a JSON file (file extension `json`) that contains the application's properties, processes, and components and their associated properties and processes. For information about JSON, see http://www.json.org/.

To export an application:

1. Navigate to **Management > Applications**.
2. Select the application's row and click **Export**.

You can load the file into a text editor, or save it. If you save it, a file is created with the same name as the selected component, such as `helloWorldApplication.json`.

Importing Applications

When you import an application, you can create a new application or upgrade an existing one. Components associated with the application are also imported, including the component properties and processes if they are available to the importing server.

Note: If imported components have **Import Versions Automatically** selected, Deployment Automation will automatically import component versions as long as the artifacts are accessible to the importing server.

To import an application:

1. Navigate to **Management > Applications**.
2. Click **Import**.
3. If you want to upgrade an existing application, select **Override Application**.

   If the application's name in the JSON file (not the name of the file itself) matches an existing application, the application's parameters are updated with new values, and new items, such as processes, environments, and components, are added. If the name is not found, the command has no effect.

   Note: The application's name is the first parameter in the JSON file. For example:

   ```
   "name": "helloWorldApplication",
   ```

4. Specify how imported components should be handled with the **Component Upgrade Type** options. For these options, the components must be on the importing server.

   - To use the same components used by the imported application, select **Use Existing Component**. The new application will contain references to the imported applications components. This option is especially useful if you are importing a lot of applications.

     If you are upgrading, the application will use the imported components, and no longer use any not used by the imported application.

   - To create new components based on those used by the imported application, select **Create Component**. New components will be created (based on the imported application's components).

    If you are upgrading, the application will use the newly created components and no longer use any it previously used.

  - When you want to create a fresh installation, select **Fail if Component Exists**. If you are creating an application, it will create both a new application and component unless the component already exists, in which case the application is not imported.

    If you are upgrading, the upgrade will fail if any imported components already exist on the importing server.

  - To ensure a component is on the importing server, select **Fail if Component Does Not Exist**. If you are creating an application, it will create both a new application and component unless the component does not exist, in which case the application is not imported.

    If you are upgrading, the upgrade will fail if an imported component does not already exist on the importing server.

  - To upgrade existing components, select **Upgrade if Exists**. This option creates an application and upgrades existing components with data from the imported application.

    If you are upgrading and existing components match imported ones (all must match), the components will be upgraded. If none of the imported components match existing ones, the imported components will be used.

5. Click **Choose File** and select the JSON file containing the application.

6. Click **Import**.

Application Environments

After you have created environments in the **Environments** page, you can add them to applications individually or as part of a pipeline. On an **Environments** page you can:

- Add more environments to an application.

- Scroll a graphical view of environments. Any pipeline is indicated by a shaded box; any individual environments follow.

- Click an environment name and select from a menu of options for that environment. Only relevant options appear. Possible options are:

  - **Run Process:** Run an application process in this environment.

  - **Promote:** Execute the active process in the next environment in the pipeline. This appears only for environments in pipelines are mandatory but without auto promote.

  - **View Details:** View and configure application-specific settings for the selected environment.

  - **Compare:** Compare the inventory of up to four environments.

  - **Copy:** Create an environment based on this environment. Upon creation, the environment is automatically added to the application.

  - **Inactivate:** Set the environment to inactive status. Inactive environments are not shown in the list by default. You can use the **Show Inactive** check box to show them in the list.

- Click **Resources** to get a quick view of the resources associated with an environment. From there you can click **View Resources** to open the resources tab for the environment.

- See a breakdown of the number of environments in online, offline, and unknown status.

- Click the double arrows beside the pipeline to contract or expand the pipeline in the view.

- Move individual environments to another position.

  Note: Pipelines cannot be moved. Only one pipeline can be associated with an application at a time and it always comes before any individual environments.

- Filter the components shown in the list by using search criteria.

- View environment inventory and compliancy.

For details on creating environments and configuring global settings, see Chapter 20: Managing Environments [page 297].

For more details about the environment inventory, see Viewing the Environment Inventory [page 377].

For more details on configuring application-specific environment settings, see the following topics:

- Adding Environments to Applications [page 270]
- Adding Application Environment Properties [page 271]
- Mapping Application Components to Resources [page 273]
- Setting Component Environment Property Values [page 273]
- Comparing Environments [page 274]

Adding Environments to Applications

After you create an environment, you add it to an application. Environments may be used by multiple applications.

Note: When you add a pipeline to an application, the environments in the pipeline are automatically added to the application. Environments that are part of a pipeline are grouped together in a section labeled with the pipeline name.

To add an environment to an application:

1. Navigate to **Management > Applications > [select application] > Environments**.

2. Click the **Add Environment** button.

3. Enter the environment information as follows:

Field	Description
Environment	Select an environment. For information on creating environments, see Creating Environments [page 298].
Require Approvals	To require an approval before components can be deployed to the environment, select this check box. If selected, Deployment Automation will enforce an approval process before the deployment can be deployed to the environment. Initial deployments are typically done in uncontrolled environments, but once the deployment is successful, you can configure an approvals process as the application moves along the development pipeline. If you are setting up more than one environment, consider creating an approvals process for at least one of them.
Lock Snapshots	If you want all snapshots used in this environment to be locked to prevent changes, select the check box.
Inherit Cleanup Settings	Determines how many component versions are kept in CodeStation, and how long they are kept. If selected, the component will use the values specified in **Administration > System > System Settings**. If deselected, additional fields appear.
Days to Keep Versions	If Inherit Cleanup Settings is not selected, this field appears, which enables you to define custom values. This is initially set to *-1*, keep indefinitely.
Number of Versions to Keep	If Inherit Cleanup Settings is not selected, this field appears, which enables you to define custom values. This is initially set to *-1*, keep indefinitely.

4. Click **Save**.

Adding Application Environment Properties

You can define environment properties at the application environment level. The application environment properties are specific to the application's use of an environment. Environment properties added at this level appear under the application environment's **Properties** tab, and not under the environment's **Properties** tab, because they are set for this application only.

Environment property values for same-named properties are resolved at the most specific level. An environment property value set at the application level overrides one with the same name set directly on the environment property level. An environment property value set at the component level overrides any with the same name set directly on the environment property level or on the application environment level.

If component environment properties have been added for a component mapped to the environment and you want to set those values, see Setting Component Environment Property Values [page 273].

To create application environment properties:

1. Navigate to **Management > Application > [select application]**.
2. Select the **Environments** tab.
3. Click the environment name to which you want to add properties and select **View Details** from the list.
4. Select the **Properties** tab.
5. To set a single property, click **Add Property**.
   a. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Secure	Select this if the property should be secure. Secure properties are stored encrypted and displayed obscured in the Deployment Automation user interface.
Value	Value for the property that will be used by steps in a process

6. To set multiple properties, click **Batch Edit**.
   a. In the **Properties** box, enter multiple properties and their values on separate lines as property=value.

   For example, if you had properties `host` and `port`, you would enter them as follows:

```
host=myServer

port=8080
```

Note:

The following restrictions apply to batch edit:

- The **Secure** option cannot be set.
- The value cannot contain the "=" symbol.

7. Click **Save**.

To use a property in a process, reference it when you configure a step that uses it. Use the following syntax: `${p:environment/<property name>}`

Mapping Application Components to Resources

Component processes are executed by Deployment Automation agents running on hosts. One instance of a component process is invoked for each resource mapped to a component in the target environment.

After you have added a component to the application, define where its artifacts should be deployed by selecting a resource (agent) or resource group. See Chapter 22: Managing Resources [page 307].

1. Navigate to **Applications > [select application]**.
2. Select the **Environments** tab.
3. Click the environment name to which you want to map components and select the **View Details** option.
4. Select the **Component Mapping** tab.
5. If the application has several components associated with it, select the one you want to use from the component list on the left. Each component associated with this application can be mapped to a different resource.
6. To associate resources with the selected component, click **Add**, and select one of the following:
   - **Add Resource To All Components** and select one or more resources.
   - **Add Resource Group To All Components** and select a resource group. For information about creating resources, see Resource Groups [page 319].
   - To map multiple resources at once, use the following options:
     - Add all resources and groups to all the components
     - Add all resources and groups to the selected component
     - Remove all resources and groups from all the components
     - Remove all resources and groups from the selected component

Setting Component Environment Property Values

After you have added component environment properties at the component level, you can set the values for them for each environment to which the component is mapped.

To set component environment properties:

1. Navigate to **Management > Applications > [select application]**.
2. Select the **Environments** tab.
3. Click the environment name to which you want to add property values and select **View Details**.
4. Select the **Component Mapping** tab.
5. Select the **Properties** tab.

6. Click **Edit**.
7. Edit values for the environment properties that were previously defined for the component. See Adding Component Environment Properties [page 227].
8. Click **Save**.

You can reference the property value in processes using `${p:environment/<property name>}`.

Comparing Environments

You can compare environments to each other to see how their inventories differ.

To compare environments:

1. Navigate to **Management > Applications > [select an application]**.
2. Select the **Environments** tab.
3. Click the **Compare** button.
4. Select the environments that you want to compare. You can compare up to four environments at a time.
5. Click **Compare**. The inventories of the selected environments are compared and the differences are displayed.
6. In the selection box, select from the following views to see the associated comparative information:
   - Versions
   - Files
   - Configurations

Application Processes

Application processes typically initiate component processes defined for their associated components.

Application processes can run manually, automatically on some trigger condition, or on a user-defined schedule. When a component has several processes defined for it, the application determines which ones are executed and in which order. For example, an *n*-tiered application might have a web tier, a middleware tier, and a database tier. The database tier must be updated before the other two, which are then deployed concurrently. An application can orchestrate the entire process, including putting servers on- and off-line for load-balancing as required.

When an application process executes, it interacts with a specific environment. At least one environment must be associated with the application before the process can be executed.

Application processes can be designed independently of any particular environment. This enables a single application to interact with separate environments, such as QA or production. To use the same application process with multiple environments, you can associate the application with a pipeline, associate individual environments with the application, or a combination of both.

In addition to deployments, several other common processes are available, such as rolling back deployments. Deployment Automation tracks the history of each component version, which enables application processes to restore environments to any point.

Application processes can be initiated as part of deployment packages to run multiple-application deployments. See Chapter 23: Managing Deployment Packages [page 325].

For information on creating and designing application processes, see the following topics:

- Creating Application Processes [page 275]
- Designing Application Processes [page 276]
- Application Process Step Details [page 277]
- Adding Application Process Properties [page 287]
- Running Application Processes [page 288]

Creating Application Processes

Application processes, like component and global processes, are created with the process editor.

To create an application process:

1. Navigate to **Management > Applications > [select application].**
2. Select the **Processes** tab and click **Create.**
3. Enter the following information:

Field	Description
Name/ Description	Typically the name and description correspond to the application you plan on deploying.
Required Application Role	Select the role a user must have in order to run the application. For information about creating application roles, see Role Configuration [page 116]. The default value is `None`.
Inventory Management	To have inventory handled automatically, leave the default value, `Automatic`, selected. If you want to handle inventory manually, select `Advanced`. See Chapter 31: Inventory Management [page 377].

Field	Description
Offline Agent Handling	Select the option for how the process should react if expected agents are offline: **Check Before Execution** Checks to see if expected agents are online before running the process. If agents are offline, the process will not run. **Use All Available; Report Failure:** The process will run as long as at least one agent defined in the environment is online; this reports any failed deployments due to offline agents. This is useful for rollbacks or configuration deployments. **Always Report Success** The process will run as long as at least one agent defined in the environment is online; this reports successful deployments.
Default for Only Changed Versions	If this is selected, the **Only Changed Versions** option will be selected by default in the Run Application Process wizard. If it is deselected, the **Only Changed Versions** option will be deselected by default. The **Only Changed Versions** option can be selected or deselected in the wizard regardless of the default setting. If you want to change the default setting of **Default for Only Changed Versions**, see Configuring the Default for Only Changed Versions Option [page 175].
Execution Timeout	Specify how long the process will execute for in hours and minutes. Default: indefinitely

4. To create additional application processes, click **Save and New**. To start configuring the application process you just created, click **Build Out**.

Designing Application Processes

Application processes tell Deployment Automation what to do when you deploy whole applications.

You design processes using the process editor. In the process editor you drag process steps from a menu of steps onto the design space and configure them immediately. As you place additional steps, you visually define their relationships with one another.

For application processes, you select from a menu of utility steps and other steps such as installing components. These enable you to operate on sets of components associated with the application.

Application Process Design

1. Select from a set of options for the process designer, including the following:
    - Zoom In
    - Zoom Out
    - Actual Size
    - Print
    - Poster Print
    - Save
    - Cancel
    - Show Outline Box: Use this to move the whole process design around in the design area. For example:
        - Drag its corner inward and move it over parts of the process design to zoom in.
        - Drag its corner outward again to zoom out.
    - Show Connection Options: Select this to show the options for all connections.
2. In **Tools**, select process steps.
3. Use the Search box to find a step by name.
4. Select and drag steps to the design area to add them to your process.
5. Connect process steps and edit or delete existing steps or connections, see Connecting Process Steps [page 242].
6. **Properties** are automatically shown when you select a step.

Application Process Step Details

After you create an application process, select it to design the process steps. The application process steps are described in the following topics:

- Finish [page 278]
- Install Component [page 278]
- Uninstall Component [page 278]
- Rollback Component [page 279]
- Run Process for Each Version [page 280]
- Apply Configuration [page 281]
- Application Process Snapshot Steps [page 282]
- Application Process Utility Steps [page 284]

Finish

The **Finish** step ends processing. A process can have more than one **Finish** step.

Install Component

The **Install Component** step installs the selected component using one of the processes defined for the component.

Install Component Properties Table

Field	Description
Name	Can be referenced by other process steps.
Component	Component used by the step; a step can affect a single component. All components associated with the application are available. If you want to install another component, add another install step to the process.
Use Versions Without Status	Restricts the components that can be used by the step. Components with the selected status are ignored. Available statuses: `Active` means ignore components currently deployed; `Staged` means ignore components currently in pre-deployment locations.
Component Process	Select a process for the component selected above. All processes defined for the component are available. Only one process can be selected per step.
Ignore Failure	When selected, the step will be considered to have run successfully.
Limit to Resource Role	User-defined resource role the agent running the step must have.
Fail Fast	When this is selected, this step will not start any more component processes if one fails.
Run on First Online Resource Only	Instead of being run by all agents mapped to the application, the step will only be run by the first online agent identified by Deployment Automation. The mechanism used to identify the "first" agent is database-dependent (thus indeterminate).
Precondition	A JavaScript script that defines a condition that must exist before the step can run. The condition must resolve to true or false.

Uninstall Component

The **Uninstall Component** step uninstalls the selected component.

Uninstall Component Properties Table

Field	Description
Name	Can be referenced by other process steps.
Component	Component used by the step; a step can affect a single component. All components associated with the application are available. If you want to uninstall another component, add another uninstall step to the process.
Remove Versions With Status	Restricts the components that are affected by the step, only components with the selected status are affected. Available statuses: `Active` means use components currently deployed; `Staged` means use components currently in pre-deployment locations.
Component Process	Select a process for the component selected above. All processes defined for the component are available. Only one process can be selected per step.
Ignore Failure	When selected, the step will be considered to have run successfully.
Limit to Resource Role	User-defined resource role the agent running the step must have.
Run on First Online Resource Only	Instead of being run by all agents mapped to the application, the step will only be run by the first online agent identified by Deployment Automation. The mechanism used to identify the "first" agent is database-dependent (thus indeterminate).
Precondition	A JavaScript script that defines a condition that must exist before the step can run. The condition must resolve to true or false.

Rollback Component

The **Rollback Component** step rolls back a component version and replaces it with an earlier one.

Rollback Component Properties

Field	Description
Name	Can be referenced by other process steps.
Component	Component used by the step; a step can affect a single component. All components associated with the application are available. If you want to roll back another component, add another rollback step to the process.

Field	Description
Remove Versions With Status	Restricts the components that are affected by the step, only components with the selected status are affected. Available statuses: `Active` means use components currently deployed; `Staged` means use components currently in pre-deployment locations.
Component Process	Select a process for the component selected above. All processes defined for the component are available. Only one process can be selected per step.
Ignore Failure	When selected, the step will be considered to have run successfully.
Limit to Resource Role	User-defined resource role the agent running the step must have.
Rollback type	Determines the type of rollback. Available statuses: `Remove Undesired Incremental Versions` and `Replace with Last Deployed`.
Run on First Online Resource Only	Instead of being run by all agents mapped to the application, the step will only be run by the first online agent identified by Deployment Automation. The mechanism used to identify the "first" agent is database-dependent (thus indeterminate).
Precondition	A JavaScript script that defines a condition that must exist before the step can run. The condition must resolve to true or false.

Run Process for Each Version

The **Run Process for Each Version** runs a selected component process for each component version.

Run Process for Each Version Properties

Field	Description
Name	Can be referenced by other process steps.
Component	Component used by the step; a step can affect a single component. All components associated with the application are available. If you want to run a process for another component, add another Run Process for Each Version step to the process.
Component Process	Select a process for the component selected above. All processes defined for the component are available. Only one process can be selected per step.

Field	Description
Ignore Failure	When selected, the step will be considered to have run successfully.
Limit to Resource Role	User-defined resource role the agent running the step must have.
Max # of concurrent jobs	The maximum number of concurrent component processes to run at any time. Use -1 for unlimited.
Fail Fast	When this is selected, this step will not start any more component processes if one fails.
Run on First Online Resource Only	When this is selected, this step will only use one resource. It will not run on all resources.
Precondition	A JavaScript script that defines a condition that must exist before the step can run. The condition must resolve to true or false.

Apply Configuration

The **Apply Configuration** step applies configuration properties to a selected component version.

Apply Configuration Properties

Field	Description
Name	Can be referenced by other process steps.
Component	Component used by the step; a step can affect a single component. All components associated with the application are available. If you want to apply a configuration to another component, add another Apply Configuration step to the process.
Component Process	Select a process for the component selected above. All processes defined for the component are available. Only one process can be selected per step. **Note:** Only processes set as Operational (No Version Needed) or Configuration Deployment can be used here.
Ignore Failure	When selected, the step will be considered to have run successfully.

Field	Description
Limit to Resource Role	User-defined resource role the agent running the step must have.
Max # of concurrent jobs	The maximum number of concurrent component processes to run at any time. Use -1 for unlimited.
Fail Fast	When this is selected, this step will not start any more component processes if one fails.
Run on First Online Resource Only	When this is selected, this step will only use one resource. It will not run on all resources.
Precondition	A JavaScript script that defines a condition that must exist before the step can run. The condition must resolve to true or false.

Application Process Snapshot Steps

Several snapshot steps are provided for application processes to enable you to dynamically work with snapshots as part of your application processes. These include the following:

- Create Snapshot Step [page 282]
- Update Snapshot Step [page 283]
- Add Status to Snapshot Step [page 283]
- Remove Status from Snapshot Step [page 284]

Create Snapshot Step

The **Create Snapshot** step enables you to create snapshots as part of your application processes. The following table describes the property for this step.

Property	Description
Snapshot Name	Enter a unique name for the snapshot to be created. The default is `${p:datetime}`. **Tip:** Combine text with the predefined date and time property to make a unique name, such as `Appname-${p:dateTime}-auto`. This will avoid step failure from duplicate snapshot names.
Description	Enter the description of the snapshot to be created.

Property	**Description**
Skip this step for snapshot process requests	If selected, this step will be skipped for processes requested for snapshots. This option does not apply to processes requested for component versions.

Update Snapshot Step

The **Update Snapshot** step enables you to update snapshots as part of your application processes. The following table describes the property for this step.

Property	**Description**
Snapshot Name	Enter the name of the snapshot to be updated.
Description	Enter the description of the snapshot to be updated.
Create a new snapshot if one does not exist	If selected, this step will create a new snapshot if a snapshot with the name specified does not exist.
Reset snapshot configuration to latest	If selected and a snapshot updated, its configuration will be reset to all of the latest versions.

Add Status to Snapshot Step

The **Add Status to Snapshot** step enables you to add statuses to snapshots as part of your application processes. The following table describes the property for this step.

Property	**Description**
Status	Select the status to be added to the snapshot.
Snapshot Name	Select the snapshot to which the status should be added. Select from the list of application snapshots or one of the following: • Snapshot from step name Add the status to the snapshot created in a preceding **Create Snapshot** step in the same process. • Use request snapshot Add the status to the snapshot for which you have requested this process. **Important:** If you select the **Use request snapshot** option, before executing the application progress request, ensure that the snapshot you have selected for the process request is configured to always use latest version. Otherwise, this step will never be executed.

Remove Status from Snapshot Step

The **Remove Status from Snapshot** step enables you to remove statuses from snapshots as part of your application processes. The following table describes the property for this step.

Property	Description
Status	Select the status to be removed from the snapshot.
Snapshot Name	Select the snapshot from which the status should be removed. Select from the list of application snapshots or one of the following: • Snapshot from step name Remove the status from the snapshot created in a preceding **Create Snapshot** step in the same process. • Use request snapshot Remove the status from the snapshot for which you have requested this process. **Important:** If you select the **Use request snapshot** option, before executing the application progress request, ensure that the snapshot you have selected for the process request is configured to always use latest version. Otherwise, this step will never be executed.

Application Process Utility Steps

Several utility steps are provided for application processes.

Note: Processes that you have defined that are configuration only and are not related to any component also appear here to be selected for application processes.

Acquire Lock and Release Lock

The **Acquire Lock** and **Release Lock** steps enable you to lock and release a process in a designated environment.

Locks are typically used to prevent specific processes from modifying the same resource at the same time, such as when a database is being updated.

In an Acquire Lock step, the process tries to acquire a lock with the specified name. If there is no other lock with that exact name, the process acquires the lock with that name and continues. If another process is using a lock with that exact name, the process waits until the lock is released by the other process. Any processes that access the resource without a lock or with a lock of a different name do not wait on this lock.

Locks are released automatically when the process completes, but you can also release the lock using the Release Lock step if you need to release it explicitly before continuing later steps in a process. Administrators can also release locks. See Managing Locks [page 108].

The following table describes the property for these steps.

Property	Description
Lock Name	Specify the lock name. The default setting is as follows: `${p:application.name}-${p:applicationProcess.name}-${p:environment.name}` **Tip:** The default lock name setting is set using a default variable. Therefore, to make the lock names exactly the same for different objects and processes, you can enter a static string for the name or use variables that will resolve to the same in both processes. For example, if you want to lock a resource against concurrent use for two processes in different components that are mapped to the same resource, you could use `${p:resource.name}` as the lock name for both.

Set Status

The **Set Status** step enables you to explicitly set the status of a process. The following table describes the property for this step.

Property	Description
Status	Values: Success or Failed

Process Sleep

The Process Sleep step enables you to enter time in seconds for a process to wait until continuing with the next steps in the process. The following table describes the property for this step.

Property	Description
Sleep Time	Default: 30

Join

The Join step enables you to join steps in a process. There are no properties for this step.

For the Join step to continue processing on to its outboard connections, all its inbound connection paths must be valid. To be valid, all that should evaluate to success do so, all that should evaluate to failure do so, all that should evaluate to any value do so, and all that should evaluate to a specific switch value do so.

You can do an implicit join without the Join step by having multiple inbound connections for another step. Regardless of whether it is an implicit or explicit join, all incoming connections must be valid for the joining step to be executed.

One reason you should choose an explicit Join step is when your process has multiple steps that must all complete to a certain outcome and then should continue on to multiple outbound steps without additional processing between them.

Manual Application Task

The Manual Application Task step enables you to interrupt a process until some manual intervention is performed according to a pre-configured application task. Application tasks must be set for an application before you can select them for this step. See Application Tasks [page 289].

The following table describes the properties for this step.

Field	Description
Name	Enter a descriptive name for the step.
Task Definition	Used to select a previously-defined user-defined task. See Application Tasks [page 289].
Environment Role	Select the role expected to respond. The user mapped to this role must respond to the generated work item before the process can continue.
Application Role	Select the role expected to respond. The user mapped to this role must respond to the generated work item before the process can continue.

If both roles are selected, all affected users must respond before the process can continue.

Inventory Update

The Inventory Update step appears only if you have selected Advanced for the **Inventory Management** option for the application process. Use this step to implement your own inventory management process.

Adding Application Process Properties

Add properties to an application process for information that may have values specific to the scope of that process.

To define an application process property:

1. Navigate to **Management > Applications > [select application].**
2. Select the **Processes** tab.
3. In the processes list, next to the process for which you want to define a property, click the **Details** icon (👁).
4. On the **Properties** tab, click the **Add Property** button.
5. Complete the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

6. Click **Save**.

You can use this property in the same process where it is defined by referencing it in one of the steps. Use the following syntax: `${p:<property name>}`

Running Application Processes

Application processes deploy artifacts for a single application.

To deploy to a particular environment, you run the application process defined for that environment. For more about deployment, see Chapter 28: Deployment Overview [page 365].

To run an application process:

1. Navigate to **Management > Applications > [select application]**.
2. Select the **Environments** tab.
3. Click the name of the environment where you want to perform the deployment and select **Run Process**. The process wizard is displayed with the application and environment pre-selected.

   Note: If environments are in a pipeline and the pipeline is mandatory, the **Run Process** command appears only for the first environment.

4. On the **Process** page select the process you want to run and click **Next**.
5. On the **Options** page select **Only Changed Version** to ensure that only changed versions are deployed. If selected, no previously deployed versions will be deployed. For example, if you check the box and select a specific version that was already deployed, the version will *not* be redeployed. Do not select this if you want to deploy a version regardless of whether or not it was already deployed, such as when the inventory is out of date.
6. If you want to use a snapshot, select it from the **Snapshot** drop-down list-box. If you select a snapshot, the deployment will automatically use the component versions defined for the snapshot. For information about snapshots, see Snapshots [page 292].
7. If you did not select a snapshot, select a component version from the **Version** list-box. If more than one component is mapped to the application, each one is listed separately.

Version Option	Description
None	No version for this component. Useful when performing multi-component deployments or testing.
Specific Version	Enables you select any version already in CodeStation.

Version Option	Description
Latest Version	Automatically uses the most recently imported version.
Latest With Status	Automatically uses the most recently imported version with the specified status. Example status values are: Latest (default value), Passed Test, Archived.
All With Status	All component versions with the selected status will be deployed.
All in Environment	All component versions already deployed in the environment with the selected status will be deployed.
All in Environment (Reversed)	All component versions already deployed in the environment with the selected status will be deployed in reverse order. Example status values are: Active (default), Staged.

8. Optionally add a description.

9. Click **Submit** to start the process.

When a process starts, the **Request** page is displayed so that you can review the process request progress. After the request completes, you can review the results on that and other related pages. See Chapter 30: Analyzing Deployments [page 371].

People notified for any approvals must approve or reject. See Approving Requests [page 291].

Note: If gate conditions are defined for a target environment, the component versions or snapshots are deployed into that environment only if their status meets those conditions. See Gates [page 295].

Application Tasks

An application task is used to interrupt an application process until some manual intervention is performed. A task-interrupted process is suspended until the assigned user or users respond. Typically, application manual tasks are removed after the process has been tested or automated.

Like approvals, triggered application tasks notify assigned users. Notifications are associated with environment- or application-defined user roles. Unlike approvals, application tasks can be incorporated within an application process.

The person notified for the task can indicate that it is complete by approving the step from the request's log in **Approval Requests**. See Approving Requests [page 291].

Create and use tasks as follows:

1. Create application tasks in applications.

2. Insert a manual task step in an application process.

See Manual Application Task [page 286].

Creating Application Tasks

To create an application task:

1. Navigate to **Management > Applications > [select application] > Tasks**.
2. Click **Create**.
3. Name the task then select a template from the **Template Name** field.

   The individual tasks map to the notification scheme used by the application. See Creating Notification Templates [page 147]. If a scheme is not specified, the default scheme is used. The available task notification template is:

   - TaskCreated

Approvals

Approval processes tell Deployment Automation what to do when an application environment requires approval. An approval process must be created if the **Requires Approval** check box is selected when adding or editing an application environment.

When a process requires approval to run, approvers are responsible for verifying that the deployment is ready to run and deciding when they will release it to run by approving it. If the system is configured with an email server and the user has an email address set, notifications are sent to users who are eligible to complete an approval step.

If a scheduled deployment requiring approval reaches its start time without approval given, the process will not run and will instead fail with a message that it could not start. To resubmit a request, you must request a new process. If an approval-requesting process does not have a scheduled deployment time, the process will remain idle until a response has been made.

For more information on approval processes, see the following topics:

- Designing Approval Processes [page 290]
- Approving Requests [page 291]

Designing Approval Processes

If an application environment requires approvals, you must design an approval process for it.

You design processes using the process editor. In the process editor you drag process steps from a menu of steps onto the design space and configure them immediately. As you place additional steps, you visually define their relationships with one another.

To create an approval process:

1. To activate the options to create an approval process for an application environment, you must first select the **Require Approvals** option for the environment. See Adding Environments to Applications [page 270].
2. Navigate to **Management > Applications > [select application]**.

3. Select the **Environments** tab.
4. Select the environment name for which you want to create, design, or edit an approval process and select **View Details**. The **Approval Process** button appears at the upper right of the page and the environment information indicates that approvals are required. If these do not appear on the page, see step 1.
5. Click the **Approval Process**.
   The process designer for the approval process appears. Available approval steps are based on the security roles for the corresponding entities:
   - Applications
   - Components
   - Environments
6. Expand the approval levels and drag a security role to the design area to create an approval step. Do this for each role that you want to approve process requests for this application, its components, and this environment. You can define steps for each level and role as needed. Any users assigned to the role on the corresponding entities, either directly or as part of a group, have permission to approve.

   Note: In versions prior to 6.1.5, DA only runs approvals for components with versions. To maintain this behavior in version 6.2.2, in the Properties dialog box select this option: **Only for Components with Versions**

Approving Requests

When approval is pending, in the dashboard the **Pending Approval Requests** button shows the number of pending approvals. The request's **Log** tab indicates that the process is waiting to start.

Viewing and Responding to Requests

Approvers can view and respond to requests in the dashboard **Approval Requests** overlay or in the **Approvals** tab of the process request.

In the **Approval Requests** overlay, approvers can:

- Expand approval requests to see the processes and tasks included
- View the name of the process, when the request was submitted, who requested the process, and the snapshot or version used.
- View environment or resource details by clicking the link in the **Environment/ Resource** column.
- View request details by clicking the **View** button.
- In the expanded view, view the entity to be approved by clicking the link in the **Name** column.

Respond to the request by clicking the **Approve** or **Reject** button. Optionally enter comments.

If you reject a request, the process will not run. If you approve, the process will start.

Snapshots

A snapshot represents a specific state of an application in an environment. Snapshots specify the combination of component versions that you are going to deploy together. When a snapshot is created, Deployment Automation gathers together information about the application for a given environment, including the component versions, process versions, and properties.

Typically, the snapshot is generated in an uncontrolled environment, where there are no approval processes or gates defined. Once a snapshot is created, it can then be promoted through a deployment pipeline. For example, as the application moves through various testing environments, Deployment Automation ensures that the exact versions, bit for bit, are used in every environment. Once all the appropriate stages and approvals for a snapshot are complete, the snapshot is deployed to production.

Snapshots help manage complex deployments, such as deployments with multiple tiers and development teams. For example, after testing and confirming that team A's component works with teams B's, a snapshot can be taken. Then, as development progresses, additional snapshots can be taken and used to model the effort and drive the entire deployment, coordinating versions, configurations, and processes.

Snapshots include the following:

- **Component Versions**

  When you create a snapshot, you can manually specify the version for each component, or you can copy the versions used in a specified environment or snapshot.

- **Configuration**

  A snapshot contains the versions of the processes that have been designed, the properties that these processes use, and other information. From the configuration, you can view and set the versions of the processes and properties used in the snapshot.

- **Status**

  You can apply a status to a snapshot, which typically indicates whether the snapshot has been deployed successfully to an environment. You can use this status along with gates, which enable you to allow deployments to an environment only if the snapshot has reached a particular status. For example, you could set the INT environment gate condition so that if the status is set to Dev_Complete, the snapshot can be deployed to the INT environment.

- **Locks**

  You can lock snapshots when they are deployed to a particular environment, ensuring that nothing can be changed. Typically, you lock a snapshot when it goes into a pre-production environment before being deployed to production.

Creating Snapshots

To create a snapshot:

1. Navigate to **Management > Applications > [select application] > Snapshots**.

2. Click **Create**.

3. Enter a name and description.
4. Click **Save**.

The current versions of all elements are saved in the snapshot. After you have created a snapshot, select the snapshot to view and configure it.

Viewing Snapshot Environments

You can view and work with environments using the selected snapshot's **Environments** tab

1. View a list of environments mapped to this snapshot.
2. Run the snapshot processes in the selected environment. For environments in pipelines, this is available only on the first environment.
3. Select the **Request History** tab to view a list of processes run for this snapshot.
4. Select the **Changes** tab to view a history of changes made to component versions and resources in this snapshot.
5. Click a process name link to go to the process page.
6. Click the **Details** button to go to the process request page.

Configuring Snapshots

After you create a snapshot, you can configure the property versions.

To configure a snapshot:

1. Navigate to **Management > Applications > [select application].**
2. Select the **Snapshots** tab and then a snapshot.
3. Click the **Configuration** tab.
4. Expand the items in the tree to see the property version numbers that are currently selected in this snapshot. Included is property version information for the following:
   - Components
   - Component Versions
   - Component Processes
   - Applications
   - Application Processes
   - Application Environments
   - Resources
   - System Properties

5. To update all of the property versions, select an option:
    - **Reset All to Latest:** Reset all configuration targets to the latest version.
    - **Lock Loose Entries:** Lock all configuration targets that are automatically using the latest version available. Select this only if you are sure you want to lock these targets.
    - **Copy from Snapshot:** This uses the component configuration from the selected snapshot. **Warning:** If you have already added a component configuration to the snapshot to which you are copying, this action will remove it and replace it with the one from the selected snapshot.
6. To view the property details, beside the property click the **Details** icon.
7. To set the version to a specific version or set it to always use the latest version, beside the property click the **Set Version** button.

Adding Component Versions to Snapshots

You can add component versions to snapshots directly or copy from environments or other snapshots.

To add component versions to a snapshot:

1. Navigate to **Management > Applications > [select application]**.
2. Select the **Snapshots** tab and then a snapshot.
3. Click the **Component Versions** tab.
4. To add component versions, do one of the following:
    - To add component versions directly in the list of component versions, click the + (**Add**) button next to the component to which you want to add the version. You can then select any version in CodeStation for the component.
    - To copy component versions, click **Copy from Environment** or **Copy from Snapshot**.
5. Expand the components in the list to see the component versions now included in this snapshot.

Adding Statuses to Snapshots

Snapshot statuses are user-managed values that can be added to snapshots. Once a status is added to a snapshot, the value can be used in component processes or application environment gates. See Gates [page 295].

Before you can add a status, statuses must be defined. See Defining Statuses [page 110].

To add statuses to snapshots:

1. Navigate to **Management > Applications > [select application] > Snapshots > [select snapshot] > Statuses**.
2. Click the **Add Status** button.

3. Select the statuses you want to add.
4. Click **Add**.

You can also set snapshot status as part of application processes using the Snapshot steps, *Add Status to Snapshot* and *Remove Status from Snapshot*.

Viewing Snapshot Tasks

If you require approvals for an application to be run in an environment, a snapshot of the application will require the same. When the process is run for the snapshot, information about the approval request appears in the snapshot's **Tasks** tab.

To view snapshot tasks:

Navigate to **Management > snapshots > [select snapshot] > Tasks**.

Comparing Snapshots

You can compare snapshots to see how their component versions or configurations differ.

To compare snapshots:

1. Navigate to **Management > Applications > [select application] > Snapshots**.
2. Click the **Compare** button.
3. Select snapshots to compare.
4. Click **Compare**.
5. In the selection box, select **Versions** or **Configurations**.

Tip: If the **Configurations** comparison is selected, you can click **Only Show Differences** to narrow the view to just what has changed.

Gates

Gates ensure that component versions or snapshots are deployed into environments only if their status meets certain conditions. For example, you could specify a gate condition that allows deploying a component version to a Production environment only if the status indicates that the version was successfully deployed to and tested in the UAT environment.

Gates can be defined for each environment associated with the application, and each environment can have one set of gate conditions.

Before you can select conditions for gates, you must first do the following:

1. Define version and snapshot status values. See Defining Statuses [page 110].
2. Set the status on the element that you want to use to enforce a gate as follows:
   - Set the status of a component version. See Adding Statuses to Component Versions [page 234].

- Set the status of a snapshot. See Adding Statuses to Snapshots [page 294].

After you have deployed, you can see the statuses of the component versions and snapshots in the deployment results. See Chapter 30: Analyzing Deployments [page 371].

Creating Gates

You must add status values before creating gates. See Using Version Statuses [page 112] and Using Snapshot Statuses [page 112].

To create a gate:

1. Navigate to **Management > Applications > [select application] > Environments > [select environment] > View Details > Gates**.

2. Click the **Edit** button on the gates page. The **Edit Gates** dialog box appears.

3. Select a value.

   Note: To view and define the available statuses, see Defining Statuses [page 110].

   The selection list includes both version and snapshot statuses. Selecting a value provides both `And` and `Or` selection boxes.

   To use And:

   Click **Add Condition**. This adds an additional value to the condition that must be satisfied. For example, defining the gate `Passed Tests And Latest` means that only component versions with both statuses, `Passed Tests` and `Latest`, satisfy the condition and can be deployed into the environment. A single condition can have as many `And`-ed values as there are statuses defined.

   To use Or:

   Click **Add Condition Group**. Using the `Or` box adds a condition group to the gate. Additional conditions are defined in the same way as the first one. A gate with two or more conditions means the component will be deployed if it meets *any* of the conditions. For example, if you define the gate `Passed Tests Or Latest`, a component will pass the gate if it has either status (or both). A single gate can have any number of conditions.

   To remove a condition:

   Click the - beside the latest selection.

4. Click **Save**.

See Defining Statuses [page 110] for more information about version and snapshot statuses.

Chapter 20: Managing Environments

Environments represent logical deployment locations. Your deployment processes must run in at least one environment. Environments and their resources are used by applications and components at runtime.

An environment brings together components with the agent that deploys them. Deployment Automation maintains an inventory of every artifact deployed to each environment and tracks the differences between them.

Environments are created at the global level and used by multiple applications. This enables IT organizations to use either shared or application-specific environments. Environments are typically modeled on stages of the software project life cycle, such as development, QA, and production.

Environments can have different topologies. For example, an environment can consist of a single machine, be spread over several machines, or be spread over clusters of machines. Approvals are often required before processes can be run in an environment.

After you have created environments, you can create pipelines, which are sequences of environments where application process requests are propagated. You can associate a pre-configured pipeline of environments and individual environments to an application.

Before you can run a deployment, you must create at least one environment and associate the environment with an application. In the application, you configure the environment specifically for that application and its component versions. The application environment configuration associates components with an agent on the target host. For details on environment configuration at the application level, see Application Environments [page 269].

The following topics describe how to manage environments.

Environments Page Overview

You typically manage environments in the **Environments** page of the user interface. Following are some of the things you can do from this page.

- Filter by the environments created by you or by all users.
- Toggle the Inactive button to include or exclude inactive environments.
- Click the **Environment Actions** button to create or import environments.

- Click the name to open the page for a listed environment.
- Click an icon beside an item to do the following:
  - Edit
  - Inactivate/Activate
  - Export

Note: Environments cannot be deleted. This prevents any alteration of the deployment history for the environments.

Creating Environments

You can create environments or import existing ones. See Importing/Exporting Environments [page 301] for information about importing environments.

To create an environment:

1. Navigate to **Management > Environments**.
2. Click **Create**.
3. Enter the information needed to create the environment as follows:

Field	Description
Name	Enter the name according to the typical use of the environment. For example, if this represents your system integration testing environment, you could use the name *SIT*.
Description	Enter an optional description.
Color	Select a color to visually identify the environment in the user interface.
Auto-select for Export	If this is selected, this environment will be added to the selection list during a replication export. It can be manually removed from the selection list before export if needed.

4. To continue creating environments and stay in the list, click **Save**. To start configuring the environment you just created, click **Build Out**.

Configuring and Monitoring Environments

After you create environments, you associate them with agents or agent resources and map the environments to applications. You can also make them part of a pipeline.

You can monitor environment activity by selecting an environment. Following are some of the things you can do from an environment's page.

- See all applications associated with this environment.

- View the inventory for this environment.

  See Chapter 31: Inventory Management [page 377].

- View the timeline specific to this environment.

  See Viewing Timelines [page 371].

- View resources associated with this environment.

  See Mapping Agents and other Resources to Environments [page 299].

- View and set properties for this environment.

  See Adding Environment Properties [page 300].

- View and add role members and view default permissions for this environment.

  See Setting up Security [page 115].

- Schedule application processes and blackouts.

  See Scheduling Deployments [page 368] and Setting Deployment Blackout Dates [page 368] respectively.

- Select from the following actions:

  - Print Page

  - Help

- Edit the selected environment details, including name, description, and color.

Note: Some configuration of environments is application-specific, and therefore is done as part of the application configuration. This includes overrides to security. For more information, see Adding Environments to Applications [page 270].

Mapping Agents and other Resources to Environments

Each environment must have at least one agent associated with it. You must do this before you can map the environment resource to components.

To map an agent to an environment:

1. Navigate to **Management > Environments > [select an environment]**.

2. Select the **Resources** tab.

3. Click **Add a Resource**.

4. Select agents or other resources.

5. Click **Add**.

The added resources are now available to be mapped to components in the related application environment. See Mapping Application Components to Resources [page 273].

Adding Environment Properties

Add properties to an environment for information that may have values specific to that environment. If an environment is shared by multiple applications, and if all or most of the applications should use the same value for an environment property setting, you should set the value at the environment level. You can override the value for the same-named property at the application environment level as needed.

Note: If component environment properties have been added for a component mapped to the environment, those properties will appear in the environment's **Component Mapping** page with the defined fields available for you to enter values. See Adding Component Environment Properties [page 227].

To create environment properties:

1. Navigate to **Management > Environments > [select environment]**.
2. Select the **Properties** tab.
3. To set a single property, click **Add Property**.
    a. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Secure	Select this if the property should be secure. Secure properties are stored encrypted and displayed obscured in the Deployment Automation user interface.
Value	Value for the property that will be used by steps in a process

4. To set multiple properties, click **Batch Edit.**
    a. In the **Properties** box, enter multiple properties and their values on separate lines as property=value.

    For example, if you had properties `host` and `port`, you would enter them as follows:

```
host=myServer

port=8080
```

Note:

The following restrictions apply to batch edit:

- The **Secure** option cannot be set.
- The value cannot contain the "=" symbol.

5. Click **Save**.

To use an environment property in a process, reference it when you configure a step that uses it. Use the following syntax: `${p:environment/<property name>}`

Importing/Exporting Environments

Environments can be exported and then imported into another server. This can be useful if you have multiple Deployment Automation servers, such as test and production servers, where you create and validate the environments on the test server and then export them and import them into the production server.

- Exporting Environments [page 301]
- Importing Environments [page 301]

Related Topics

Chapter 27: Replication [page 351]

Exporting Environments

Exporting an environment creates a JSON file that contains the environment's configuration information. For information about JSON, see www.json.org.

To export an environment:

1. Navigate to **Management > Environments**.
2. In the column beside an environment, click the **Export** icon.

You can open the file in a text editor or save it. If you save it, a file is created with the same name as the selected environment, such as `UAT.json`.

Importing Environments

When you import an environment, you can create a new environment or upgrade an existing one.

To import an environment:

1. Navigate to **Management > Environments**.
2. Click **Import**.
3. If you want to upgrade an existing environment, select **Upgrade Environment**.

   If the environment's name in the JSON file (not the name of the file itself) matches an existing environment, the environment's properties are updated with the new values, and new items, such as resources, are added. If the name of the environment is not found, the command has no effect.

   Note: The environment's name is the first parameter in the JSON file. For example:

   ```
   "name": "UAT"
   ```
4. Click **Choose File** and select the JSON file containing the environment.
5. Click **Import**.

Chapter 21: Managing Pipelines

A pipeline is a pre-defined sequence of environments in which application process requests are executed.

For details on creating and configuring pipelines, see the following topics:

- Pipelines Page Overview [page 303]
- Creating Pipelines [page 303]
- Configuring Pipelines [page 304]
- Adding Pipelines to Applications [page 304]
- Importing/Exporting Pipelines [page 305]

Pipelines Page Overview

You typically manage pipelines in the **Pipelines** page of the user interface. Following are some of the things you can do from this page.

- Filter the pipelines created by you or by all users.
- Click the **Pipeline Actions** button to create or import pipelines.
- Click the name link to open the page for a specific pipeline.
- Click an icon beside an item to do the following:
  - Edit
  - Export

Creating Pipelines

You can create pipelines or import existing ones. See Importing/Exporting Pipelines [page 305] for information about importing pipelines.

To create a pipeline:

1. Navigate to **Management > Pipelines**. Click **Create**.
   - **Name and Description**

     Enter a name and description that indicate the purpose of the pipeline.
   - **Mandatory**

     If selected, you can request a process only on the first environment in the pipeline. Thereafter, you can only promote on subsequent environments. This ensures that the same configuration is used for each environment in the pipeline.

     If not selected, you can request processes on all of the environments in the pipeline.

The default is deselected.

Note: If you promote multiple requests for a particular process and they queue up waiting to be executed, only the last request for that process is executed in the target environment and the earlier ones are ignored.

- **Auto promote**

  If selected, after the process is requested in the first environment, upon successful deployment into that environment, the application is automatically promoted to the next environment in the pipeline sequence. If each promotion is successful, the auto-promotion continues from environment to environment until promotion into the last environment in the pipeline. The **Promote** button will not be available for environments belonging to an auto-promoted pipeline.

  Pipeline execution request operations are not queued if there are several requests coming to the same environment. Requests for each process are executed and may be done in parallel on the same environment.

  Note: If a situation occurs where multiple requests for the same process are done before execution in an environment, the last request for that process is executed and the earlier ones are ignored.

  The default is deselected.

2. To continue creating pipelines and stay in the list, click **Save**. To start configuring the pipeline you just created, click **Build Out**.

Configuring Pipelines

After you create a pipeline, you must configure it to define the sequence of environments.

To configure a pipeline:

1. Navigate to **Management > Pipelines > [select pipeline]**.
2. In the **Pipeline** tab, click **Add/Remove Environments**.
3. Drag environments to the **Pipeline** section.
4. Reorder by dragging the environments to different positions.
5. Remove an environment by dragging it out of the pipeline.
6. Click **Save**.

Adding Pipelines to Applications

After you have configured a pipeline, you can add it to an application. An application can have only one pipeline.

When you associate a pipeline with an application, all environments associated with the pipeline are automatically added to the application. When you add a pipeline to an application that already has environments associated with it, the order of environments in the pipeline will override the order of any duplicate environments already mapped to the application.

Note: You can add a pipeline to an application as you create the application, or you can add or change a pipeline selection in an existing application.

To add a pipeline to an existing application:

1. Navigate to **Management > Applications**.
2. Beside the application to which you want to add a pipeline, click the **Edit** icon.
3. From the **Pipeline** list, select the pipeline you want to add.
4. Click **Save**.

Importing/Exporting Pipelines

Pipelines can be exported and then imported into another server. This can be useful if you have multiple Deployment Automation servers, such as test and production servers, where you create and validate the pipelines on the test server and then export them and import them into the production server.

- Exporting Pipelines [page 305]
- Importing Pipelines [page 305]

Related Topics

Chapter 27: Replication [page 351]

Exporting Pipelines

Exporting a pipeline creates a JSON file (file extension `json`) that contains the pipeline's configuration information. For information about JSON, see www.json.org.

To export a pipeline:

1. Navigate to **Management > Pipelines**.
2. In the column beside a pipeline, click the **Export** icon.

You can open the file in a text editor or save it. If you save it, a JSON file is created with the same name as the selected pipeline.

Importing Pipelines

When you import a pipeline, you can create a new pipeline or upgrade an existing one.

To import a pipeline:

1. Navigate to **Management > Pipelines > Import**.
2. Click **Choose File** and select the JSON file containing the pipeline.

3. If you want to upgrade an existing pipeline, select **Upgrade Pipeline**.

   If the pipeline's name in the JSON file (not the name of the file itself) matches an existing pipeline, the pipeline's properties are updated with the new values, and new items, such as resources, are added. If the name of the pipeline is not found, the command has no effect.

   Note: The pipeline's name is the first parameter in the JSON file.

4. Click **Import**.

Chapter 22: Managing Resources

The term resources in Deployment Automation refers to all objects managed through the resources page. This includes:

- Deployment targets, such as the agent itself, or entities managed by the agent, such as databases or J2EE containers. Resources managed by agents are usually found on the same target server where the agent that manages them is located. The target server can be a physical machine, virtual machine, or a cloud-based location.
- Agent relays and network relays, which define the connection details and manage the communication between the server and agent.

For details on the various entities managed through the Resources page, see the following topics:

Resources Page Overview

You typically manage resources in the **Resources** page of the user interface. Following are some of the things you can do from this page.

- Filter the resources list by the type of resource by selecting from the drop-down menu.
- Filter by the resources created by the Status: Online, Offline, or Unknown.
- Toggle the Inactive button to include or exclude inactive resources.
- Click the **Add Resources** button to create resources.

- Click the name to open the page for a listed resource.
- See the status of the resource.
- Click an icon beside a resource to do the following:
  - Edit
  - Copy
  - Inactivate/Activate
  - Delete

Resources

Resources represent a deployment target on a Deployment Automation environment. Examples include physical machines, virtual machines, databases, or J2EE containers.

The agents themselves appear in the **Resources** tab list, as do resources logically defined to subdivide or group the agents. Resources generally reside on the same host where its managing agent runs. If an agent is configured to handle multiple resources, a separate agent process is invoked for each one. Components are deployed to resources by agents, which are physical processes.

A resource can represent a physical machine, which is the simplest configuration, or a specific target on a machine, such as a database or server. A host machine can have several resources represented on it. In addition, a resource can represent a process distributed over several physical or virtual machines.

To perform a deployment, at least one resource must be defined and for non-trivial deployments, at least one agent must be defined. Typically, each host in a participating environment has an agent running on it to handle the resources located there.

A proxy resource is a resource effected by an agent on a host other than the one where the resource is located. If an agent does not require direct interaction with the file system or with process management on the host, a proxy resource can be used. When a deployment needs to interact with a service exposed on the network, such as a database or J2EE server, the interaction can happen from any machine that has access to the networked service.

Creating Resources

You can create resources to make logical mappings for agents, agent resources, and agent pools.

To create an agent resource:

1. Navigate to **Management > Resources**.
2. Click **Create**.
3. Enter the information needed to create the resource as follows:

Field	Description
Name	Property name
Description	Property description
Child of	Select the parent of the resource from the following options: • Agent • Resource • Agent Pool
Agent / Resource Parent / Agent Pool*	Select the parent of the resource.
Default Impersonation	If you want to use user impersonation, fill out the resulting fields for the default login for the given resource. See User Impersonation [page 251].
Always Use Default Impersonation	Select this if this login should be used each time this resource is accessed.

* The field name changes depending on the option selected in the **Child of** field.

4. Click **Save**.

Resource Configuration

After you create a resource, select it in the **Resources** page to configure it. Following are some of the things you can do from this page.

- View the resource details.
- Edit the resource details. From here you can change the parent of sub-resources if needed.
- **Inventory:** Select from the selection box for the following pages:
  - **Current:** View the current inventory for this resource
  - **Configuration:** View the properties set for this resource and all its associations
  - **Sub-Resources:** View and create sub-resources for this resource. This lists resources that are children of the selected resource. From here you can create additional resources. See Creating Resources [page 308].
- **Roles & Properties:** Select from the selection box for the following pages:
  - **Roles:** Add resource roles that you've previously created to this resource.

    - **Properties:** Add resource properties.

- **History:** View resource deployment and change history.
- **Security:** Set role membership and default permissions for this resource.
- Select the number of resources to view per page.

Agents

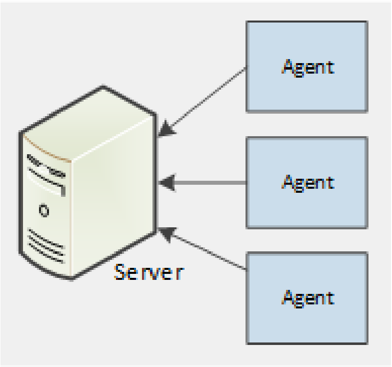
Agents are physical resources for deployment. To run a deployment, an agent must be installed on the target server.

Typically, an agent is installed in every environment that an application passes through.

An agent is a lightweight process that runs on a target server and communicates with the Deployment Automation server. Agents are integral to the client/server architecture. Agents perform the actual work of deploying components and so relieves the server from the task, making large deployments involving thousands of targets possible.

Process requests, such as deployments, are orchestrated by the server and performed by agents distributed throughout the network. Server-agent communication is built around transferring, or deploying, components.

Simple Server-Agent Communication



Managing Agents and Agent Pools

After an agent is installed and communication is established between the server and agent, the agent will be visible in the Deployment Automation user interface where it can be configured. Active agents, regardless of operating system, can be upgraded using the user interface.

Agent configuration consists of assigning an agent to at least one environment; agents can be assigned to multiple environments. If an agent is assigned to several environments, it can perform work on behalf of all of them.

Agents can be grouped into sets of agents called agent pools.

You typically manage agents and agent pools in the **Resources** page of the user interface in the **Agents / Pools** pane. Following are some of the things you can do from this page.

- Click an icon beside an agent or agent pool to do the following:
    - Edit

    - Restart
    - Upgrade
    - Test Connectivity
    - Inactivate
    - Remove

- Click the name link to open the page for a specific agent or agent pool.
- Click the agent relay name link to open the page for the agent relay to which an agent is connected.

For information on managing agents and agent pools through the web application, see the following topics:

- Assigning Notification Schemes to Agents [page 311]
- Installing Agents Remotely [page 311]
- Managing Agents Remotely [page 313]
- Upgrading Agents [page 314]
- Dynamically Registering Agents [page 314]
- Agent Pools [page 315]

Assigning Notification Schemes to Agents

If you want to send email notifications to appropriate personnel when agents go offline, you can configure and assign notification schemes to specific agents.

For details on configuring notification schemes See Configuring Email Notifications [page 144] .

To assign a notification scheme to an agent:

1. Navigate to **Management > Resources**.
2. Select the **Agents** tab.
3. Click the edit icon beside the agent for which you want to send notifications.
4. In the **Notification Scheme** field, select the scheme you configured for this purpose.

Installing Agents Remotely

You can install an agent onto a UNIX machine using the web application. A remotely-installed agent cannot be installed as a service.

To install an agent:

1. Navigate to **Management > Resources**.
2. Select the **Agents** tab.

3. Click **Install Agent**.

4. Enter the information needed to install the agent as follows:

Field	Description
Target Hosts*	Host names or IP addresses of the machines where the agent will be installed.
SSH Port*	Secure Shell (SSH) port addresses of the machines where the agent will be installed.
SSH Username*	SSH user name used on the machines where the agent will be installed.
Use Public Key Authentication	Check this box if you want to authenticate using public key authentication instead of a password.
SSH Password*	SSH password associated with the user name used on the machines where the agent will be installed.
Agent Name*	Name of the agent.
Agent Dir*	Directory where agent should be installed.
Java Home Path*	Path to Java on the machine where the agent will be installed.
Temp Dir Path*	Path to the directory used to perform the installation on the target machine.
Server Host*	Host name or IP address of the Deployment Automation server or agent relay to which the agent will connect.
Server Port*	Deployment Automation server port (7918) or agent relay (7916) to which the agent will connect.
Mutual Auth	Check this box if the agent should enforce certificate validation for mutual authentication.
Proxy Host	Host name or IP address of the agent relay if used.
Proxy Port	HTTP port of the agent relay (20080) if used.

5. Click **Save**.

Remotely installed agents will start running automatically. If a remotely installed agent stops running, it must be restarted on the host machine.

Managing Agents Remotely

Once an agent is installed, you can manage many of its features from the Deployment Automation user interface. Agent properties can be also changed directly on the agent machine by editing the agent's `conf/agent/installed.properties` file and restarting the agent.

To manage an agent:

1. Navigate to **Management > Resources**.
2. Select the **Agents** tab.

   In the user interface, you can see the following information about each agent:

   - Name and description
   - Licenses assigned
   - Status, such as Offline, Connected, and Online
   - Version
   - Agent relay it is connected through, if any
3. Click an action link for the desired agent.

Action	Description
Edit	This option enables you to edit the agent's description and impersonation settings. You can also specify an internal agent name that is resolved dynamically to a regular agent name. The server updates the agent name with the internal agent name that you specified. For more information see Dynamically Registering Agents [page 314].
Restart	This option will stop and restart the agent. While the agent is stopped, its status will be `Offline`.
Upgrade	This option will stop the agent and apply the upgrade. While the agent is stopped, its status will be `Offline`. After the upgrade is applied, the agent will be restarted. Before its status is `Online`, it might briefly be `Connected`.
Test	This option will perform an agent settings and connection test. Test results are displayed in the **Connection Test** dialog.
Inactivate	This option will deactivate the agent. Agents that are deactivated cannot perform deployments. To reactivate the agent, check the `Show Inactive Agents` check box on the Agents pane, then click `Activate` for the agent.
Delete	Removes the agent.

Upgrading Agents

After upgrading to a new version of the server or receiving notice of available agent upgrades, you can upgrade your agents from the Deployment Automation user interface. The agents must be online to be upgraded.

To upgrade your agents:

1. Navigate to **Management > Resources**.
2. Select the **Agents** tab.
3. If any agents are noted as eligible for upgrade in the **Upgrade Status** column, review any important messages indicated by the messages icon (⚠) before proceeding. Upgrade status will appear only if data collection is enabled for your implementation. See Chapter 15: Data Collection and Check for Update [page 183].
4. To upgrade your agents:

   a. Select one or more agents in the list using the check boxes at the left of the agent names and then click the **Upgrade Selected Agents** button in the upper right. If you want to upgrade a single agent, you can click the **Upgrade** button beside the agent that you want to upgrade.

   Note: If the agent is not eligible for upgrade through the user interface, neither the check box nor the **Upgrade** button will appear.

   b. To confirm that you want to upgrade, click **Yes**.

All existing agents using Java 6 and greater will be upgraded by the server.

Note:

- Deployment Automation version 4.5 and later agents use Java 6 and later. Any agents using Java 6 or above can be upgraded and do not have to be uninstalled.
- Any 4.0.1 and prior agent using Java 5 *cannot* be upgraded. Most of the new functionality will *not* work with these agents. You must uninstall the agents using Java 5 before installing the Java 6 and later agents.

Dynamically Registering Agents

You can dynamically register agents by specifying internal agent names, which are resolved automatically on remote machines. Property values are taken from these locations:

- *Agent*: system and Java properties that are collected from an agent when it connects to a server.
- *System*: located in Administration -> System -> Properties.

You can use short and full notations when specifying an internal agent name:

- Short: `${propertyName}`
- Long: `${p:propertyName}`

When an agent connects or re-connects to a server, the server resolves the agent's name by reading its properties. When you modify an internal agent name, the server updates the name with the value that you specified. The server notifies you if the change succeeds or fails.

Conflicting Names

If you specify a conflicting name, the agent's regular name is modified and a suffix is added. This enables you to modify the internal agent name on the server at a later date. Conflicting names are enabled by default. This behavior is controlled by the setting "Allow conflicting name for agents" in System | System Settings.

Agent Pools

An agent pool is a group of agents. Agent pools help you organize and manage agents installed in different environments.

For information on agent pools, see the following topics:

- Creating Agent Pools [page 315]
- Adding Agents to Agent Pools [page 315]
- Updating Agent Pools [page 316]

Creating Agent Pools

To create an agent pool:

1. Navigate to **Management > Resources**.
2. Select the **Pools** tab.
3. Click **Create**.
4. Enter the pool name in the **Name** field.
5. Optionally enter a description in the **Description** field.
6. Click **Save**.

Adding Agents to Agent Pools

To create an agent pool:

1. Navigate to **Management > Resources**.
2. Select the **Pool** tab.
3. In the list, select the agent pool to which you want to add agents.
4. Click the **Add Agents** button.
5. Select the agent or agents to add to the pool. Optionally, you can filter the listed agents by entering search text into the text field.
6. Click **Add**.

Updating Agent Pools

To update agent pools:

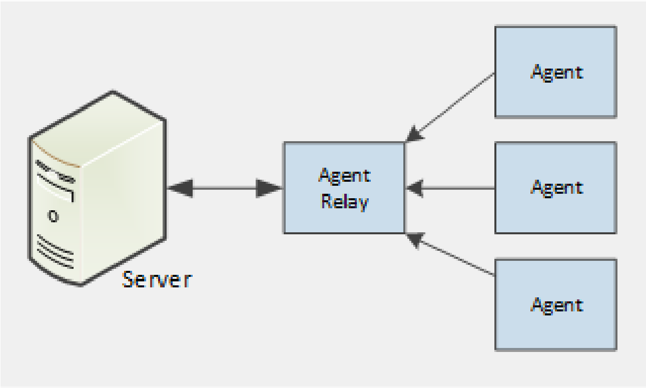
1. Navigate to **Management > Resources**.
2. Select the **Pools** tab.
3. Click an action icon for the desired agent pool. Actions are described in the following table.

Action	Description
Edit	Add and remove agents and edit the agent pool's name and description.
Duplicate	Copy and create a new agent pool with the same agents as the selected agent pool.
Inactivate	Deactivate the agent pool.
Remove	Remove the agent pool.

Agent Relays

Agent relays are used to manage communication between servers and agents. Agent relays are typically used when agents are dispersed across geographic locations or must communicate through firewalls. Agent relays can also be used to manage network traffic in implementations where you have many agents.

Simple Agent Relay Communication



Managing Agent Relays

You can do the following through the user interface:

- Remotely restart or delete agent relays
- Change name, description, and set whether HTTP communication is server or agent initiated

- Detect agent relays that were upgraded from agent relays or network relays in previous versions of Deployment Automation

For information on managing agent relays through the user interface, see the following topics:

- Configuring Agent Relays [page 317]
- Detecting Agent Relays [page 318]

Configuring Agent Relays

After you have installed an agent relay and detected it as needed, you can change the configuration as needed for your implementation.

The full set of options are set for agent relays at the time of installation. You can modify some, but not all, of the options of the agent relays remotely from the user interface. These can be changed only if the agent relay is online.

To change agent relay settings that cannot be changed through the user interface, edit the `agentrelay.properties` file located in the `relay_installation\conf` directory. See Agent Relay Installation Options [page 62].

To edit an agent relay:

1. Ensure that the agent relay is installed and is online.
2. Select the **Agent Relays** tab.
3. Select the agent relay.
4. View and change the agent relay information as follows:

Option	Description
Name	Name of the agent relay. This field can be edited only if the agent relay is online.
Host	Remote hostname or IP address the server will use to connect to the agent relay. This field can be edited only for agent relays that are online and have **Server Initiated** selected.
JMS Port	View only. Remote port used to communicate over JMS protocol.
Server Initiated	Select this if you want to ensure that the server initiates HTTP communication for requests over a given size. See Server-initiated HTTP Communication Using Agent Relays [page 74].

CAUTION:

Agent relays for version 5.1.5 that are configured with Server Initiated selected are compatible only with Deployment Automation servers version 5.1.5 and later. If you attempt to connect a version 5.1.5 agent relay that has Server Initiated selected to a server earlier than version 5.1.5, the agents connected through the agent relay will not be in Online status.

5. Click **Save**.

Detecting Agent Relays

If you upgrade a server configuration where the agent relays communicate with the server through network relays, the server cannot automatically detect the agent relay, and you will need to tell the server where to find it.

After upgrading an agent relay:

- If the agent relay name appears as an automatically-generated name with an ID string appended to it and no status information appears, you must use the Detect Agent Relay option for that agent relay.
- If the agent relay name appears as the correct name of the agent relay and you can see the status information, you *must not* use the Detect Agent Relay option.

Note: If you take the server offline and upgrade all agent relays before bringing the server back online, you may not need to use the **Detect Agent Relay** option. This should be used only if necessary.

To detect an agent relay:

1. Upgrade your agent relays. See Upgrading Agent Relays [page 72].
2. Navigate to **Management > Resources > Relays**.
3. If the agent relay name does not appear as the correct name and the **Status** field is blank, click **Detect Agent Relay**.
4. Enter the requested information as follows:

Option	Description
Host	Remote hostname or IP address the server will use to connect to the relay
JMS Port	Remote port used to communicate over JMS protocol
Timeout	How long the search should continue before timing out, in seconds

5. Click **Save**. In the **Relays** list, the agent relay should now appear with its given name.

6. If the agent relay shows a status of **Offline**, in the **Actions** column, click **Activate** to activate it on this server.

The correct name, status, and other important information should now appear for the detected agent relay.

Managing Network Relays

Network relays can be used in multiple-server configurations to manage the communication between servers, and are used in high availability, active-active server implementations. For configuring server to agent communication for either agent- or server-initiated communication of HTTP communication, you should use agent relays.

To create and configure a network relay, see Configuring Agents for High Availability (Alternate) [page 94].

Click the **Filter Network Relays** filter to see the network relays in the **Relays** page list.

Note: The behavior of agent and network relays changed in version 5.1.5 of Deployment Automation. Ensure you are using the correct version of the documentation for your version of Deployment Automation. If you are using a Deployment Automation server of version 5.1.5 or later, ensure all agent relays are upgraded to version 5.1.5 or later.

Resource Groups

A resource group is a logical collection of resources. Resource groups help you organize and manage the agents installed in different environments throughout the network. For example, resource groups can help you manage multi-tenant scenarios in which several machines share the same resources.

The groups you create and how you organize the groups, such as using subgroups, depends on your existing organizational processes and infrastructure.

Note: Before continuing, ensure that at least one agent has been installed in a target environment.

See the following.

- Creating Resource Groups [page 319]

Creating Resource Groups

To create a resource group:

1. Navigate to **Management > Resources > Groups**.

2. Expand **All Resource Groups** in the left pane and select the resource where you want to create a group.

3. Click **Create Resource Group**.

4. Enter the information needed to create the resource group as follows:

Field	Description
Name	Typically, the name will correspond to either the environment the resource participates in, the component that uses the resource group, or a combination of both, for example SIT, DB, or SIT-DB.
Description	Give a description that tells how you intend to use the resource to which this group is assigned.
Type	Select a type. Options are: • **Static:** Typical • **Dynamic:** If you select this, you can add conditions to direct the system to find and select a resource name to add to the group. This lets you use name patterns so that resources of certain naming conventions are dynamically added to the group.

5. To continue creating resource groups and stay in the list, click **Save**. To start configuring the resource group you just created, click **Build Out**. Once you assign a group to a resource, you can add sub-resources. A *sub-resource* enables you to apply logical identifiers, or categories, within any given group. During deployment configuration, you can select a given sub-resource that the process will run on.

6. To create a sub-resource, select a group and click **Create Resource Group**.

Adding Resource Properties

Add properties to a resource for information that may have values specific to that resource. If a resource is shared by multiple components, and if all or most of the components should use the same value for a resource property setting, you should set the value at the resource level. For more information, see How Properties Are Resolved [page 345].

To create resource properties:

1. Navigate to **Management > Resources > [select resource]**.

2. Select the **Properties** tab.

3. To set a single property, click **Add Property**.

   a. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description

Field	Description
Secure	Select this if the property should be secure. Secure properties are stored encrypted and displayed obscured in the Deployment Automation user interface.
Value	Value for the property that will be used by steps in a process

4. To set multiple properties, click **Batch Edit**.

   a. In the **Properties** box, enter multiple properties and their values on separate lines as property=value.

   For example, if you had properties `host` and `port`, you would enter them as follows:

   ```
   host=myServer
   ```

   ```
   port=8080
   ```

   Note:

   The following restrictions apply to batch edit:

   - The **Secure** option cannot be set.
   - The value cannot contain the "=" symbol.

5. Click **Save**.

To use a property in a process, reference it when you configure a step that uses it. Use the following syntax: `${p:resource/<property name>}`

Resource Roles

A role enables you to further refine how a resource is used, and is similar to sub resources. For most deployments, you will not need to define a role.

During process configuration, you select a specific role when determining the resource. A role can be used to set up Deployment Automation for rolling deployments, balancing, and so on.

For example, you can set up your process as follows:

1. First do a partial deployment to only a small percentage of target machines
2. Next require a manual task for the first role to execute after they have tested the partial deployment
3. Finally, once the manual task has completed, assign the rest of the process to a second role who is responsible for deploying to the rest of the target machines

See the following:

- Creating Resource Roles [page 322]
- Adding Resource Roles to Resources [page 322]

- Adding Resource Role Properties [page 322]

Creating Resource Roles

You can create resource roles to limit activities during deployment to resources that play a certain role in your network configuration or application deployment strategy.

To create resource roles:

1. Navigate to **Management > Resources > Roles**.
2. Click **Create**.
3. Fill in the fields as shown in the following table.

Field	Description
Name	Resource role name
Description	Resource role description
Color	Select a color to visually identify the resource role in the user interface.

4. Click **Save**.

You can now add the resource role to a resource.

Adding Resource Roles to Resources

After you create resource roles, you can add them to resources. This enables you to limit activities during deployment to resources that play a certain role in your network configuration or application deployment strategy.

To add resource roles to resources:

1. Navigate to **Management > Resources > [select resource]**.
2. Select the **Roles** tab.
3. Click **Add Resource Roles**.
4. Select roles resource roles and click **Add**.

Adding Resource Role Properties

Add properties to a resource role for information that may have values specific to that resource role.

To define an application process property:

1. Navigate to **Management > Resources > Roles**.
2. Select a resource role for which you want to define a property.

3. Select the **Properties** tab.
4. Click **Add Property**.
5. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

6. Click **Save**.

You can now use the resource role property in a process. Use the following syntax: `${p:resource/<resource role>/<property name>}`

For example, if you create a `WS` resource role and define a property for it named `serverURL`, you can access the property value from a process step as follows: `${p:resource/WS/serverURL}`

Chapter 23: Managing Deployment Packages

Deployment packages enable you to deploy artifacts for multiple applications. They may also include component processes where components are shared among multiple applications and associated versions are to be deployed as part of the larger package.

For details on managing deployment packages, see the following topics:

- Creating Deployment Packages [page 325]
- Deployment Package Configuration [page 326]
- Deployment Package Processes [page 327]
- Deployment Package Tasks [page 333]

Related Topics

- Chapter 19: Managing Applications [page 263]
- Part 5: Deployment [page 363]

Creating Deployment Packages

Create deployment packages if you want to deploy multiple applications from a single process.

To create a deployment package:

1. Navigate to **Management > Deployment Packages**.
2. Click **Create**.
3. Enter the information needed to create the deployment package as follows:

Field	Description
Name and Description	Enter a name and description.
Notification Scheme	Select a notification scheme if you want to send email notifications based on triggering events, such as when an application deployment fails or succeeds or when a request needs approval. See Configuring Email Notifications [page 144].

Field	Description
Pipeline	Select the pipeline you want to use for this deployment package's requests. See Chapter 21: Managing Pipelines [page 303]. **Note:** Deployment packages cannot be deployed through pipelines that differ from the pipelines used in the application processes they initiate.
Execution Timeout	(Recommended) Enter how long the process execution should continue before timing out, in minutes. If this value is set to 0, the process may run indefinitely.

4. To continue creating deployment packages and stay in the list, click **Save**. To start configuring the deployment package you just created, click **Build Out**.

Deployment Package Configuration

After you create a deployment package, select it in the **Deployment Packages** page to view and configure it. Following are some of the things you can view and do from this page.

Details: Click the **Details** button at the top right of the page to view deployment package information.

- **Activity:** View a graphical roll-up of activity for the selected deployment package, including deployment summary and scheduled deployments.
- **Details:** View and edit the deployment package details, including designating the pipeline. From this page you can also add deployment package properties.
- **Inventory:** View deployments to all application environments updated by the deployment package.
- **Tasks:** Configure manual tasks that you will later incorporate into a deployment package process.
- **History:** View deployment package deployment and change history.
- **Security:** Set role membership and default permissions for this deployment package.

Design: Click the **Design** button at the top right of the page to invoke the process design editor for deployment packages.

See Designing Deployment Package Processes [page 327].

Run: Click the **Run** button at the top right of the page to invoke the process wizard.

See Running Deployment Package Processes [page 331].

Deployment Package Processes

Deployment package processes deploy artifacts for multiple applications and components that span applications.

Deployment package processes can run manually, automatically on some trigger condition, or on a user-defined schedule. When a deployment has several application processes or a combination of application and component processes defined for it, the deployment package determines which ones are executed and in which order. For example, deployment package might deploy all applications for a quarterly release. A deployment package can orchestrate the entire process, including all included processes plus additional utility steps included at the deployment package level.

For information on application processes, see Application Processes [page 274].

For information on creating and designing deployment package processes, see the following topics:

- Designing Deployment Package Processes [page 327]
- Deployment Package Process Utility Steps [page 328]
- Adding Deployment Package Properties [page 330]
- Running Deployment Package Processes [page 331]
- Deployment Package Inventory [page 332]

Designing Deployment Package Processes

You design deployment package processes to tell Deployment Automation what to do when you deploy multiple applications or components that span applications.

You design processes using the process editor. In the process editor you drag process steps from a menu of steps onto the design space and configure them immediately. As you place additional steps, you visually define their relationships with one another.

Deployment package processes can run manually, automatically on some trigger condition, or on a user-defined schedule. When you want to deploy multiple applications or components that span applications, the deployment package determines which ones are executed and in which order.

For deployment package processes, you can run application and component processes and select from a menu of utility steps.

Deployment Package Process Design

1. Select from a set of options for the process designer, including the following:
    - Zoom In
    - Zoom Out
    - Actual Size
    - Print
    - Poster Print

- Save
- Cancel
- Show Outline Box: Use this to move the whole process design around in the design area. For example:
    - Drag its corner inward and move it over parts of the process design to zoom in.
    - Drag its corner outward again to zoom out.
- Show Connection Options: Select this to show the options for all connections.

2. In **Tools**, select process steps.
3. Use the Search box to find a step by name.
4. Select and drag steps to the design area to add them to your process. Select from the following:
    - Finish
    - Run Component Process
    - Run Application Process
    - Utility Steps
5. Connect process steps and edit or delete existing steps or connections, see Connecting Process Steps [page 242].
6. **Properties** are automatically shown when you select a step.
7. Click **Run** to initiate the **Run Deployment Package** process wizard.

Deployment Package Process Utility Steps

Several utility steps are provided for deployment package processes.

Acquire Lock and Release Lock

The **Acquire Lock** and **Release Lock** steps enable you to lock and release a deployment package process in a designated environment.

Locks are typically used to prevent specific processes from modifying the same resource at the same time, such as when a database is being updated.

In an Acquire Lock step, the process tries to acquire a lock with the specified name. If there is no other lock with that exact name, the process acquires the lock with that name and continues. If another process is using a lock with that exact name, the process waits until the lock is released by the other process. Any processes that access the resource without a lock or with a lock of a different name do not wait on this lock.

Locks are released automatically when the process completes, but you can also release the lock using the Release Lock step if you need to release it explicitly before continuing later steps in a process. Administrators can also release locks. See Managing Locks [page 108].

The following table describes the property for these steps.

Property	Description
Lock Name	Specify the lock name. The default setting is as follows: `${p:deploymentPackage.name}` **Tip:** The default lock name setting is set using a default variable. Therefore, to make the lock names exactly the same for different objects and processes, you can enter a static string for the name or use variables that will resolve to the same in both processes. For example, if you want to lock a resource against concurrent use for two processes in different components that are mapped to the same resource, you could use `${p:resource.name}` as the lock name for both.

Set Status

The **Set Status** step enables you to set the status of a deployment package process. The following table describes the property for this step.

Property	Description
Status	Values: Success or Failed

Process Sleep

The Process Sleep step enables you to enter time in seconds for a process to wait until continuing with the next steps in the process. The following table describes the property for this step.

Property	Description
Sleep Time	Default: 30

Join

The Join step enables you to join steps in a process. There are no properties for this step.

For the Join step to continue processing on to its outboard connections, all its inbound connection paths must be valid. To be valid, all that should evaluate to success do so, all that should evaluate to failure do so, all that should evaluate to any value do so, and all that should evaluate to a specific switch value do so.

You can do an implicit join without the Join step by having multiple inbound connections for another step. Regardless of whether it is an implicit or explicit join, all incoming connections must be valid for the joining step to be executed.

One reason you should choose an explicit Join step is when your process has multiple steps that must all complete to a certain outcome and then should continue on to multiple outbound steps without additional processing between them.

Manual Deployment Package Task

The Manual Deployment Package Task step enables you to interrupt a process until some manual intervention is performed according to a pre-configured deployment package task. Deployment package tasks must be set for a deployment package before you can select them for this step. See Deployment Package Tasks [page 333].

The following table describes the properties for this step.

Field	Description
Name	Enter a descriptive name for the step.
Task Definition	Select a previously-defined deployment package task. See Deployment Package Tasks [page 333].
Deployment Package Role	Select the role expected to respond. The user mapped to this role must respond to the generated approval request before the process can continue.

Adding Deployment Package Properties

Add properties to a deployment package for information that may have values specific to that package. If a deployment package is shared by multiple applications, and if all or most of those applications should use the same value for a deployment package level property setting, you should set the value at the package level. For more information, see How Properties Are Resolved [page 345].

To create deployment package properties:

1. Navigate to **Management > Deployment Packages > [select package]**.
2. Select the **Details** tab.
3. Select the **Properties** tab.
4. Click **Add Property**.
5. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.

Field	Description
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

6. Click **Save**.

To use a property in a deployment package process, reference it when you configure a step that uses it. Use the following syntax: `${p:<property name>}`

Running Deployment Package Processes

Deployment packages typically deploy artifacts for multiple applications.

For more about deployment, see Part 5: Deployment [page 363].

To run a deployment package process:

1. Navigate to **Management > Deployment Packages**.
2. Select a deployment package.
3. Click **Run**. The process wizard is displayed.
4. If the deployment package has a pipeline associated with it, select an execution type. Click **Next**.
5. If prompted, select one or more environments in which to run the package. These may be limited based on the pipeline or environment you select.
   - If you selected **Do not use pipeline**, selected environments do not initiate a pipeline execution even if you select the first environment in the pipeline.
   - If you select multiple environments, they are executed in parallel and steps that do not have environments use the same set of environments.
   - The list shows environments that have application environment counterparts in all used applications. If you select **Show all available environments**,

environments that are used in at least one of the used applications are added to the list.

Click **Next**.

6. If prompted, enter or select values. The selections for versions, snapshots, and other properties are taken from the application or component process properties.

7. Click **Submit** to start the process. When a process starts, the **Request** page is displayed with a visual indication of the progress. The status of each step is shown in color on the process graphic:

    - Blue: Waiting to start
    - Yellow: Running
    - Green: Succeeded
    - Red: Failed
    - Grey: Cancelled

    Information for each step is displayed in the process list.

See Chapter 30: Analyzing Deployments [page 371].

People notified for any approvals must approve or reject. See Approving Requests [page 291].

Note: If gate conditions are defined for a target environment, the component versions or snapshots are deployed into that environment only if their status meets those conditions. See Gates [page 295].

Deployment Package Inventory

When you deploy using a deployment package, the inventory is tracked in the deployment package and in included applications.

Select a deployment package **Inventory** tab to see the environment inventories that have been updated using that deployment package.

- For application processes, the inventory is taken from the application.
- A deployment package process execution updates both deployment package and application inventory. However, an application process executed outside of a deployment package updates only the application process. If an application process contained in a deployment package process is executed outside of the deployment package, it will appear as non-compliant in the deployment package.
- For component processes, the inventory is created.
- If a component version with different statuses or different component versions are deployed on the same environment, the one with the later date overrides earlier versions.
- Inventory for a deployment package is compliant for an environment only if all mapped resources for that environment are compliant.

Deployment Package Tasks

A deployment package task is used to interrupt a deployment package process until some manual intervention is performed. A task-interrupted process is suspended until the assigned user or users respond. Typically, deployment package manual tasks are removed after the process has been tested or automated.

Like approvals, triggered deployment package tasks notify assigned users. Unlike approvals, deployment package tasks can be incorporated within a deployment package process.

The person notified for the task can indicate that it is complete by approving the step from the request's log in **Approval Requests**. See Approving Requests [page 291].

Create and use tasks as follows:

1. Create deployment package tasks in deployment packages.

2. Insert a manual task step in a deployment package process.

   See Manual Deployment Package Task [page 330].

Creating Deployment Package Tasks

To create a deployment package task:

1. Navigate to **Management > Deployment Packages > [select deployment package]**.

2. Select the **Tasks** tab and click **Create**.

3. Name the task and select a template from the **Template Name** field.

   The individual tasks map to the notification scheme used by the deployment package. See Creating Notification Templates [page 147]. The default available task notification template is:

   - TaskCreated

Chapter 24: Managing Global Processes

You can define processes that are not associated with a particular application or component.

For details on global processes, see the following topics:

Creating Global Processes

A process is created in two steps:

1. Create the process to configure basic information, such as name, description, and default working directory
2. Design the process using the process editor

To create a global process:

1. Navigate to **Management > Global Processes**.
2. Click **Create**.
3. Fill in the fields as shown in the following table.

Field	Description
Name	Identifies the process; appears in many UI elements. Required.
Description	The optional description can be used to convey additional information about the process.
Default Working Directory	Required. Defines the location used by the agent running the process, for temporary file storage. The default value resolves to `*agent_directory*\work\*component_name_directory*`. The default properties work for most components; you might need to change it if a component process cannot be run at the agent's location.
Execution Timeout	Specifies the timeout in hours and minutes. Default: run indefinitely

4. To create additional global processes, click **Save and New**. To start configuring the global process you just created, click **Build Out**.

Designing Global Processes

You design processes using the process editor. In the process editor you drag process steps from a menu of steps onto the design space and configure them immediately. As you place additional steps, you visually define their relationships with one another.

For information on designing processes, see Designing Component Processes [page 241].

Global Process Utility Steps

Several utility steps are provided for global processes.

Acquire Lock and Release Lock

The **Acquire Lock** and **Release Lock** steps enable you to lock and release a process for a designated resource.

Locks are typically used to prevent specific processes from modifying the same resource at the same time, such as when a database is being updated.

In an Acquire Lock step, the process tries to acquire a lock with the specified name. If there is no other lock with that exact name, the process acquires the lock with that name and continues. If another process is using a lock with that exact name, the process waits until the lock is released by the other process. Any processes that access the resource without a lock or with a lock of a different name do not wait on this lock.

Locks are released automatically when the process completes, but you can also release the lock using the Release Lock step if you need to release it explicitly before continuing later steps in a process. Administrators can also release locks. See Managing Locks [page 108].

The following table describes the property for these steps.

Property	Description
Lock Name	Specify the lock name. The default setting is as follows: `${p:component.name}-${p:componentProcess.name}-${p:resource.name}` **Tip:** The default lock name setting is set using a default variable. Therefore, to make the lock names exactly the same for different objects and processes, you can enter a static string for the name or use variables that will resolve to the same in both processes. For example, if you want to lock a resource against concurrent use for two processes in different components that are mapped to the same resource, you could use `${p:resource.name}` as the lock name for both.

Set Status

The **Set Status** step enables you to explicitly set the status of a process. The following table describes the property for this step.

Property	Description
Status	Values: Success or Failed

Set Property

The **Set Property** step enables you to set a value of a property. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.
Property Name	Enter a name for the property.
Property Value	Enter a value for the property.

Switch

The **Switch** step enables you to define a conditional step. The outbound connections for this step are set to check for values to determine the ongoing path of the process. See Using Conditions and Switch Steps in Processes [page 249]. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.
Property Name	Enter a name for the property to be checked for values in the outgoing connections.

Join

The Join step enables you to join steps in a process. There are no properties for this step.

For the Join step to continue processing on to its outboard connections, all its inbound connection paths must be valid. To be valid, all that should evaluate to success do so, all that should evaluate to failure do so, all that should evaluate to any value do so, and all that should evaluate to a specific switch value do so.

You can do an implicit join without the Join step by having multiple inbound connections for another step. Regardless of whether it is an implicit or explicit join, all incoming connections must be valid for the joining step to be executed.

One reason you should choose an explicit Join step is when your process has multiple steps that must all complete to a certain outcome and then should continue on to multiple outbound steps without additional processing between them.

Manual Task

The Manual Task step enables you to interrupt a process until some manual intervention is performed. Manual tasks in global processes are submitted dynamically as part of the step.

The following table describes the properties for this step.

Field	Description
Name	Enter a name for the step. Include clearly identifiable information in the name as it appears in the notification about the task.
Users	Select the users who can approve or reject this task.
Groups	Select the groups who can approve or reject this task.
Template Name	Enter the template name to use for the task notifications. The default is `TaskCreated`.
Require Comment	Require a comment upon approval or rejection.
Comment Prompt	Enter a prompt for the comment box that appears in the approval form. For example: `Enter reason and phone number:`

All roles are required, so all affected users must respond before the process can continue.

Run Global Process

The **Run Global Process** step enables you to run a global process as part of the current process. The following table describes the properties for this step.

Property	Description
Name	Enter a name for the step.
Process	Select the global process name.
Resource Name	Enter the resource name. The default value is as follows: `${p:resource.name}`

Adding Global Process Properties

Add properties to a global process for information that may have values specific to the scope of that process.

To define a global process property:

1. Navigate to **Management > Global Processes**.
2. In the processes list, beside the process for which you want to define a property, click the **Details** icon (👁).
3. On the **Properties** tab, click the **Add Property** button.
4. Fill in the fields as shown in the following table.

Field	Description
Name	Property name
Description	Property description
Label	Enter a label to be associated with the property in the user interface.
Required	Select this if the property is required.
Type	Specify the type of expected value. The supported types are as follows: • text • text area • check box • select • multi select • secure The default type is `text`.
Default Value	Enter a default value (if any).

5. Click **Save**.

You can use this property in the same process where it is defined by referencing it in one of the steps. Use the following syntax: `${p:<property name>}`

Importing/Exporting Global Processes

Global processes can be exported and then imported into another server. This can be useful if you have multiple Deployment Automation servers, such as test and production servers, where you create and validate the processes on the test server and then export them and import them into the production server. From the **Global Processes** list you can export global processes.

- The Export option creates a JSON export file

- You can then import the JSON file into a global process on another server

Exporting Global Processes

Exporting a global process creates a JSON file that contains the global process's configuration information. For information about JSON, see www.json.org.

To export a global process:

1. Navigate to **Management > Global Processes**.
2. In the column beside a global process, click **Export.**

You can open the file in a text editor or save it. If you save it, a JSON file is created with the same name as the selected global process.

Importing Global Processes

When you import a global process, you can create a new global process or upgrade an existing one.

To import a global process:

1. Navigate to **Management > Global Processes**.
2. Click **Import**.
3. Select the JSON file containing the global process.
4. Click **Import**.

Running Global Processes

You can run global processes directly from the Global Processes page or you can run them from component and other global processes using the **Utility Steps > Run Global Process** step. For more about deployment, see Chapter 28: Deployment Overview [page 365].

To run a global process from the Global Processes page:

1. Navigate to **Management > Global Processes**.
2. Select a global process.
3. Click **Run**.
4. Select resources and options.
5. Click **Submit**.

When a process starts, the **Request** page is displayed so that you can review the process request progress. After the request completes, you can review the results on that and other related pages. See Chapter 30: Analyzing Deployments [page 371].

Chapter 25: Properties

Properties are parameters that are referenced in process steps. Values can be preset or resolved at runtime. For process properties, like plugin step properties themselves, a user interface is created so that users can enter values as input properties at runtime. Properties can also be written as output from a process. Property versions are maintained and remain available.

Property names can be user-defined or built-in. A number of built-in properties are available for referencing in your process steps. If there is no built-in property that suits your needs, you can define your own properties at the levels you may need them and designate the default values according to the logic of your configuration.

Property values are restricted to 4,000 characters.

For details on available properties, see the following topics:

- Built-in Properties [page 341]
- User-Defined Properties [page 343]
- All Properties Pairs [page 344]
- How Properties Are Resolved [page 345]
- Property Picker [page 346]

Built-in Properties

Deployment Automation has a number of built-in properties that can be used by your processes. These are described in the following table. To use a property in a process, reference it when you configure a step using the syntax shown in the **Reference Syntax** column.

Built-in Properties Table

Property Name	Reference Syntax	Description
agent.id	${p:agent.id}	Agent ID is the unique number Deployment Automation gives an agent at installation to distinguish it from others.
agent.name	${p:agent.name}	Agent name is a user-defined name given at installation that distinguishes the agent from others. The agent name can be changed by editing the agent's `conf/agent/installed.properties` file and restarting the agent.

Property Name	Reference Syntax	Description
application.id	${p:application.id}	An application ID is a unique number Deployment Automation gives the application at creation to distinguish it from others.
application.name	${p:application.name}	An application name is a user-defined name that distinguishes the application from others. It is entered when editing or creating a new application.
component.id	${p:component.id}	Component ID is the unique number Deployment Automation gives a component at creation to distinguish it from others.
component.name	${p:component.name}	A component name is a user-defined name that distinguishes the component from others. It is entered when creating or editing a component.
environment.id	${p:environment.id}	Environment ID is the unique number Deployment Automation gives an environment at creation to distinguish it from others.
environment.name	${p:environment.name}	An environment name is a user-defined name that distinguishes the environment from others. It is entered when creating or editing an environment.
resource.id	${p:resource.id}	Resource ID is the unique number Deployment Automation gives a resource at creation to distinguish it from others.
resource.name	${p:resource.name}	A resource name is a user-defined name that distinguishes the resource from others. It is entered when creating or editing a resource.
version.id	${p:version.id}	Version ID is the number assigned to a version when it is imported into CodeStation.

Property Name	Reference Syntax	Description
version.name	${p:version.name}	A version name is a user-defined name that distinguishes the version from others. It is given when the version is imported.
prevStep	${p:prevStep/<property name>}	This enables you to access previous step values without having to explicitly fill in the previous step name. Warning: If a step has multiple possible previous steps in a conditional process design, the results will be unpredictable.
logPath	${p:<step name>/logPath}	This is an output property that resolves to the full path of a given step's log file during process execution. For example, this can be used with the `prevStep` property as follows: `${p:prevStep/logPath}` **Important:** This functionality was introduced in Deployment Automation version 5.1.5. Processes that were executed before this functionality was implemented will not have the `logPath` property.

User-Defined Properties

If there is no built-in property that suits your needs, you can define your own properties at the levels you may need them. The user-defined properties are described in the following table. To use a property in a process, reference it when you configure a step using the syntax shown in the **Reference Syntax** column.

User-Defined Properties Table

Property Name	Reference Syntax	Description
agent/<property name>	${p:agent/<property name>}	Agent properties are user-defined and are set on the agent's **Properties** tab.

Property Name	Reference Syntax	Description
application/ <property name>	${p:application/ <property name>}	Application properties are user-defined and are set on the application’s **Properties** tab.
component/ property_name	${p:component/ <property name>}	Component properties are user-defined and are set on the component’s **Properties** tab.
environment/ property_name	${p:environment/ <property name>}	Environment properties are user-defined and can be set on the environment, application environment, and component **Properties** tabs. Values for same-named environment properties are overridden at the most specific level, with component being the most specific.
<property_name>	${p:<property name>}	You can define properties on a process's **Properties** tab and enter them in a process. If the property does not have an assigned value, the user running the process is prompted for the value.
resource/ <property name>	${p:resource/ <property name>}	Resource properties are user-defined and are set on the resource's **Roles & Properties** tab.
resource/<role name>/<property name>	${p:resource/ <role name>/ <property name>}	Resource role properties are user-defined and are defined on the resource's **Roles & Properties** tab. The values are set when adding a role to a resource.
<stepname>/ <property name>	${p:<stepname>/ <property name>}	You can reference a property name in a process step that is set in a previous step name in the same process. The property value is set as output from the named step.
system/<property name>	${p:system/ <property name>}	System properties are user-defined and are set in **Administration > System > Properties**. These can be thought of as global properties, and should be used to set values that are system-wide defaults.

All Properties Pairs

If you want to parse properties you have defined during a process, you can have all properties returned to subsequent steps. You can then search for a required property or property value and use that as part of your process. After the `allProperties` property is resolved, all of the properties for the given element are listed in a comma-separated series of `name=value`, including each property on the given element.

These are available only during process execution, and are not automatically written to the process request's Output Properties. You can echo the properties in a step of your process to list them to the output.

If you use the wrong property name, your property name string won't be replaced. For example, for component/allProperties, it would remain `${p:component/allProperties}`. If the system passes or echos an empty string, this indicates that there are no properties for the property name you specified.

All Properties Pairs Table

The available All Properties pairs are shown in the following table.

Property Name	Reference Syntax	Description
component/allProperties	${p:component/allProperties}	Selects all the properties with the same value in a given component.
environment/allProperties	${p:environment/allProperties}	Selects all the properties with the same value in a given environment.
resource/allProperties	${p:resource/allProperties}	Selects all properties with the same value in a given resource. **CAUTION:** ⚠ ${p:agent/allProperties} will return properties most of the time, but because it usually contains a carriage return, the step may fail.
system/allProperties	${p:system/allProperties}	Selects all properties with the same value in a given system.

How Properties Are Resolved

To use a user-defined property in a process, reference it when you configure a step that uses it. Use the following syntax:

```
${p:<element>/<property name>}
```

where the `${` indicates the start of the reference, `p` stands for property, `<element>` is the element the property is defined on, such as component, `<property name>` is the user-defined property name, and `}` indicates the end of the reference. If the property is defined for a process, you can reference that property in the same process as `${p:<property name>}`.

A running process fills in values for properties from various property settings, including the following:

1. system
2. environment
3. resource
4. resource role
5. application environment
6. application
7. application process
8. component environment
9. component
10. component process
11. component version
12. global process
13. properties set by process steps

For example, a component process step has access to properties set before or during the process execution, including those that are:

- set earlier by other steps within the process
- set within the scope of the currently executing process
- set by the component
- set by the application
- set for the target environment and resource
- set at the system level

Step property values become unavailable once the component process ends.

Typically, you set the value for a same-named property where the value is most likely to change. For example, the value for a web server port number would probably be set in the environment or resource properties. This helps to make the processes flexible enough to execute in multiple environments, such as INT, UAT, and PROD.

The process resolves the values from one of the settings in hierarchical order, with values given for properties closest to the process step overriding any previous values. If the property value is unresolved, the process can prompt users for information at runtime and then incorporate the values into the process.

Property Picker

A typical DA environment may contain hundreds of properties. You do not have to remember the names of all properties, instead use the property picker to find and select them. The picker displays properties:

- alphabetically in ascending order.
- that are based on your current context and any linked context.
- for which you have security permission.

You can use the property picker in the:

- Process designer
- Process wizard
- Properties editor

Note: The first time that you start a new DA server, the properties are indexed and the picker may take a while to display. In subsequent start ups the property picker is available immediately. To disable the property picker go to Administration | System | System Settings.

Using the property picker

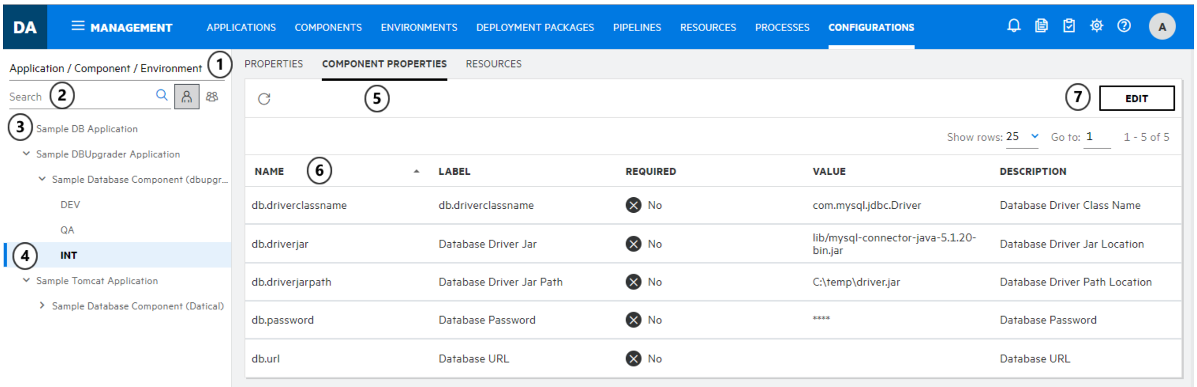
1. To invoke the property picker, in a properties field do one of the following:
   - Enter: `${`
   - Press the *Control* and *Space* keys together.

   The picker opens and displays a list of all available properties.
2. Do one of the following:
   - To find a property, scroll down the list.
   - To search for a property, start typing its name after the `${` characters. The list displays all the properties that match the string, for example:
     - To search for all properties that have 'da' in their name, type: `da`
     - To narrow the list and search for agent properties that have 'da' in their name, type: `agent.da` or `agent/da`
3. To close the property picker, enter `}` or press ESC.

Chapter 26: Configuring Within the Application Hierarchy

You can configure properties, resources, and configuration templates directly through the application hierarchy on the **Configuration** page.

Configuration Page



NAME	LABEL	REQUIRED	VALUE	DESCRIPTION
db.driverclassname	db.driverclassname	☒ No	com.mysql.jdbc.Driver	Database Driver Class Name
db.driverjar	Database Driver Jar	☒ No	lib/mysql-connector-java-5.1.20-bin.jar	Database Driver Jar Location
db.driverjarpath	Database Driver Jar Path	☒ No	C:\temp\driver.jar	Database Driver Path Location
db.password	Database Password	☒ No	****	Database Password
db.url	Database URL	☒ No		Database URL

1. Navigate to **Management > Configuration**.

2. Select how the Deployment Automation elements are sequenced in the hierarchy:

   - Application / Component / Environment

   - Environment / Application / Component

   - Component / Application / Environment

3. Search for the name of the application, component, or environment you want to configure. You can also filter the results by owner—those owned by the current user or those owned by all users.

4. Scroll the list to see the applications, components, and environments. The results are displayed based on the selected sequence, search key, and owner filter.

5. Select the application, component, or environment you want to configure.

6. The following tabs are available

   - For applications: Properties

   - For components: Properties, Environment Properties, Version Properties, and Templates (configuration templates)

   - For environments: Properties, Component Properties, and Resources (resources and resource groups)

7. View the listing for your selection.

8. Use the edit, add, or create button that appears for your selection to add property values, add properties, create resources, and create configuration templates.

Chapter 27: Replication

You can export and import whole sets of Deployment Automation objects using replication.

You can export from a source server and import on a target server to replicate a cluster of objects from one server to another. This can be used to maintain automation data between master and slave servers, even across an air gap.

Data is exported to and imported from a single file. If the servers are not physically connected, a flat file that contains source server data must be manually uploaded to the target server.

Replication is implemented as follows:

- You can select a set of applications that you want to export. All the linked objects, such as components, environments, and pipelines, are exported automatically with the applications.
- You can select a set of environments that you want to export. If an environment that is part of a pipeline is omitted from the selection, the pipeline is also omitted from the export.
- Exported files contain component versions with files of the associated components.
- Snapshots can be used instead of component versions if needed.
- Export and Import works with a single file containing all the data.
- Process execution history is not exported.
- Component mapping works only if the target server has agents with the same names as those on the source server.
- During the import procedure, the creation date for all objects is set to that of the current system. The creation date from the source server is ignored.
- All objects imported to the target server will have the same ID as corresponding objects from the source server. The import procedure identifies whether the object was previously imported from the source server based on its ID.

For details, see the following topics:

- Exporting Replication Data [page 352]
- Importing Replication Data [page 353]
- Replication Export Details [page 355]
- Replication Import Details [page 357]
- Replication Merge Strategies [page 360]

Exporting Replication Data

You can export replication data for one or more applications. You would typically export replication data from a primary or master server. You can also use the replication data to share processes across testing and demonstration servers.

To export replication data:

1. Navigate to **Management > Applications**.
2. From the **Replication** menu select **Export**.
3. Select the applications that you want to export and click **Next**.
4. Select the environments that you want to export and click **Next**.
5. Select the export options as follows:

Option	Description
Component versions	Determines which component versions to export. Select one of the following: - **Deployed**: include all deployed versions from the current inventory status of the environments (default) - **All**: include all component versions - **Latest**: include only the latest versions of the components - **None**: do not include component versions **Note:** Inactive component versions are not exported even if all component versions are selected.
Versions objects	Determines which property sheet or process versions to export. Select one of the following: - **Latest**: include only the latest versions (default) - **All**: include all versions
Include snapshots	Select this to include snapshot information in the export. The default is selected. **Note:** If snapshots are included in the export, all related component, process, and property sheet versions will be exported with them regardless of the selections for **Component versions** and **Versioned objects** options.

Option	Description
Include plugins	Select this to include plugins in the export. The default is selected.
Export format	Determines the metadata export format. Select one of the following: • **JSON** (default) • **XML**

6. Click **Export**.

The exported replication data is saved in a single zipped file that contains the file type designated in the **Export format** option. Component version files will be saved within folders named by component ID and version ID. The zipped file is saved to the downloads location defined in your web browser.

Importing Replication Data

You can import replication data that has been exported from another server. You would typically import data into a secondary or slave server.

To import replication data:

1. Navigate to **Management > Applications**.
2. From the **Replication** menu select **Import**.
3. Select the replication data file that you want to import.
4. Select the import options as follows:

Option	Description
Merge strategy	Determines the merge strategy to be used during import. Select one of the following: • Override • Merge without conflicts • Merge with master priority • Merge with slave priority For more details, see Replication Merge Strategies [page 360].
Disable auto-import	Select to disable automatic import for component versions during the import procedure. The default is selected. For information on automatic import of component versions, see Importing Versions Automatically [page 232].
Override resource mapping	Select this if you want resource mapping for environments and application environments to be imported and the existing resource mapping overridden. The default is deselected.

5. From this point you can either import the export file information as-is, or you can selectively import as follows:

   - To import without changing the application and environment selections, click **Finish**.

   - To change the application and environment selections, click **Next**.

6. If you selected **Next**, continue as follows:

a. Select the applications that you want to import and click **Next**.

b. Select the environments that you want to import and click **Import**.

The imported replication data is populated on this server as defined in the import options.

Replication Export Details

The following sections explain how replication export processing is done for each kind of object that is exported.

Dependency Analysis

The export procedure automatically identifies all dependencies associated with the selected applications set, such as components, component templates, global processes, environments, resources, resource groups, agent pools, plugins, statuses, notification schemes, resource roles, and security roles. Inactive component versions, snapshots, and environments are excluded from the export. The rest of the objects are exported regardless of their status.

If the **Include Snapshots** option is selected, the export procedure automatically identifies all the versions used in snapshots and adds those versions to the exported file. Therefore, even if the **Versioned Objects** parameter is set to *Latest*, earlier versions will be added to the exported file if they are used in snapshots and the **Include Snapshots** option is selected.

Environments

Environments selected in the export wizard are exported.

List of items exported with an environment:

- List of Resources (only resources associated with selected environments are included)
- Associated Pipeline (included only if all environments in the pipeline are selected for export)

Applications

Applications selected in the export wizard are exported.

List of items exported with an application:

- List of Environments (only environments selected for export are included)
- List of Components
- Component Mapping
- Application Processes
- Application Snapshots
- Tasks Definition
- PropSheet
- Associated Pipeline

Inactive applications and environments within the application are not exported.

Components

List of items exported with a component:

- Component Versions (both full and incremental versions are supported)
- Component Processes
- Post-processing scripts
- Tasks Definition
- Templates
- PropSheet

Global Processes

List of items exported with a global process:

- PropSheet
- Post-processing scripts

Although a Global Process is a versioned object, only the latest version of a Global Process is exported from the source environment. This happens because users can't use a Global Process version on the source server other than the latest one.

Component Templates

List of items exported with a component template:

- Processes
- PropSheet

Pipelines

Only those pipelines that are associated with applications are exported.

Environments

Only those environments that are associated with applications are exported.

List of items exported with an environment:

- Resource Set

Resources

Only objects used in environment resource sets are exported.

Resource Groups

Only objects used in environment resource sets are exported.

Agent Pools

Linked agents are not included in the export. Only agent pools used in exported resource sets are exported.

Plugins

If the **Include plugins** option is selected, the export procedure automatically determines which plugins are used in the exported processes and includes them in the data file. Only used plugins are included.

Statuses

Only statuses used in exported component versions and snapshots are included.

Notification Schemes

Only notification schemes used in exported applications are included.

Resource Roles

Only resource roles used in exported resources are included.

Security Roles

Only security roles used in exported processes, notification schemes, and version statuses are included.

Export Errors

You cannot export a process version if it invokes another process that has already been deleted, uses a plugin that has been removed from the system, or references a security role that has been removed. In these cases, the export procedure tries to remove the specific version of the process from the exported file. However, if after the process version is removed there are no versions of the process selected that can be exported, an error is generated and the export operation cancelled.

You cannot export component and global processes that contain recursive process invocation; for example, where process A invokes process B and process B in turn invokes process A. If you attempt this, the export procedure will immediately fail. Export parameters should be set up to exclude any recursive process invocation.

Replication Import Details

The following sections explain how replication import processing is done for each kind of object that is imported.

Environments

Environments that are exported are automatically selected for import. However, they can be de-selected in the import details if needed.

An environment is created on the target server if an environment with the given name does not already exist. If an environment with the given name already exists and its ID is different from the ID of the imported environment, an error is generated and the import operation is cancelled. During environment creation, the set of resources and resource groups associated with the environment are recreated on the target server based on resource and resource group IDs.

A list of resources and resource groups are recreated for the existing environment if the **Override resource mapping** option is specified during the import procedure.

Component Templates

Component templates are fully imported to the target server. If a component template with the given name already exists and its ID is different from the ID of the imported component template, an error is generated and the import operation is cancelled. If the

IDs match, a new version of component template is created based on the data provided in the data file.

Components

Components are fully imported to the target server. Component versions (artifacts) contained in a data file are imported as well. If a component with the given name already exists and its ID is different from the ID of the imported component, an error is generated and the import operation is cancelled. If the IDs match, the existing component parameters are overridden based on the data provided in the data file. If the component was deleted on the target server but it exists in a data file, the component is recreated on the target server.

Old versions (artifacts) are not deleted, but any new versions are imported to the component. There are no limitations on using component version type for imported component versions. Both full and incremental modes are fully supported by the import procedure.

Global Processes

Global processes are fully imported to the target server. If a global process with the given name already exists and its ID is different from the ID of the imported global process, an error is generated and the import operation is cancelled. If the IDs match, a new version of the global process is created based on the data provided in the data file.

Applications

Applications that are exported are automatically selected for import. However, they can be de-selected in the import details if needed.

Applications are fully imported to the target server. If an application with the given name already exists and its ID is different from the ID of the imported application, an error is generated and the import operation cancelled. If the IDs match, the existing application parameters are overridden based on the data provided in the data file.

Environment to environment linkage will be performed based on environment names rather than IDs.

Component mapping are performed based on the existing resource set. If some of the resources are missing, they are omitted during the component mapping import.

Application snapshots are fully imported, including dependencies on the component versions, application processes, component processes, properties, and other objects. During the import procedure, the snapshots are automatically linked to the imported versions of the corresponding objects used on the target environment.

Resources

Resources are fully imported to the target server. If a resource with the given name already exists and its ID is different from the ID of the imported resource, an error is generated and the import operation cancelled. If the IDs match, the existing resource parameters are overridden based on the data provided in the data file.

If an environment is deselected for the import and resources are dependent on that environment, those resources will not be imported.

Resource Groups

Resource group meta-information is fully imported to the target server.

Agent Pools

Only basic meta-information related to agent pools is imported to the target server.

Plugins

During the import procedure, all plugins with the same version as those contained in the data file are overridden on the target server. Plugins added on the target server are not deleted during the import procedure.

Statuses

The same rules apply as for the Resources import.

Notification Schemas

The same rules apply as for the Resources import.

Resource Roles

The same rules apply as for the Resources import.

Security Roles

The same rules apply as for the Resources import.

Pipelines

The same rules apply as for the Resources import.

If an environment is deselected for the import and it is part of a pipeline, that pipeline will not be imported.

Versioned Objects

Versioned objects, such as property sheets and processes, are imported according to the merge technique described below. According to this technique, new versions are added on the target server.

The following example explains what happens if the **Versioned objects** option is specified as *All*.

1. Assume there are 2 versions of a property sheet on the master server:

   Master 1, 2

2. The replication procedure recreates these versions on the slave server:

   Slave 1, 2

3. After this, a release engineer decides to make some changes on the slave server:

   Slave 1, 2, 3*

4. Another release engineer makes changes on the master server:

   Master 1, 2, 3

5. In the end, replication is performed and all versions added on the master server since the last import will be imported as new versions.

   Master 1, 2, 3*, 4 (3)

In this example, if there was a snapshot referring to version 3 on the master server, it will refer to version 4 on the slave server after the import procedure is performed.

The replication mechanism keeps track of all the imports that were previously performed on the target environment and during import, omits versions that have already been imported to the target environment. In the preceding example, versions 1 and 2 from the source server won't be imported to the target sever during the second import because they were already imported during the first import.

This technique can be used to get rid of unnecessary versions on the slave server during the replication procedure. The following example explains what happens if the **Versioned objects** option is specified as *Deployed*.

1. Assume that we have 2 versions of a property sheet on the master server and only version 2 is used by the deployed process:

   Master 2

2. The replication procedure recreates this version on the slave server:

   Slave 1(2)

In the example above, if there was a snapshot referring to version 2 on the master server, it will refer to version 1 on the slave server after the import procedure is performed.

Replication Merge Strategies

You can specify a properties merge strategy for the import procedure. Properties can be merged according to one of the following strategies:

- **Override**: The new version is created solely based on data from a data file. Target server changes are ignored and will be overridden.

- **Merge without conflicts**: The new version is created as a result of a properties merge from source and target servers. If a property was changed on either the target or source server, the merge result will be successful and the updated property value will be used. However, if conflicting changes were made to the property on the target and source servers, an error will be generated and the import operation will be cancelled.

- **Merge with master priority**: This is the same as the merge without conflicts strategy but when a conflict arises, it is resolved using the source server property change.

- **Merge with slave priority**: This is the same as the merge without conflicts strategy but when a conflict arises, it is resolved using the target server property change.

Results for each merge strategy using example data cases are shown in the following table. These assume the import has already been performed once from the source server to the target server.

Initial data before second import		Properties Merge Strategy and import result on Target Server (Slave)			
on Master (M)	on Slave (S)	Override	Without conflicts	Master priority	Slave priority
No changes	No changes	No changes	No changes	No changes	No changes
No changes	Delete	Data restored from M	Data deleted	Data deleted	Data deleted
No changes	Edit	Data restored from M	Edited data from S	Edited data from S	Edited data from S
Delete	No changes	Data deleted	Data deleted	Data deleted	Data deleted
Delete	Delete	Data deleted	Data deleted	Data deleted	Data deleted
Delete	Edit	Data deleted	Conflict, stop import	Data deleted	Edited data from S
Edit	No changes	Edited data from M	Edit data from M	Edit data from M	Edited data from M
Edit	Delete	Edited data from M	Conflict, stop import	Edit data from M	Data deleted
Edit	Edit	Edited data from M	Conflict, stop import	Edit data from M	Edited data from S
Add data on M	Add data on S	Only added M data	Added M and S data	Added M and S data	Added M and S data

Part 5: Deployment

This section contains information on deployment in Deployment Automation.

Included are the following topics:

Chapter 28: Deployment Overview

You typically manage deployments in the **Deployment** page of the user interface. The **Deployment** page is one of several places you can view deployment-related information.

Following are some of the selections on this page.

- **Timeline**

  See Chapter 30: Analyzing Deployments [page 371].

- **Inventory**

  See Chapter 31: Inventory Management [page 377].

- **Reports**

  See Chapter 32: Deployment Report Overview [page 379].

- **Schedule**

  Select from options to schedule a deployment to run later and set blackout dates. See Scheduling Deployments [page 368] and Setting Deployment Blackout Dates [page 368].

- **Run**

  Select a deployment to run now. See Chapter 29: Ways of Deploying [page 367].

- **Action**

  Select from deployment actions, including the following:

  - Print Page
  - Help

Chapter 29: Ways of Deploying

There are several ways to run processes in Deployment Automation. Although not all processes are deployment processes, for simplicity, we refer to running processes as "deploying" here. Deployment options are available in both **Management** and **Deployment** pages so that deployment can be done by those users with access to only one or the other in the user interface.

Regardless of where you select the **Run** option, the process wizard is displayed. Depending on the context, some of the selections are pre-selected.

- In **Management**, you run processes from the element whose processes you want to run: deployment package, application, component, or global.
- In **Deployment**, you can select and run a process using the **Run** option in the global menu. In this menu, you can select from the following:
    - **Application Process:** This invokes the application process wizard.
    - **Promote Application Process:** This invokes the application process wizard and lets you select and run a process in the next environment in a pipeline. You will see only applications that have a pipeline in them and will see only those environments where there are pending pipeline executions. This is used to continue promotion for pipelines that do not require auto-promote, and the process must already have been run in a prior environment in the pipeline.
    - **Deployment Package:** This invokes the deployment package process wizard.
    - **Promote Deployment Package:** This invokes the deployment package process wizard and lets you select and run a process in the next environment in a pipeline. You will see only applications that have a pipeline in them and will see only those environments where there are pending pipeline executions. This is used to continue promotion for pipelines that do not require auto-promote, and the process must already have been run in a prior environment in the pipeline.
    - **Component Process:** This invokes the component process wizard.
    - **Global Process:** This invokes the global process wizard.

See the following for running processes from **Management**. The procedures are the same for running the same type of processes from **Deployment** other than the initial navigation and any pre-selected information based on context.

- Running Deployment Package Processes [page 331]
- Running Application Processes [page 288]
- Running Component Processes [page 255]
- Running Global Processes [page 340]

For more information on deploying, see the following topics:

- Scheduling Deployments [page 368]

- Setting Deployment Blackout Dates [page 368]

Scheduling Deployments

Deployment Automation has a built-in deployment scheduling system for setting up deployments that occur on a regular basis.

You can schedule deployments for applications and deployment packages into application environments.

To schedule a deployment:

1. Do one of the following:
   - In **Deployment**, in the global menu, click **Schedule** and then select **Application Process**.
   - In **Management > Environments > (selected environment)**, in the global menu, click **Schedule** and then select **Application Process** or **Deployment Package.**
2. Follow the **Schedule Application Process** wizard to make your selections. Selections are relevant for deployment requests for applications, depending on you are deploying.
3. In **Options**, give the date and time you want the deployment to run. Click **Make Recurring** to deploy the application on a regular schedule. For example, if you are practicing continuous delivery, you could select the **Daily** option to deploy applications to the target environment every day.
4. When finished, click **Submit** to start the process, which will remain pending until the scheduled time.

The selected processes do not start until the scheduled time. After you have scheduled the deployment, it will be added to the Timeline. In the Timeline grid or table view you can you can view, edit, and delete the deployment.

Setting Deployment Blackout Dates

A *blackout* can be set at the global environment level or application environment level. Once set, no deployments or snapshots can be scheduled to occur in the designated environment.

To set up a blackout:

1. In **Deployment**, in the global menu, click **Schedule** and then select one of the **Blackout** options as follows:
   - Application Environment Blackout
   - Global Environment Blackout
2. Follow the **Schedule Blackout** wizard to make your selections. Selections are relevant for application or global environments, depending on at what level you are adding the blackout.
3. Select an environment for which to set the blackout.

4. Select the date and time range for which you want the blackout to remain in effect. Optionally, you can enter a name for the blackout.

5. Click **Submit**.

In the Timeline, the blackout dates now have a black icon to indicate a blackout has been set. An A or G indicate the environment level for the blackout, application or global. In the Timeline grid or table view you can view, edit, or delete the blackout.

Important: If any previously-scheduled deployments to the environment are within the blackout date range, they will fail.

Chapter 30: Analyzing Deployments

Deployment Automation provides a rich set of information about your deployments into the application environments. Drill down to see successive levels of detail.

Following are some of the ways you can analyze the results of your deployments into your environments and ensure the environments have the component versions and configurations you need.

- Viewing Timelines [page 371]
- Viewing Application Deployments [page 372]
- Viewing Deployment History [page 373]
- Viewing an Application Process Request [page 373]
- Viewing a Component Process Request [page 374]
- Deployment Results in E-mail Notifications [page 374]

Viewing Timelines

Timelines show overview information on the processes run and link to more detail for convenient access to information.

You can view timeline information by navigating to:

- **Deployment > Timeline:** Timeline of all deployments run in the given time range for this Deployment Automation server
- **Management > Environments > (selected environment) > Timeline**: Timeline of all deployments run in the given time range for each application environment based on this environment

On a timeline you can do the following:

- Select to show deployments for applications, components, snapshots, or deployment packages. (Applications and deployment packages selections are available in environment timelines.)
- Search for a request name; this narrows the display to just the entries including that name.
- Filter by deployments owned by you or by everyone.
- Click the calendar view icon to see the calendar view.
- Click the grid view icon to view a list of processes.
- Use the arrows to change the year, month, week, or day viewed in the calendar, depending on the period selected.

- Select the period to show.
- In the month and year views, click anywhere in the blank area of a block to show the list of requests for that date block in a tabular view.

  Depending on the view and selection, columns include the following:

  - Status
  - Time
  - Name (click to go to the process request page)
  - Application (click to go to the application page)
  - Component for the process (click to go to the component page)
  - Process Version
  - Environment (click to go to the application environment page)
  - Process Version
  - Deployment Owner
- In the month view, status icons give you a quick view of the health of the processes. Point to a status icon for a quick view of information about the request. Click the icon to open the request page.
- In the month view, icons indicate scheduled processes and global and application environment blackouts. Point to an icon for a quick view or click the icon to open the related page.

  Tip: In the tabular view, you can view, edit, and delete blackout and schedule entries.

Related Topics

Chapter 31: Inventory Management [page 377]

Configuring and Monitoring Environments [page 298]

Viewing Application Deployments

Select an application's **Activity** tab see its deployment summary. This gives a graphical overview and allows you to select a time range of information to view. Information given includes the following:

- **Deployment Summary**

  See a graphical view of deployments by status in the selected time period.
- **Next Scheduled Deployment**

  See the date and time the next deployment for this application is scheduled.
- **Last Scheduled Deployment**

  See the date and time the last deployment for this application was run.

- **Deployments to Environments by Process**

  View a list of processes with the number of times they were run in each environment in the selected time period. You can select the time period independently of the Deployment Summary view.

- **View Deployment History**

  In one click go to the application's **Deployment History** page.

Related Topics

Chapter 31: Inventory Management [page 377]

Viewing Timelines [page 371]

Configuring and Monitoring Environments [page 298]

Viewing Deployment History

You can view deployment history in the **History** tab of application, application environment, and component pages.

To view the applications that have been deployed into a particular environment:

1. Navigate to **Applications > [select application]**.
2. Select the **Environments** tab.
3. Click the name of an environment and select **View Details**.
4. Select **History > Deployment History.** From there you can click name links and the **View Execution Details** icon to view deployment request information.

Viewing an Application Process Request

There are several ways to navigate to application process request details in context as you run processes and view deployment information.

To navigate from anywhere to the application request details:

1. Navigate to **Management > Applications > [select application]**.
2. Select the **History** tab.
3. Select the **Deployment History** tab.
4. Locate the application in the list of processes and click **View Execution Details**.

   Note: The process list item indicates the environment in which the process was run.

5. Click **Repeat Request** to repeat the process request using the same options.
6. The tab display the following information:
   - **Log:** View the results of each process step.
   - **Properties:** View the properties used in the process.

- **Manifest:** View the artifacts deployed by the process.

On the Log tab you can.

- Expand process steps to see information about each step and link to additional information.
- Click a resource name to go that resource's page.
- Click a version name to go to the component version's **Artifacts** tab where you can view and download the artifacts for this version.
- Click **Details** to see the step-by-step breakdown of the execution.

Viewing a Component Process Request

To view the component process request details, you can view them in context immediately after running the process or you can navigate to the details in several different ways. If you continue to drill down into the details when viewing an application process request, you will see the component request details. You can also navigate to the component process request details directly from the resource on which you ran the process.

To view the details for a process request for a selected resource:

1. Navigate to **Management > Resources > [select resource]**.
2. Select the **History** tab.
3. Select the **Deployment History** tab.
4. Locate the application in the list of processes and click **View Execution Details** . The component process request results for each step in the process is displayed. From here you can review the deployment's output log, error log, and input/output properties.

On the Log tab you can:

1. View an overview of each step.
2. Click the link in the **Type** column to view the artifacts.
3. Click one of the following icons to view more information:
   - **Output Log:** View the output log for the selected process request.
   - **Error Log:** View the error log for the selected process request.
   - **Input/Output Properties:** View a dialog that lets you toggle between views of the input and output property values for the selected process step.

Deployment Results in E-mail Notifications

If email notification schemes are configured, deployment results are sent to the email IDs of users with the selected roles as follows:

- The body of a notification email shows the details and the success or failure status of the deployment for each target.

- If the email notification scheme is configured to attach logs, the email notification includes a log file for each target that shows the success or failure status of each step.

Attached log files may be compressed into a zip file, depending on whether the maximum archival level is met by the number of log files to attach. The zip file includes a directory for each target with subdirectories for the log files for each step.

The log file naming convention and structure are as follows:

- If the log files are not zipped, they have the following naming convention:

  ```
  CompProcessStepIndex_CompProcessStepName_CompletionStatus.txt
  ```

  Example

  ```
  1_Download Artifacts_SUCCEEDED.txt
  ```

- If the log files are zipped, each zip file is named the same as the corresponding application process and has the following structure and naming convention:

  ```
      AppProcessStepIndex_AppProcessStepName
        AgentName
          ComponentVersionName (if more than one version is deployed)
            CompProcessStepIndex_CompProcessStepName_CompletionStatus.txt
  ```

  Example

  Zip file name:

  ```
  helloWorld App Process.zip
  ```

  Zip file contents:

  ```
  1_hello_worldInstall\test_agent1\1_Download Artifacts_SUCCEEDED.txt
  ```

Related Topics

Configuring Notification Schemes [page 145]

Chapter 31: Inventory Management

You can track the components, snapshots, and configurations deployed to any resource by using the inventory management features provided in Deployment Automation.

Only the latest process request is added to the inventory, rewriting all previous ones. Operational process types are not added to the inventory.

You can view inventory information on **Inventory** tabs in several places in the user interface, including the following:

- The **Deployment** page
- A selected component: shows applications and resources with that component
- A selected resource: shows the current inventory of component versions for that resource and the configuration inventory, which includes properties for all levels for that resource
- An application environment's details: shows component version inventory within each resource for that environment; you can toggle the view to see versions by component or by resource
- A selected environment: shows component version inventory for each application environment based on this environment; you can click the link in the **Environment** column of an inventory item to go to the application environment view

Inventory statuses can be used to associate more meaningful information with the component versions shown in the resource inventory.

To compare the inventory of two environments, see Comparing Environments [page 274].

Following is additional information about inventory management.

- Viewing the Environment Inventory [page 377]
- Advanced Inventory Management [page 378]

Viewing the Environment Inventory

Select an application in the **Deployment > Inventory** page to see the inventory information for each environment associated with the application.

On an **Inventory** page with an application selected you can:

- Click **Resources** to get a quick view of the resources associated with the environment. From there you can click **View Resources** to open the resources tab for the environment.
- Compare the inventories of up to four environments.
- Use the search box to narrow the display to those components you are interested in viewing.

- View the components and latest compliant version of each component. Click a component name to open the page for that component.
- Click the version name to go to the component version's **Artifacts** tab, where you can view and download the artifacts for this version.
- See whether each environment is compliant. An environment is compliant if the last requested processes were successfully executed in that environment.

This is a subset of the information shown on the application's **Environments** page. For more details, see Application Environments [page 269].

Advanced Inventory Management

When you create an application process, if you set the **Inventory Management** field to **Advanced**, an additional step appears in the Process Editor **Component / Process** window called **Inventory Update**. You can use this step to implement manual inventory management for application processes.

The options for the **Inventory Update** step are as follows:

- Add Desired Inventory
- Remove Desired Inventory

By managing your own inventory, you can control the compliancy of the inventory on the environments based on success or failure of the deployment process. For example, if you are trying to install a specific version of the component and the deployment fails, you can remove that component from the inventory, so that it isn't added and the environment becomes non-compliant. If it is successful, you can add the desired component version to the inventory and the environment remains compliant.

If you use automatic inventory management, for a failed step, the inventory won't be updated, but the environment will remain compliant. For a successful step, both types of inventory management update the inventory and the environment remains compliant.

Related Topics

Using Inventory Statuses [page 111]

Creating Application Processes [page 275]

Inventory Update [page 287]

Chapter 32: Deployment Report Overview

Deployment reports contain historical information about deployments, such as the total number executed and their average duration. On a report page you can:

- Select from default reports or reports that you've saved with specific report criteria. When you select the report that you want to run, the selected report appears on the page.
- Select from report filters, which vary by report type.
- Run the report to apply your filter selections.
- Save the report selections. Saved reports appear in the **My Reports** section of the selection box.
- Export the report. The tabular data is downloaded in HTML format.
- Print the report.
- View a graphical or listing report, depending on the report type.
- For graphical reports, view the report key to see the meaning of lines by color and graph point shape.
- For graphical reports, you can hold your cursor over a graph point to see more information, or grab a point on the graph and drag to zoom in or out.

For information on each report, see the following topics:

Deployment Average Duration Report

The *Deployment Average Duration Report* provides average deployment times for applications executed during a user-specified reporting period. The report provides both a tabular presentation and line graph of the data. Each table row represents an environment used by an application for the reporting period and interval.

The line graph elements are:

- y-axis represents deployment duration average times
- x-axis represents reporting intervals
- plot lines represent environments used by the applications

The units along the y-axis are scaled to the number of records reported. The units along the x-axis represent the reporting interval, which can be: months, weeks, or days. Each color-coded plot line represents a single environment used by the deployment during the reporting period.

When selected, the report runs automatically for the default reporting period (current month)and reporting interval (days), and with all filters set to `Any`. The default report provides average deployment times for all deployments that ran during the current month.

For details on configuring and running this report, see the following topics:

- Deployment Average Duration Fields [page 380]
- Running the Deployment Average Duration Report [page 380]
- Sample Reports: Deployment Average Duration [page 382]

Deployment Average Duration Fields

The deployment average duration fields are shown in the following table.

Average Duration Fields

Field	Description
Application	Name of the application that executed the deployment.
Environment	Name of the environment used by the application.
Reporting Interval	The remaining columns display the average deployment times for the reporting interval.

Running the Deployment Average Duration Report

To run a report:

1. Set the reporting period.

   Use the **Date Range** selection to set the report's start and end dates. The selected values determine the columns in the tabular report and the coordinate interval on the graph's x-axis. Default value: `Current Month`.

   Date Range

Field	Description
Current, Prior Week	Start day is either Sunday or Monday, depending on what is defined in your system. Reporting interval is set to days.
Current, Prior Month	Start day is first day of the month. Reporting interval is set to days.

Field	Description
Current, Prior Quarter	Quarters are bound by calendar year. Reporting interval is set to weeks.
Current, Prior Year	Reporting interval is set to months.
Custom	Displays the **Custom** pop-up which enables you to pick an arbitrary start and end date.

The start-time is automatically set to 00:00 (24-hour clock) for the selected date in the reporting period.

The end-time is automatically set to 24:00 for the selected date.

2. Set report filters.

   You can select from the following filters.

   Filters Table

Field	Description
Application	Only deployments executed by the selected applications appear in the report. To select applications: a. Click the **Application** button. b. To include an application in the report, click the corresponding check box. If a large number of applications are listed, type the first few letters of the application's name in the text box to scroll the list. Multiple applications can be selected. c. Click **OK**.
Status	Only deployments with the selected status appear in the report. Default value: `Success or Failure`, which means all deployments.
Plugin	Only deployments that used the selected plugin appear in the report. Default value: `Any`. **Note:** `Any` value also includes deployments that did not use a plugin.

3. Run the report.

   Click **Run** to apply your filter conditions to the data and produce the report.

A line graph and tabular report are displayed. Each environment used by a reporting application is represented by an individual plot line and table row.

You can interact with the view to show or hide information, as follows:

- Hide/Show a plot line by clicking the corresponding item in the graph legend.
- See information about a graph coordinate by moving your pointer over the graph point.
- Zoom a graph area by dragging the pointer over the area. Click Reset Zoom to return to the normal view.

Sample Reports: Deployment Average Duration

The following table contains examples of reports that can be produced using the Deployment Average Duration Report.

Sample Reports Table

Field	Description
Show me: Average durations for two specific applications during the past ten days that used a particular plugin	• **Application**: Select both applications from the Applications dialog. • **Status**: `Success or Failure` • **Plugin**: Select the plugin from the drop-down list box. • **Date Range**: Use the Custom pop-up to set the ten-day range.
Show me: Average durations for successful deployments for a given application during the past six months	• **Application**: Select the application from the Applications dialog. • **Status**: `Success` • **Date Range**: Use the Custom pop-up to set the range to the previous six months.

Deployment Total Duration Report

The *Deployment Total Duration Report* provides total deployment times for applications executed during a user-specified reporting period. The report provides both a tabular presentation and line graph of the data. Each table row represents an environment used by one of the selected applications for the reporting period and interval.

The line graph elements are:

- y-axis represents deployment duration times
- x-axis represents reporting intervals
- plot lines represent environments used by the applications

The units along the y-axis are scaled to the number of records reported. The units along the x-axis represent the reporting interval, which can be: months, weeks, or days. Each color-coded plot line represents a single environment used by an application during the reporting period.

When selected, the report runs automatically for the default reporting period (current month)and reporting interval (days), and with all filters set to `Any`. The default report provides total deployment times for all deployments that ran during the current month.

For details on configuring and running this report, see the following topics:

- Deployment Total Duration Fields [page 383]
- Running the Deployment Total Duration Report [page 383]
- Sample Reports: Deployment Total Duration [page 385]

Deployment Total Duration Fields

The deployment total duration fields are shown in the following table.

Total Duration Fields Table

Field	Description
Application	Name of the application that executed the deployment.
Environment	Name of the environment used by the application.
Reporting Interval	The remaining columns display the total deployment times for the reporting interval.

Running the Deployment Total Duration Report

To run a report:

1. Set the reporting period.

   Use the `Date Range` date-picker to set the report's start- and end-dates. The selected values determine the columns in the tabular report and the coordinate interval on the graph's x-axis. Default value: `Current Month`.

   Date Range Table

Field	Description
Current, Prior Week	Start day is either Sunday or Monday, depending on what is defined in your system. Reporting interval is set to days.

Field	Description
Current, Prior Month	Start day is first day of the month. Reporting interval is set to days.
Current, Prior Quarter	Quarters are bound by calendar year. Reporting interval is set to weeks.
Current, Prior Year	Reporting interval is set months.
Custom	Displays the **Custom** pop-up which enables you to pick an arbitrary start- and end-date.

The start-time is automatically set to 00:00 (24-hour clock) for the selected date in the reporting period.

The end-time is automatically set to 24:00 for the selected date.

2. Set the report filters.

   You can select from the following filters.

   Filters Table

Field	Description
Application	Only deployments executed by the selected applications appear in the report. To select applications: a. Click the **Application** button. b. To include an application in the report, click the corresponding check box. If a large number of applications are listed, type the first few letters of the application's name in the text box to scroll the list. Multiple applications can be selected. c. Click **OK**.
Status	Only deployments with the selected status appear in the report. Default value: `Success or Failure`, which means all deployments.
Plugin	Only deployments that used the selected plugin appear in the report. Default value: `Any`. **Note:** The `Any` value also includes deployments that did not use a plugin.

3. Run the report.

   Click the **Run** button to apply your filter conditions to the data and produce the report.

A line graph and tabular report are displayed. Each environment used by a reporting application is represented by an individual plot line and table row.

You can interact with the view to show or hide information, as follows:

- Hide/Show a plot line by clicking the corresponding item in the graph legend.
- See information about a graph coordinate by moving your pointer over the graph point.
- Zoom a graph area by dragging the pointer over the area. Click Reset Zoom to return to the normal view.

Sample Reports: Deployment Total Duration

The following table contains examples of reports that can be produced using the Deployment Total Duration report.

Sample Reports Table

Field	Description
Show me: Total duration times for two specific applications during the past ten days that used a particular plugin	• **Application**: Select both applications from the Applications dialog. • **Status**: `Success or Failure` • **Plugin**: Select the plugin from the drop-down list box. • **Date Range**: Use the Custom pop-up to set the ten-day range.

Field	Description
Show me: Total duration times for successful deployments for a given application during the past six months	• **Application**: Select the application from the **Applications** dialog. • **Status**: `Success` • **Time Unit**: `Months` • **Date Range**: Use the Custom pop-up to set the six-month range.

Deployment Count Report

The *Deployment Count Report* provides information about the number of deployments executed during a user-specified reporting period. The report provides both a tabular presentation and line graph of the data. Each table row represents an environment used by an applications for the reporting period and interval.

The line graph elements are:

- y-axis represents the number of deployments
- x-axis represents reporting intervals
- plot lines represent environments used by applications

The units along the y-axis are scaled to the number of records reported. The units along the x-axis represent the reporting interval, which can be: months, weeks, or days. Each color-coded plot line represents a single environment used by the deployment during the reporting period.

When selected, the report runs automatically for the default reporting period (current month)and reporting interval (days), and with all filters set to `Any`. The default report provides a count of all deployments that ran during the current month.

For details on configuring and running this report, see the following topics:

- Deployment Count fields [page 386]
- Running the Deployment Count Report [page 387]
- Report Samples: Deployment Count [page 388]

Deployment Count fields

The deployment count fields are shown in the following table.

Deployment Count Fields Table

Field	Description
Application	Name of the application that executed the deployment.
Environment	Name of the environment used by the application.
Reporting Interval	The remaining columns display the number of the deployments for the selected reporting interval.

Running the Deployment Count Report

To run a report:

1. Set the reporting period.

   Use the `Date Range` date-picker to set the report's start- and end-dates. The selected values determine the columns in the tabular report and the coordinate interval on the graph's x-axis. Default value: `Current Month`.

   Date Range

Field	Description
Current, Prior Week	Start day is either Sunday or Monday, depending on what is defined in your system. Reporting interval is set to days.
Current, Prior Month	Start day is first day of the month. Reporting interval is set to days.
Current, Prior Quarter	Quarters are bound by calendar year. Reporting interval is set to weeks.
Current, Prior Year	Reporting interval is set months.
Custom	Displays the Custom pop-up which enables you to pick an arbitrary start- and end-date.

   The start-time is automatically set to 00:00 (24-hour clock) for the selected date in the reporting period.

   The end-time is automatically set to 24:00 for the selected date.

2. Set the report filters.

   Filters are set with the properties drop-down list boxes. To set a filter, select it from the corresponding property list box.

   Filters Table

Field	Description
Application	Only deployments executed by the selected applications appear in the report. To select applications: a. Click **Application**. b. To include an application in the report, click the corresponding check box. If a large number of applications are listed, type the first few letters of the application's name in the text box to scroll the list. Multiple applications can be selected. c. Click **OK**.
Status	Only deployments with the selected status appear in the report. Default value: `Success or Failure`, which means all deployments.
Plugin	Only deployments that used the selected plugin appear in the report. Default value: `Any`. Note: The `Any` value also includes deployments that did not use a plugin.

3. Run the report.

   Click **Run** to apply your filter conditions to the data and produce the report.

A line graph and tabular report are displayed. Each environment used by a reporting application is represented by an individual plot line and table row.

You can interact with the view to show or hide information, as follows:

- Hide/Show a plot line by clicking the corresponding item in the graph legend.
- See information about a graph coordinate by moving your pointer over the graph point.
- Zoom a graph area by dragging the pointer over the area. Click Reset Zoom to return to the normal view.

Report Samples: Deployment Count

The following table contains examples of reports that can be produced using the Deployment Count Report.

Sample Reports Table

Field	Description
Show me: The number of successful deployments for two specific applications during the past ten days that used a particular plugin	• **Application**: Select both applications from the Applications dialog. • **Status**: `Success` • **Plugin**: Select the plugin from the drop-down list box. • **Date Range**: Use the **Custom** pop-up to set the ten-day range.
Show me: The number of failed deployments for a given application during the past month	• **Application**: Select the value from the Applications dialog. • **Status**: `Failure` • **Date Range**: Select `Prior Month`.
Show me: The number of failed deployments that used a specific plugin yesterday	• **Application**: Select the applications from the Applications dialog. • **Status**: `Failure` • **Plugin**: Select the value from the drop-down list box. • **Date Range**: Use the **Custom** pop-up to select the previous day.

Deployment Detail Report

The *Deployment Detail Report* provides information about deployments executed during a user-specified reporting period. Each report row represents a deployment that executed during the reporting period and matched the filter conditions.

When selected, the report runs automatically for the default reporting period, current month, and with all filters set to `Any`. The default report represents all deployments that ran during the current month.

For details on configuring and running this report, see the following topics:

- Deployment Detail Fields [page 390]
- Running the Deployment Detail Report [page 391]
- Report Samples: Deployment Detail [page 392]

Deployment Detail Fields

The deployment detail fields are shown in the following table.

Deployment Detail Fields Table

Field	Description
Application	Name of the application that executed the deployment.
Environment	Target environment of the deployment.
Date	Date and time when the deployment was executed.
User	Name of the user who performed the deployment.
Status	Final disposition of the deployment. Possible values are: • Success • Failure • Running • Scheduled • Approval Rejected • Awaiting Approval
Duration	Amount of time the deployment ran. For a successful deployment, the value represents the amount of time taken to complete successfully. If deployment failed to start, no value is given. If a deployment started but failed to complete, the value represents the amount of time it ran before it failed or was cancelled.
Action	This field provides links to additional information about the deployment. The `View Request` link displays the **Application Process Request** pane See Chapter 19: Managing Applications [page 263].

Running the Deployment Detail Report

To run a report:

1. Use the **Date Range** date-picker to set the report's start- and end-dates.

   Date Range Table

Field	Description
Current, Prior Week	Start day is either Sunday or Monday, depending on what is defined in your system.
Current, Prior Month	Start day is first day of the month.
Current, Prior Quarter	Quarters are bound by calendar year.
Current, Prior Year	Current year includes today's date.
Custom	Displays the **Custom** pop-up which enables you to pick an arbitrary start- and end-date.

   The start-time is automatically set to 00:00 (24-hour clock) for the selected date in the reporting period.

   The end-time is automatically set to 24:00 for the selected date.

2. Set the report filters.

   Filters are set with the properties drop-down list boxes. To set a filter, select it from the corresponding property list box.

   Report Filters Table

Field	Description
Application	Only deployments executed by the selected application appear in the report. Default value: `Any`.
Environment	Only deployments executed by the application selected with the **Application** list box that also used this environment appear in the report. If the application value is `Any`, the available value is `Any`; otherwise, environments defined for the selected application are listed.
User	Only deployments executed by the selected user appear in the report. Default value: `Any`.

Field	Description
Status	Only deployments with the selected status appear in the report. Default value: `Any`.
Plugin	Only deployments that used the selected plugin appear in the report. Default value: `Any`. Note: the `Any` value also includes deployments that did not use a plugin.

3. Run the report.

   Click the **Run** button to apply your filter conditions to the data and produce the report.

By default, the report is sorted by application. You can sort the report on any field by clicking on the column header.

Report Samples: Deployment Detail

The following table contains examples of reports that can be produced using the Deployment Detail Report.

Sample Reports Table

Field	Description
Show me: *All failed deployments that occurred on July 4th during the previous year.*	• **Application**: Any • **Status**: Failure • **Date Range**: Use the Custom pop-up to set the start- and end-dates to July 4th.
Show me: *Deployments for an application that used a specific environment.*	• **Application**: Select the value from the drop-down list box. • **Environment**: Select the environment from the drop-down list box. When an application is selected, only environments defined for it are available in the Environment drop-down list box.

Field	Description
Show me: *Failed deployments that used a specific plugin yesterday.*	• **Status**: Failure • **Plugin**: Select the value from the drop-down list box. • **Date Range**: Use the Custom pop-up to set the start- and end-dates to the previous day.
Show me: *My deployments that used a specific application during the past month.*	• **Application**: Select the value from the drop-down list box. • **User**: Select your user ID. • **Date Range**: Select `Prior Month`.

Saving and Printing Reports

You can save and print report data for all report types. In addition, you can save filter and sort order information for deployment-type reports.

For details, see the following topics:

- Saving Report Data [page 393]
- Saving Report Filters [page 393]
- Printing Reports [page 394]

Saving Report Data

Deployment Automation saves report data in CSV files (comma separated value).

To save report data:

1. Set the filters (if any) and run the report.
2. Click **CSV**.
3. Use the **Opening File** dialog. You can save the data to file, or open the data with an application associated with CSV-type files on your system.

Note: Sort-order and hidden/visible column information is not preserved in the CSV file.

Saving Report Filters

You can save filter and sort-order settings for deployment reports. Saved reports can be retrieved from the **Reports page > My Reports** menu.

To save a report:

1. Set the filter conditions.
2. Define the reporting period.
3. Run the report.
4. Optional. Set the sort order.

   You can change the sort order for any column by clicking the column header.
5. Optional. Change column visibility. Click **Edit** to display the Select Columns dialog. By default, all columns are selected to appear in a report. To hide a column, click the corresponding check box.
6. Click **Save**. The **Save Current Filters** dialog displays.
7. Enter a name for the file, and save your work.

To run your report, click the report name in the **My Reports** menu.

To delete your report, click **Delete**.

Printing Reports

To print a report:

1. Set the filter conditions.
2. Define the reporting period.
3. Run the report.
4. Optional. Set the sort order.

   Your changes are reflected in the printed report.
5. Optional. Change column visibility. By default, all columns are selected to appear in the printed report. Hidden columns will not appear in the output.
6. Click **Print** to print your report.

Glossary

Agents

Agents are physical resources for deployment. To run a deployment, an agent must be installed on the target server.

See Managing Agents and Agent Pools [page 310].

Agent Pools

An agent pool is a set of agents that is logically grouped together.

See Managing Agents and Agent Pools [page 310].

Agent Relays

Agent relays are used to manage communication between servers and agents. Agent relays are typically used when agents are dispersed across geographic locations or must communicate through firewalls. Agent relays can also be used to manage network traffic in implementations where you have many agents.

See Managing Agent Relays [page 316].

ALF Events

ALF (Application Lifecycle Framework) is an integrative layer for some Micro Focus products. These events can be used for custom integrations from SBM to Deployment Automation.

See ALF Events [page 155] and the *Deployment Automation User's Guide*.

Aliases

Aliases are alternate names that are assigned to agents and agent relays when configuring mutual authentication. See Configuring Mutual Authentication Mode [page 166].

Applications

Applications bring together components with their deployment targets and orchestrate multi-component deployments.

See Chapter 19: Managing Applications [page 263].

Application Environments

Application environments are environments that have been added to an application. Each application environment has its own set of operations and data within the context of the application, including inventory and compliance status. Environments can be added to multiple applications individually or as part of pipelines [page 399].

See Adding Environments to Applications [page 270].

Application Processes

Application processes are typically configured from processes defined for their associated components. See Application Processes [page 274].

Application Tasks

An application task is used to interrupt an application process until some manual intervention is performed. See Application Tasks [page 289].

Approvals

Approvals enable process requests to execute in application environments that require them. See Approvals [page 290].

Approval Processes

Approval processes tell Deployment Automation what to do when an application environment requires approval. See Approvals [page 290].

Artifacts

Files or other software elements associated with components. See Components [page 397].

Artifact Repository

See CodeStation [page 396].

Authentication Realms

Authentication realms determine a user's identity within an authorization realm. See Chapter 8: Security Administration [page 115].

Authorization Realms

Authorization realms associate users with roles and work with authentication realms to determine which users can access Deployment Automation. See Chapter 8: Security Administration [page 115].

Blackout

A blackout of a date means that no deployments can be scheduled and no snapshots can be made on that date. A blackout is set per environment, per application. See Setting Deployment Blackout Dates [page 368].

CodeStation

CodeStation handles artifact versioning in Deployment Automation. You can direct Deployment Automation to introduce artifacts into CodeStation from the file system or from external source control tools that you identify when you select the **Source Config Type** for a component. See Chapter 13: Artifact Repository [page 177].

Compliancy

An environment is compliant if the last requested processes were successfully executed in that environment. See Chapter 31: Inventory Management [page 377].

Components

Components typically map to a functional part of a real world application, and represent deployable items, or artifacts, such as files, images, databases, and configuration materials.

See Chapter 18: Managing Components [page 199].

Component Processes

Component processes define how the component should be deployed, installed or interacted with. A process typically contains a number of plugin steps that are inter-dependent, and may include complex logic for actions to perform if a step of the process fails.

See Component Processes [page 238].

Component Tasks

A component task is used to interrupt a component process until some manual intervention is performed. See Component Tasks [page 256].

Component Templates

Component templates enable you to save and reuse component processes and properties and create new components from them; template-based components inherit the template properties and processes. See Creating Configuration Templates [page 260].

Component Versions

Instances of a Deployment Automation component. See Component Versions [page 231].

Configuration Templates

Configuration templates enable you to save and reuse configuration data in processes. Typically, the data is for server configurations, for example on Tomcat servers, but the data can be for any purpose. See Configuration Templates [page 260].

Deployment

Deployment is a process request execution that results in component versions or snapshots being installed onto one or more target environments. See Chapter 29: Ways of Deploying [page 367].

Deployment Packages

Deployment packages enable you to deploy artifacts for multiple applications. They may also include component processes where components are shared among multiple applications and associated versions are to be deployed as part of the larger package.

See Chapter 23: Managing Deployment Packages [page 325].

Environments

Environments represent logical deployment locations. Your deployment processes must run in at least one environment. Environments and their resources are used by applications and components at runtime.

See Chapter 20: Managing Environments [page 297].

Export

You can export objects from a Deployment Automation server and subsequently import [page 398] them into another Deployment Automation server. Export is done on an object by object basis, whereas replication export can alternatively be used to export an entire application. See Replication [page 400].

Gates

Gates ensure that component versions or snapshots are deployed into environments only if their status meets certain conditions. See Gates [page 295].

Global Processes

Global processes are processes that are not associated with a particular application or component. See Chapter 24: Managing Global Processes [page 335].

Groups

See Authorization Realms [page 396].

Import

You can export [page 398] objects from a Deployment Automation server and subsequently import them into another Deployment Automation server. Import is done on an object by object basis, whereas replication import can alternatively be used to import an entire application. See Replication [page 400].

Inventory

The components, snapshots, and configurations deployed to any resource. See Chapter 31: Inventory Management [page 377].

Inventory Statuses

Inventory statuses can be applied to component versions when they are successfully deployed to resources. See Defining Statuses [page 110].

Locks

Deployment Automation uses locks to ensure that processes do not interfere with one another. See Locks [page 108].

Network Relays

Network relays can be used in multiple-server configurations to manage the communication between servers, and are used in high availability, active-active server implementations. For configuring server to agent communication for either agent- or server-initiated communication of HTTP communication, you should use agent relays [page 395]. See Managing Network Relays [page 319].

Notification Schemes

Notification schemes define what events should generate notifications and who should get them. See Configuring Email Notifications [page 144].

Notification Templates

Notification templates define the format of the information that you want to send in the email notifications for given situations. See Configuring Email Notifications [page 144].

Mutual Authentication

In mutual authentication mode, communications are encrypted as usual, but users are also required to authenticate themselves by providing digital certificates. See Configuring Mutual Authentication Mode [page 166].

Output Log

The output log is the Deployment Automation server log. See Output Log [page 151].

Pipelines

A pipeline is a pre-defined sequence of environments in which application process requests are executed.

See Chapter 21: Managing Pipelines [page 303] .

Plugins

Plugins provide functionality in discrete steps to be used in component and global processes for configuration of or deployment into target environments.

See Plugins [page 244].

Processes

See Component Processes [page 397], Application Processes [page 396], Global Processes [page 398], and Approval Processes [page 396].

Properties

Properties are parameters that are referenced in process steps. See Chapter 25: Properties [page 341].

Replication

Replication enables you to export and import whole sets of Deployment Automation objects. See Chapter 27: Replication [page 351].

Resources

Resources represent a deployment target on a Deployment Automation environment. Examples include physical machines, virtual machines, databases, or J2EE containers.

See Resources [page 308].

Resource Groups

A resource group is a logical collection of resources. Resource groups help you organize and manage the agents installed in different environments throughout the network. See Resource Groups [page 319].

Roles

Roles provide the building blocks for the security system. Roles have permissions that define the actions their members, users or groups, can perform with product features. See Role Configuration [page 116] and Setting Role Membership by Product Area [page 138].

Schedule

The schedule option enables you to schedule application processes and blackout dates for application environments. See Scheduling Deployments [page 368].

SSL (Secure Socket Layer)

SSL (Secure Socket Layer) Secure Socket Layer (SSL) technology enables clients and servers to communicate securely by encrypting all communications. See Chapter 11: Secure Socket Layer (SSL) Configuration [page 165].

Single Sign-On (SSO)

Single Sign-On (SSO) enables Deployment Automation to integrate more easily with other Micro Focus products. Login information is passed automatically through SSO so that there is no need to prompt for login credentials as information flows between products. See Creating Authentication Realms [page 124].

Snapshots

A snapshot captures an application's current state, and as the application moves through different environments, the snapshot ensures that proper component versions are used. See Snapshots [page 292].

Snapshot Statuses

You can apply and enforce conditions on application gates based on snapshot statuses. See Defining Statuses [page 110].

Source Config Type

An integration from which you can load artifacts into Deployment Automation as component versions. See Source Configuration Type Fields [page 203].

Statuses

See Inventory Statuses [page 398], Snapshot Statuses [page 401], and Version Statuses [page 401].

Tasks

See Application Tasks [page 396] and Component Tasks [page 397].

Timelines

Timelines show the number of processes run and graphical indicators of success or failure. See Viewing Timelines [page 371].

Tokens

Tokens provide authorization for agents and users. See Tokens [page 138].

Users

See Authorization Realms [page 396].

Versions

See Component Versions [page 397].

Version Statuses

You can apply and enforce conditions on application gates based on component version statuses. See Defining Statuses [page 110].