IBM uDeploy® Administration Guide

5.0.0

IBM uDeploy Administration Guide: 5.0.0

Publication date January 2013

Document Number: 5.0.0.1a

About This Book

This book describes how to use UrbanCode, an IBM Company's IBM uDeploy product and is intended for all users.

Product Documentation

IBM uDeploy's online Help is installed along with the product software and can be accessed from the product's web-based user interface. Product books are available in PDF format at UrbanCode, an IBM Company's Documentation portal: http://docs.urbancode.com/, and are included with the installation package.

In addition to this book, the books comprising the documentation suite include:

Table 1. Product Documentation Suite

Book	Description
IBM uDeploy User Guide	Describes how to use the product; contains chapters for core features, such as components, applications, and resources.
IBM uDeploy Administration Guide	Describes how to install and configure the product, and how to set up security.
IBM uDeploy Introduction	Provides an overview of the product's significant features and describes its architecture.

Product Support

The UrbanCode, an IBM Company Support portal, http://support.urbancode.com/, provides information that can address any of your questions about the product. The portal enables you to:

- review product FAQs
- download patches
- view release notes that contain last-minute product information
- review product availability and compatibility information
- access white papers and product demonstrations

Document Conventions

This book uses the following special conventions:.

- `Program listings, code fragments, and literal examples are presented in this typeface.`
- Product navigation instructions are provided like this:

  `Home > Components > [selected component] > Versions > [selected version] > Add a Status [button]`

This example, which explains how to add a status to a component version, means: from the IBM uDeploy home page click the Components tab (which displays the Components pane); select a component (which displays a pane with information for the selected component); click the Versions tab (which displays a pane with information about the selected version); and click the Add a Status button.

- User interface objects, such as field and button names, are displayed with initial Capital Letters.
- Variable text in path names or user interface objects is displayed in *italic* text.
- Information you are supposed to enter is `displayed in this format`.

Installing and Upgrading Servers and Agents

A IBM uDeploy installation consists of the IBM uDeploy server (with a supporting database), and at least one agent. Typically, the server, database, and agents are installed on separate machines, although for a simple evaluation they can all be installed on the same machine. In addition, Java must be installed on all agent and server machines.

> ## Note
>
> For evaluation purposes, the supplied Derby database should be adequate and can be installed on the machine where the server is located. If you are installing IBM uDeploy in a production environment, UrbanCode, an IBM Company recommends the use one of the supported databases--Oracle Database (all versions), SQL Server, or MySQL.

Installation Steps

1. Review the system requirements. See the section called “System Requirements”. Requirements and recommendations, including performance recommendations, are provided.

2. Ensure that Java is installed on the server and agent machines (and agent relay machine if used). All machines require Java JRE 5 or greater. Set the JAVA_HOME environment variable to point to the directory you intend to use. A JDK can be used.

3. Download the server, agent, agent relay, and CLI client (command line interface) installation packages. Installation files can be downloaded from the UrbanCode support portal http://support.urbancode.com. If you are installing an evaluation version, the license is included with the downloaded files. For evaluations, the agent relay (used to communicate with remote networks) and the CLI client can be skipped. At a minimum, an installation must have the server, a database, and at least one agent.

4. If you are installing an agent relay, download the agent relay installation files as well.

5. If you are not installing an evaluation version, install one of the supported databases. The database should be installed before the server and on a separate machine. See the section called “Database Installation”

6. Complete database installation by configuring the appropriate JDBC driver (typically supplied by the database vendor).

7. Create an empty database for IBM uDeploy and at least one dedicated user account.

8. Install the server. You will need to supply values for the IP address, ports for HTTP communication (secured and unsecured), port for agent communication, and URL. The installation program provides default values for many parameters. The properties set during installation are recorded in the `installed.properties` file located in the `server_install/conf/server/` directory. If you intend to turn on SSL, see the section called “SSL Configuration” See the section called “Server Installation”.

9. If you are using an agent relay, install the relay. See the section called “Installing Agent Relays”.

10. Finally, install at least one agent. Agents are installed on target machines and communicate with the server. When installing an agent, you supply several values defined during server installation. See the section called “Agent Installation” for instructions about installing agents. An agent requires various

access privileges for the machine where it is installed, which are described in that section. To determine if the agent is in communication with the server, display the web application's Resource pane. A value of *Online* in the agent's Status field means the agent is successfully connected.

For information about using the CLI (command line interface, see ???.

For information about running the installed items and accessing the IBM uDeploy web application, see the section called “Running IBM uDeploy”.

Installation Recommendations

Because the IBM uDeploy agent performs most of the deployment processing, agent installation is critical for good performance. Except for evaluation purposes, an agent should never be installed on the same machine as the server. In addition, many IBM uDeploy users have found that by following some general guidelines they are able to reduce the chances of performance-related issues:

- **Install the server as a user account.** The server should be installed as a dedicated system account whenever possible. While not recommended, IBM uDeploy can run as the root user (or local system user on Windows) and running in this manner avoids all permission errors.
- **Install each agent as a dedicated system account.** Ideally, the account should only be used by IBM uDeploy. Because IBM uDeploy 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 effect the host file system.
- **Except for evaluation purposes, do not install an agent on the IBM uDeploy server machine.** Because the agent is resource intensive, installing one on the server machine can degrade performance.
- **Install a single agent per host machine.** Multiple agents on the same machine can negatively impact each other's performance. When you must install multiple agents, you might see performance degradation when multiple agents are busy simultaneously.

System Requirements

IBM uDeploy will run on Windows and Unix-based systems. While the minimum requirements provided below are sufficient for an evaluation, you will want server-class machines for production deployments.

Server Minimum Installation Requirements

- Windows: Windows 2000 Server (SP4) or later.
- Processor: Single core, 1.5 GHz or better.
- Disk Space: 300 MB or more.
- Memory: 2 GB, with 256 MB available to IBM uDeploy.
- Java version: JRE 5 or greater.

Recommended Server Installation

- **Two server-class machines**

UrbanCode, an IBM Company recommends two machines for the server: a primary machine and a standby for fail-over. In addition, the database should be hosted on a separate machine.

- **Separate machine for the database**
- **Processor** 2 CPUs, 2+ cores for each.
- **RAM** 8 GB
- **Storage** Individual requirements depend on usage, retention policies, and application types. In general, the larger number of artifacts kept in IBM uDeploy's artifact repository (CodeStation), the more storage needed.

  ### Note

  CodeStation is installed when the IBM uDeploy 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 UrbanCode, an IBM Company support.

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

Agent Minimum Requirements

Designed to be minimally intrusive (typically, an idle agent uses 5Mz CPU), agents require 64-256 MB of memory and 100 MB of disk space. Additional requirements are determined by the processes the agent will run. For a simple evaluation, the agent can be installed on the same physical machine as the server. In production environments, agents should be installed on separate machines.

32- and 64-bit JVM Support

The IBM uDeploy server must use the 32-bit JDK for the Windows 2003 64-bit server; the 64-bit JDK can be used for agents. Because IBM uDeploy does not require a multi-gigabyte heap, there is little advantage to using a 64-bit JVM. For 64-bit Windows installations, IBM uDeploy uses a 32-bit JVM; for other 64-bit platforms, IBM uDeploy uses a 64-bit JVM, as the following table illustrates:

Table 2. JVM Support

Operating System	Operating System	JVM 64-bit
Windows 32-bit	yes	NA
Windows 64-bit	yes	yes

Operating System	Operating System	JVM 64-bit
Non-Windows 32-bit	yes	NA
Non-Windows 64-bit	yes	yes

Performance Recommendations

Since the IBM uDeploy agent performs most of the processing, optimal agent configuration is important. Except when evaluating IBM uDeploy, an agent should not be installed on the same machine where the server is located.

By following these recommendations, you should avoid most performance-related issues:

- **Install the server as a dedicated user account.** The server should be installed as a dedicated system account whenever possible. However, IBM uDeploy runs well as a root user (or local system user on Windows), and running this way is the easiest method to avoid permission errors.
- **Install the agent as dedicated system account.** Ideally, the account used should be dedicated to IBM uDeploy. Because IBM uDeploy agents are remote command-execution engines, it is best to create a user just for the agent and grant it only the appropriate privileges.
- **Do not install an agent on the IBM uDeploy server machine.** Because the agent is resource intensive, installing one on the server machine will degrade server performance whenever a large deployment runs.
- **Install one agent per machine.** Several agents on the same machine can result in significant performance reduction, especially when they are running at the same time.

Download IBM uDeploy

The installation package is available from the UrbanCode, an IBM Company support portal--Supportal. If you are evaluating IBM uDeploy, the Supportal account where you download IBM uDeploy also enables you to create support tickets.

1. Navigate to the UrbanCode, an IBM Company Support Portal *`support.UrbanCode, an IBM Company.com/tasks/login/LoginTasks/login`*. If you do not have an account, please create one.

   ### Note

   You must have a license in order to download the product. For an evaluation license, go to *`UrbanCode, an IBM Company.com/html/products/deploy/default.html`*.

2. Click the **Products** tab and select the IBM uDeploy version you want to download.
3. Select the appropriate package for your environment for the server, agent, command line client, and agent relay. The contents of the zip and tar packages are the same.

   IBM uDeploy enables you to install agents on any supported platform, regardless of the operating system where the server is installed.

4. Download the license. If you do not see a license, ensure that you are the Supportal account administrator. Licenses are not available to all Supportal users.

Database Installation

Currently, IBM uDeploy supports Derby, Oracle, SQL Server, and MySQL.

Installing Oracle

Before installing the IBM uDeploy server, install an Oracle database. If you are evaluating IBM uDeploy, you can install the database on the same machine where the IBM uDeploy server will be installed.

When you install IBM uDeploy, you will need the Oracle connection information, and a user account with table creation privileges.

IBM uDeploy supports the following editions:

- Oracle Database Enterprise
- Oracle Database Standard
- Oracle Database Standard One
- Oracle Database Express

Version 10g or later is supported for each edition.

To install the database

1. Obtain the Oracle JDBC driver. The JDBC jar file is included among the Oracle installation files. The driver is unique to the edition you are using.
2. Copy the JDBC jar file to *IBM uDeploy_installer_directory*\lib\ext.
3. Begin server installation, see the section called “Server Installation”. When you are prompted for the database type, enter `oracle`.
4. Provide the JDBC driver class IBM uDeploy will use to connect to the database.

   The default value is `oracle.jdbc.driver.OracleDriver`.
5. Provide the JDBC connection string. The format depends on the JDBC driver. Typically, it is similar to:

   `jdbc:oracle:thin:@[DB_URL]:[DB_PORT]`

   For example:

   `jdbc:oracle:thin:@localhost:1521`.
6. Finish by entering the database user name and password.

   ### Note

   The schema name must be the same as the user name.

Installing MySQL

Before installing the IBM uDeploy server, install MySQL. If you are evaluating IBM uDeploy, you can install the database on the same machine where the IBM uDeploy server will be installed.

When you install IBM uDeploy, you will need the MySQL connection information, and a user account with table creation privileges.

To install the database

1. Create a database:

   ```
   CREATE DATABASE IBM uDeploy;

   GRANT ALL ON IBM uDeploy * TO 'IBM uDeploy'@'%'

   IDENTIFIED BY 'password' WITH GRANT OPTION;
   ```

2. Obtain the MySQL JDBC driver. The JDBC jar file is included among the installation files. The driver is unique to the edition you are using.

3. Copy the JDBC jar file to *IBM uDeploy_installer_directory*\lib\ext.

4. Begin server installation, see the section called “Server Installation”. When you are prompted for the database type, enter `mysql`.

5. Provide the JDBC driver class IBM uDeploy will use to connect to the database.

   The default value is `com.mysql.Driver`.

6. Next, provide the JDBC connection string. Typically, it is similar to:

   ```
   jdbc:mysql[DB_URL]:[DB_PORT]:[DB_NAME]
   ```

   For example:

   `jdbc:mysql://localhost:3306/IBM uDeploy`.

7. Finish by entering the database user name and password.

Installing Microsoft SQL Server

Before installing the IBM uDeploy server, install a SQL Server database. If you are evaluating IBM uDeploy, you can install the database on the same machine where the IBM uDeploy server will be installed.

When you install IBM uDeploy, you will need the SQL Server connection information, and a user account with table creation privileges.

Before installing the IBM uDeploy server, install an SQL Server database. If you are evaluating IBM uDeploy, you can install the database on the same machine where the IBM uDeploy server will be installed:

```
CREATE DATABASE IBM uDeploy;

USE IBM uDeploy;

CREATE LOGIN IBM uDeploy WITH PASSWORD = 'password';

CREATE USER IBM uDeploy FOR LOGIN IBM uDeploy WITH DEFAULT_SCHEMA = IBM uDeploy;

CREATE SCHEMA IBM uDeploy AUTHORIZATION IBM uDeploy;
```

```
GRANT ALL TO IBM uDeploy;
```

1. Obtain the SQL Server JDBC driver from the Microsoft site. The JDBC jar file is not included among the installation files.
2. Copy the JDBC jar file to *IBM uDeploy_installer_directory*\lib\ext.
3. Begin server installation, see the section called “Server Installation”. When you are prompted for the database type, enter *`sqlserver`*.
4. Provide the JDBC driver class IBM uDeploy will use to connect to the database.

   The default value is *`com.microsoft.sqlserver.jdbc.SQLServerDriver`*.
5. Next, provide the JDBC connection string. The format depends on the JDBC driver. Typically, it is similar to:

   `jdbc:sqlserver://[DB_URL]:[DB_PORT];databaseName=[DB_NAME]`

   For example:

   `jdbc:sqlserver://localhost:1433;databaseName=IBM uDeploy`.
6. Finish by entering the database user name and password.

Server Installation

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. The properties set during installation are recorded in the `installed.properties` file located in the `server_install/conf/server/` directory.

If the following steps fail, contact UrbanCode support and provide the log from standard out put.

Note

If you are installing the server in a production environment, install and configure the database you intend to use before installing the server. See the section called “Database Installation”.

Windows Server Installation

1. Download and unpack the installation files to the *installer_directory*.
2. From the *installer_directory*, run *`install-server.bat`*.

   ### Note

   Depending on your Windows version, you might need to run the batch file as the administrator.

   The IBM uDeploy Installer is displayed and prompts you to provide the following information:
3. **Enter the directory where the IBM uDeploy Server will be installed.** Enter the directory where you want the server located. If the directory does not exist, enter *`Y`* to instruct the Installer to create it for you. If you enter an existing directory, the program will give you the option to upgrade the server. For information about upgrading, see the section called “Upgrading IBM uDeploy”.

Note

Any default values suggested by the program (displayed within brackets) can be accepted by simply pressing **Enter**. If two options are given, such as `Y/n`, the capitalized option is the default value.

4. **Please enter the home directory of the JRE/JDK used to run the server.**

   If Java has been previously installed, IBM uDeploy will suggest the Java location as the default value. To accept the default value, press *`ENTER`*, otherwise override the default value and enter the correct path.

5. **Enter the IP address on which the Web UI should listen.** UrbanCode, an IBM Company suggests accepting the default value *`all available to this machine`*.

6. **Do you want the Web UI to always use secure connections using SSL?**

   Default value is *`Y`*.

   If you use SSL, turn it on for agents too, or the agents will not be able to connect to the server. This also applies if using mutual authentication. If you change the port numbers for agent communication, you need to provide the port numbers when installing the agents.

   This sets the `install.server.web.always.secure=` property in the `installed.properties` file.

7. **Enter the port where IBM uDeploy should listen for secure HTTPS requests.** The default value is *`8443`*.

   This sets the `install.server.web.ip=` property in the `installed.properties` file.

8. **Enter the port on which the IBM uDeploy server should redirect unsecured HTTP requests.**

   The default value is *`8080`*.

9. **Enter the URL for external access to the web UI**.

10. **Enter the port to use for agent communication**.

    The default value is *`7918`*.

11. **Do you want the Server and Agent communication to require mutual authentication?**

    If you select *`Y`*, a manual key must be exchanged between the server and each agent. The default value is *`N`*.

    This sets the `server.jms.mutualAuth=` property in the `installed.properties` file.

12. **Enter the database type IBM uDeploy should use.**

    The default value is the supplied database *`Derby`*. The other supported databases are: *`mysql, oracle, and sqlserver`*.

    If you enter a value other than *`derby`*, the IBM uDeploy Installer will prompt you for connection information, which was defined when you installed the database. See the section called “Database Installation”.

13. **Enter the database user name.**. The default value is *IBM uDeploy*. Enter the user name you created during database installation.

14. **Enter the database password.**. The default value is *password*.

15. **Do you want to install the Server as Windows service?**. The default value is *N*.

    When installed as a service, IBM uDeploy only captures the value for the PATH variable. Values captured during installation will always be used, even if you make changes later. For recent Windows versions, you will need to execute the command as Administrator.

    ### Note

    If you install the server as a service, the user account must have the following privileges:

    - SE_INCREASE_QUOTA_NAME "Adjust memory quotas for a process"
    - SE_ASSIGNPRIMARYTOKEN_NAME "Replace a process level token"
    - SE_INTERACTIVE_LOGON_NAME "Logon locally"

    The LOCAL SYSTEM account is on every Windows machine and automatically has these privileges. You might want to use it as it requires minimal configuration.

Unix/Linux Installation

1. Download and unpack the installation files to the *installer_directory*.

   ### Note

   If you are installing IBM uDeploy on Solaris, UrbanCode, an IBM Company recommends the Korn shell (ksh).

2. From the *installer_directory* run *install-server.sh*. The IBM uDeploy Installer is displayed and prompts you to provide the following information:

3. **Enter the directory where the IBM uDeploy Server will be installed.** If the directory does not exist, enter *Y* to instruct the Installer to create it for you. The default value is Y.

   ### Note

   Whenever the IBM uDeploy Installer suggests a default value, you can press *ENTER* to accept the value.

4. **Please enter the home directory of the JRE/JDK used to run the server.**

   If Java has been previously installed, IBM uDeploy will suggest the Java location as the default value. To accept the default value, press *ENTER*, otherwise override the default value and enter the correct path.

5. **Enter the IP address on which the Web UI should listen.** UrbanCode, an IBM Company suggests accepting the default value *all available to this machine*.

6. **Do you want the Web UI to always use secure connections using SSL?**

   Default value is *Y*.

If you use SSL, turn it on for agents too, or the agents will not be able to connect to the server. This also applies if using mutual authentication. If you change the port numbers for agent communication, you need to provide the port numbers when installing the agents.

This sets the `install.server.web.always.secure=` property in the `installed.properties` file.

7. **Enter the port where IBM uDeploy should listen for secure HTTPS requests.** The default value is *`8443`*.

   This sets the `install.server.web.ip=` property in the `installed.properties` file.

8. **Enter the port on which the IBM uDeploy should redirect unsecured HTTP requests.**

   The default value is *`8080`*.

9. **Enter the URL for external access to the web UI**.

10. **Enter the port to use for agent communication**.

    The default value is *`7918`*.

11. **Do you want the Server and Agent communication to require mutual authentication?**

    If you select *`Y`*, a manual key must be exchanged between the server and each agent. The default value is *`N`*.

    This sets the `server.jms.mutualAuth=` property in the `installed.properties` file.

12. **Enter the database type IBM uDeploy should use.**

    The default value is the supplied database *`Derby`*. The other supported databases are: *`mysql, oracle, and sqlserver`*.

    If you enter a value other than *`derby`*, the IBM uDeploy Installer will prompt you for connection information, which was defined when you installed the database. See the section called “Database Installation”.

13. **Enter the database user name.**. The default value is *`IBM uDeploy`*. Enter the user name you created when you installed the database.

14. **Enter the database password.**. The default value is *`password`*.

Agent Installation

For production environments, UrbanCode, an IBM Company recommends creating 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. But if you plan to run deployments on several machines, a separate agent 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.

Each agent needs the appropriate rights to communicate with the IBM uDeploy server (if the agent will communicate with IBM uDeploy via an agent relay, see the section called “Connecting Agents to Agent Relays”).

At a minimum, each agent should have permission to:

- **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 IBM uDeploy user interface in order to download artifacts from the CodeStation repository.
- **Access the file system**. Many agents need read/write permissions to items on the file system.

Installing an Agent

After downloading and expanding the installation package, open the *installer_directory*.

From the *installer_directory* run `install-agent.bat` (Windows) or `install-agent.sh` (Unix-Linux).

Note

If you install the agent as a Windows service, the user account must have the following privileges:

- SE_INCREASE_QUOTA_NAME "Adjust memory quotas for a process"
- SE_ASSIGNPRIMARYTOKEN_NAME "Replace a process level token"
- SE_INTERACTIVE_LOGON_NAME "Logon locally"

The LOCAL SYSTEM account is on every Windows machine and automatically has these privileges. You might want to use it as it requires minimal configuration.

The IBM uDeploy Installer is displayed and prompts you to provide the following information. Any default values suggested by the program (displayed within brackets) can be accepted by simply pressing **Enter**. If two options are given, such as `Y/n`, the capitalized option is the default value.

1. **Enter the directory where agent should be installed.**. For example: C:\Program Files\urban-deploy \agent (Windows) or /opt/urban-deploy/agent (Unix). If the directory does not exist, enter `Y` to instruct the Installer to create it for you. If you enter an existing directory, the program will give you the option to upgrade the agent. For information about upgrading, see the section called “Upgrading IBM uDeploy”.

   ### Note

   Any default values suggested by the program (displayed within brackets) can be accepted by simply pressing **Enter**. If two options are given, such as `Y/n`, the capitalized option is the default value.

2. **Please enter the home directory of the JRE/JDK used to run the agent.** If Java has been previously installed, IBM uDeploy will suggest the Java location as the default value. To accept the default value, press `ENTER`, otherwise override the default value and enter the correct path.
3. **Will the agent connect to a agent relay instead of directly to the server?** The default value is `N`. If the agent will connect to an agent relay, see the section called “Connecting Agents to Agent Relays”.
4. **Enter the host name or address of the server the agent will connect to.** The default value is `localhost`.

5. **Enter the agent communication port for the server**. The default value is *7918*.

6. **Does the server agent communication use mutual authentication with SSL?**. Default value is *Y*.

   If you use SSL, turn it on for server too or the agent will not be able to connect to the server. This also applies if using mutual authentication. If you change the port numbers for agent communication, you need to provide them when installing the agents.

7. **Enter the name for this Agent.** Enter a unique name; the name will be used by IBM uDeploy to identify this agent. Names are limited to 256 characters and cannot be changed once connected.

8. **Do you want to install the Agent as Windows service?** (Windows only). The default value is *N*. When installed as a service, IBM uDeploy only captures the value for the PATH variable. Values captured during installation will always be used, even if you make changes later. For recent Windows versions, you will need to execute the command as Administrator.

Agents that will run on Unix machines can also be installed directly from the IBM uDeploy web application, see ???

Note

If the agent is configured properly, IBM uDeploy will recognize it automatically—you do not need to perform further actions in order to start using it.

Connecting Agents to Agent Relays

Remote agents--agents that will communicate with the server via an agent relay--are installed in much the same way local agents are installed (see the section called “Agent Installation”): you run the install script, *install-agent.bat*, and supply agent configuration information as described above, along with the relay-specific parameters.

When you answer *Yes* when asked if you want to connect the agent to a agent relay, you will be prompted to configure the following parameters:

Table 3. Agent-Agent Relay Connection

Parameter	Description
hostname or address of the agent relay the agent will connect to	Enter the host name or IP address of the agent relay. Supply the value you used when you installed the agent relay.
agent communication port for the agent relay	Enter the port which the agent will use for JMS-based communications with agent relay. The default value is 7916.
HTTP proxy port for the agent relay	Enter the port on which the agent will use for HTTP communications with the agent relay. The default value is 20080.

Installing Agent Relays

An agent relay is a communication proxy for agents that are located behind a firewall or in another network location. As long as there is at least a low bandwidth WAN connection between the server and remote agents, the IBM uDeploy server can send work to agents located in other geographic locations via the

relay. An agent relay requires that only a single machine in the remote network contact the server. Other remote agents communicate with the server by using the agent relay. All agent-server communication from the remote network goes through the relay.

You can download the agent relay installation package from the UrbanCode, an IBM Company support portal--Supportal. Before installing, ensure that:

- Java 1.5.0 or later is installed.
- The server with which the relay will connect is already installed.
- The user account and password created during server installation is available.

To install an agent relay:

1. Expand the compressed installation file.
2. From within the expanded `agent-relay-install` directory run the `install.cmd` script.
3. The installation program will prompt you for the following information. Any default values suggested by the program (displayed within brackets) can be accepted by simply pressing **Enter**. If two options are given, such as `Y/n`, the capitalized option is the default value.

Table 4. Agent Relay Configuration

Parameter	Description
Directory where you would like to install the agent relay	Enter the directory where you want the agent relay installed. If you enter an existing directory, the program will prompt you to upgrade the relay. For information about upgrading, see the section called “Upgrading IBM uDeploy”.
Java home	Directory where Java is installed. Ensure that the JAVA_HOME environment variable points to this directory.
Name of this relay	Enter the name of the agent relay. Each relay must have a unique name. The default name is `agent-relay`.
IP or hostname which this agent relay should use	Enter the IP or hostname on which the relay will listen.
Port which this agent relay should proxy HTTP requests on	Enter the port on which the agent relay should listen for HTTP requests coming from agents. The default value is 20080.
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 value is 7916.
Connect the agent relay to a central server?	Specify whether you want the relay to connect to the IBM uDeploy server.
IP or hostname of your central server	If you indicated that you want to connect the relay to the server, enter the IP or host name where the relay can contact the server.
Port which the central server uses for communication	If you indicated that you want to connect the relay to the server, enter the port the server uses

Parameter	Description
	to communicate with agents. The default value is 7918.
Use secure communication between the agent, relay and server?	Specify whether you want to use SSL security for communications between server, relay, and remote agents. The default value is `Y`. **Important** To use the relay, you must answer yes. Answering yes activates SSL security for HTTP- and JMS-based communications. If you answer no, the relay *will not* be able to communicate with the server (which uses JMS for most communications).
Use mutual authentication between the agent, relay and server.	If mutual authentication is required, enter `Y`. See the section called “SSL Configuration” for information about activating mutual authentication.
Install the Agent Relay as Windows service?	If you are installing the relay on Windows, you can install it as a Windows service. The default value is `N`.

If you need to modify the relay, you can edit these properties in the `agentrelay.properties` file located in the `relay_installation\conf` directory.

Upgrading IBM uDeploy

You upgrade the IBM uDeploy server, agents, and agent relays independently. Before upgrading, download the appropriate installation package from the UrbanCode, an IBM Company support portal (upgrades for servers and agent relays are done with the same package used for installation), and uncompress it.

1. Run the installation script for the server or agent relay you want to upgrade. To upgrade the server, for example, run the *`install-server`* script.

2. When prompted for the location of the installation directory, enter the path to an existing installation. When you specify an existing installation, IBM uDeploy will ask if you want to upgrade the installation (instead of installing a new version). If you answer Yes, the script will lead you through the required steps. The upgrade steps are a subset of the installation steps. If you need information about the steps, see the section related to the item you are upgrading.

Upgrading Agents

Agents can be upgraded individually on in batch. To upgrade an agent, display the Agents pane (*`Resources > Agents`*), then use the Upgrade action. To upgrade multiple agents, select the agents and use the Upgrade Selected button.

SSL Configuration

SSL (Secure Socket Layer) technology enables clients and servers to communicate securely by encrypting all communications. Data are encrypted before being sent and decrypted by the recipient--communications cannot be deciphered or modified by third-parties.

IBM uDeploy enables the server to communicate with its agents using SSL in two modes: unauthenticated and mutual authentication. In unauthenticated mode, communication is encrypted but users do not have to authenticate or verify their credentials. IBM uDeploy 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, as explained below.

Important

IBM uDeploy automatically uses SSL in unauthenticated mode for JMS-based communications between the server and agents (JMS is IBM uDeploy's primary communication method). 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.

In mutual authentication mode, the server, local agents, and agent relays each provide a digital certificate to one another. A digital certificate is a cryptographically signed document intended to assure others about the identity of the certificate's owner. IBM uDeploy certificates are self-signed. When mutual authentication mode is active, IBM uDeploy uses it for JMS-based server, local agents, and agent relay communication.

To activate this mode, the IBM uDeploy 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 it will use to communicate with the server. This mode can be implemented during installation or activated afterward, as explained below

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 all must use it.

Configuring SSL Unauthenticated Mode for HTTP Communications

To activate unauthenticated mode for HTTP:

1. Open the `installed.properties` file which is located in the `server_install/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.

Note

To complete unauthenticated mode for HTTP, contact UrbanCode, an IBM Company Support.

Configuring Mutual Authentication

To use mutual authentication, the server and agents must exchange keys. You export the server key (as a certificate) and import it into the agent keystore, then reverse the process by exporting the agent key and importing it into the server keystore. When using an agent relay, the relay must swap certificates with the server and with the remote agents that will use the relay.

Before exchanging keys, ensure that the following properties are set:

1. The `server.jms.mutualAuth` property in the server's `installed.properties` file (located in the `server_install/conf/server` directory) is set to *true*.
2. For each agent, the `locked/agent.mutual_auth` property in the agent's `installed.properties` file (located in the `agent_install\conf\agent` directory) is set to *true*.
3. For each agent relay, the `agentrelay.jms_proxy.secure` property in the relay's `agentrelay.properties` file (located in the `relay_install\conf` directory) is set to *true*.
4. For each agent relay, the `agentrelay.jms_proxy.mutualAuth` property in the relay's `agentrelay.properties` file is set to *true*.

To exchange keys:

1. Open a shell and navigate to the server installation `conf` directory.
2. Run:

   ```
   keytool -export -keystore server.keystore -storepass changeit
       -alias server -file server.crt
   ```

3. Copy the exported file (certificate) to the local agent/agent relay installation `conf` directory.
4. Import the file by running from within the agent's `conf` directory (or agent relay's `jms-relay` directory):

   ```
   keytool -import -keystore ud.keystore -storepass changeit
       -alias server -file server.crt -keypass changeit -noprompt
   ```

   You should see the `Certificate was added to keystore` message.

   ### Note

   For agent relays, replace `ud.keystore` with the name of the relay's keystore--`agentrelay.keystore`

5. For each local agent or agent relay , export the key by running the following (change the name of the file argument to match the agent name):

   ```
   keytool -export -keystore ud.keystore -storepass changeit
   ```

```
-alias ud_agent -file [agent_name].crt
```

You should see the `Certificate stored in file (agent_name.crt)` message.

Note

For agent relays, replace `ud.keystore` with the name of the relay's keystore--`agentrelay.keystore`

6. Copy the exported file to the server's `conf` directory.
7. From within the server's `conf` directory, import each certificate by running the following command (change the name of the file argument and alias to match the certificate's name):

```
keytool -import -keystore ud.keystore -storepass changeit
    -alias [agent_name] -file [agent_name].crt -keypass changeit -noprompt
```

You should see the `Certificate was added to keystore` message.

8. Restart the server and agents/agent relays.

To connect an agent relay with the remote agents that will use it, swap certificates as explained above: each remote agent must import the certificate for the relay it will use, and the relay must import the certificate from each remote agent that will use it. Agents using relays do not have to swap certificates with the server.

To list the certificates loaded into a keystore, run the following from within the keystore directory:

```
keytool -list -keystore ud.keystore -storepass changeit
```

Running IBM uDeploy

Both Unix- and Windows-based installations require the IBM uDeploy server and at lest one agent. If you are using a Oracle or MySQL database, make sure you have installed and configured the appropriate driver, see the section called “Database Installation”.

Running the Server

1. Navigate to the *server_installation*\bin directory
2. Run the *`run_server.cmd`* batch file (Windows), or *`start_server.cmd`* (Unix/Linux).

Running an Agent

After the server has successfully started:

1. Navigate to the *agent_installation*\bin directory
2. Run the *`run_udagent.cmd`* batch file (Windows), or *`start_udagent.cmd`* (Unix/Linux).
3. Once the agent has started, navigate to the IBM uDeploy web application and display the **Resources** tab. If installation went well, the agent should be listed with a status of `Online`.

Running an Agent Relay

After the server has successfully started:

1. Navigate to the *agent_relay_installation*\bin directory
2. Run the *run_agentrelay.cmd* batch file (Windows), or *start_agentrelay.cmd* (Unix/Linux).

Start the agent relay before starting any agents that will communicate through it.

Accessing IBM uDeploy

1. Open a web browser and navigate to the host name you configured during installation.
2. Log onto the server by using the default credentials.

   User name: *admin*

   Password: *admin*

   You can change these later by using the **Settings** tab on the IBM uDeploy web application, see *System Settings*
3. Activate the license. A license is required in order for the agents to connect to the server. Without a license, IBM uDeploy will be unable to run deployments. For information about acquiring and activating a license, see the section called “Licenses”.

IBM uDeploy Security

IBM uDeploy provides a flexible, role-based security model that maps to your organizational structure. Different product areas, such as components, can be secured by roles. Each area has a set of permissions available to it. To configure security for an area, you create roles using the available permissions—execute, read, write, and so forth.

So, how are permissions applied to users? First, global default permissions can be granted. Default permissions are granted by product area and apply to all users. If default permissions are granted for, say, the agent area, a user will have those permissions even if she is also part of a group or role that does not.

Another way users can be granted permissions is by being a member of a group. Groups can have default permissions that apply to all group members. If a user is assigned to a group with default permissions for the agent area, as above, she will have those permissions even if she is also assigned a role that does not have them.

Finally, users can be assigned to roles. Role members inherit a role's permissions. Except for UI and system security, users are assigned to roles on an item by item basis. For example, a user can be assigned a role that enables them to see only one application or only one component. Both groups and individual users can be assigned to roles.

Roles and permissions, including default permissions, are configured on an area by area basis; granting the execute permission to one role does not grant it to another. The default admin role has all permissions, but you can create another user with all permissions by creating a role for each area with all permissions granted, then assigning the user to each role. Typically, new roles are added to product areas during setup and occasionally thereafter.

While any number of roles can be created for an area, areas themselves cannot be created, modified (the available pool of permissions cannot be changed), or deleted.

Generally, you perform the following steps in order when setting-up security:

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

   Use the UI security area to quickly assign access permissions to the different areas of IBM uDeploy.

   Use the system security area to assign usage permissions, including the ability to define security for other users.

2. **Authorization Realms**. Authorization realms are used by authentication realms to associate users with groups and to determine user access. IBM uDeploy includes both an internal database for storing security information as well as 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 IBM uDeploy, the LDAP integration is recommended. If you are evaluating IBM uDeploy, it is not necessary to set up the LDAP integration—full security is configured and enforced by the server.

3. **Create Groups and Define Default Permissions.** Determine default permissions by product area. Global default permissions can be granted.

4. **Create Authentication Realm**. The authentication realm is used to determine a user's identity within an authorization realm. If more than on 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.

5. **Add Users.** Add users to an authentication realm, then assign them to groups and roles. If your are using LDAP, you can import users and map them to the security system.

Roles and Permissions

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.

IBM uDeploy maps key product features or areas to security roles. Each area has several permissions defined for it (listed below). 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.

Generally, permissions fall into one of these groups:

Table 5. Common Permissions

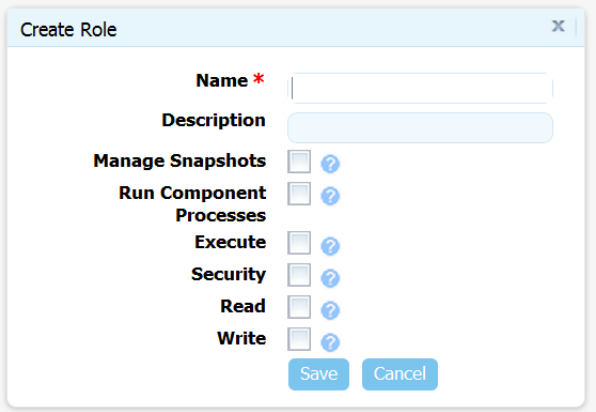
Permission	Description
Security	Enables users to change an item's security settings. For example, a user with this permission for agents can determine which users can view, configure, and set security for them.
Write	Enables users to add, change, and delete items. A user with this permission for components can create a component.
Read	Enables users to read (view) an item, but not change it or create another of its type. A user with this permission for agents, say, will be able to see agents within the user interface, but will not be able to 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.

Default Roles

IBM uDeploy ships with several role types mapped to product areas. Every area or type has a set of available permissions. The *application* type, for instance, has the Manage Snapshots permission in addition to the common permissions. User-defined roles within a type can choose from among the permissions available for that type.

Every product area has one role typically called `Admin` or `Administrator` that 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.

Figure 1. Application Role Permissions



You can quickly grant a role type's permissions to all users using the Default Permissions tab. Note that default permissions cannot be granted for system and UI security.

Creating and Editing Roles

1. Display the Role Configuration pane (*`Settings > Security Role Configuration`*).
2. From the list of product areas, select the area where you want to add a role.
3. Display the Create Role dialog (`Create Role [button]`).

   All permission available for this product area are displayed.
4. Select the permissions you want granted to this role.

All roles have the following permissions available. Other permissions—if any—are described in the following sections.

Table 6. Permissions Available for Every Role

Permission	Description
Security	Manage security for the effected feature area.
Write	Create, edit, or delete items for this product area.
Read	Access or view items for this product area.

Agent Roles

Agent roles define the functions users can perform with agents and agent pools. Available permissions are read, write, and security.

To add users to agent roles:

1. Display the Security tab for the target agent (*`Resources > Agents/Agent Pools > [selected agent/agent pool] > Security)`*.

   All roles defined for agents and agent pools are displayed.
2. Use the Add Role Member action for a specific role, then select the user.

All users are available. As shipped, IBM uDeploy provides an `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

Application Roles

Application roles define the functions users can perform with applications. In addition to the standard permissions, others are:

Table 7. Application Roles

Permission	Description
Manage Snapshots	Create and edit snapshots for this application.
Run Component Processes	Run associated component processes outside of the application.

To add users to application roles:

1. Display the Security tab for the target application (*`Applications > [selected application] > Security`*).

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides an `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

Component Template Roles

These roles define the functions users can perform with component templates. Available permissions are read, write, and security.

To add users to component template roles:

1. Display the Security tab for the target template (*`Components > Templates > [selected template] > Security`*).

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides an `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

Component Roles

These roles define the functions users can perform with components. In addition to the standard permissions, others are available:

Table 8. Component Roles

Permission	Description
Manage Versions	Create and delete versions for this component.

To add users to component roles:

1. Display the Security tab for the target component (*Components > [selected component] > Security)*.

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides an `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

Environment Roles

These roles define the functions users can perform with environments. Available permissions are read, write, execute, and security.

To add users to environment roles:

1. Display the Security tab for the target environment (*Components > [selected component] > Security)*.

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides an `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

License Roles

These roles define the functions users can perform with licenses. Available permissions are read, write, and security.

To add users to license roles:

1. Display the Security tab for licenses (*Settings > Licenses > Security)*.

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides an `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

Resource Roles

These roles define the functions users can perform with resources. Available permissions are read, write, execute, and security.

To add users to resource roles:

1. Display the Security tab for the target resource (*Resources > [selected resource] > Security)*. (For resource groups: *Resources > Resource Groups > [Edit Group action] > Security*).

All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides `Admin` role with all configured permissions granted. By default, `Admin` has a single user—`admin`.

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 areas' 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 tab to set default permissions, for both the groups you create and those shipped with the product.

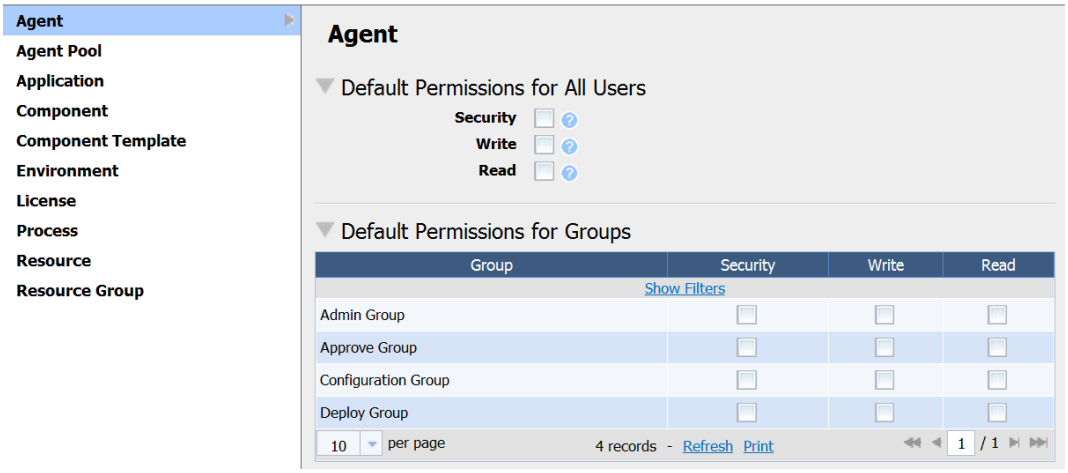
Users added to a group inherit the group's default permissions.

Setting Default Permissions

To set default permissions:

1. Display the Default Permissions pane (*`Settings > Default Permissions`*).
2. From the list of product areas, select the area you want to use.

Figure 2. Default Permissions for Agent Area



Selecting an area displays the permissions available for it. User-defined groups are configured independently.

3. Check the permissions you want to grant for the selected group.

   The following table lists the available permission.

Table 9. Product Area Privileges

Role	Read	Write	Security	Execute	Snapshots	Comp. Procss.	Versions
Agent	X	X	X				

Role	Read	Write	Security	Execute	Snapshots	Comp. Procss.	Versions
Agent Pool	X	X	X				
Application	X	X	X	X	X	X	
Component	X	X	X	X			X
Component Template	X	X	X				
Environment	X	X	X	X			
License	X	X	X				
Resource	X	X	X	X			
Resource Group	X	X	X	X			

Authorization Realms

The Authorization Realms pane is used to create user groups and authorization realms. Authorization realms associate users with roles and work with authentication realms to determine which users can access IBM uDeploy. The authorization realms available are:

- **Internal Storage.** Uses internal role management. The default authorization realm—`Internal Security`—is of this type.
- **LDAP.** Uses external LDAP role management.

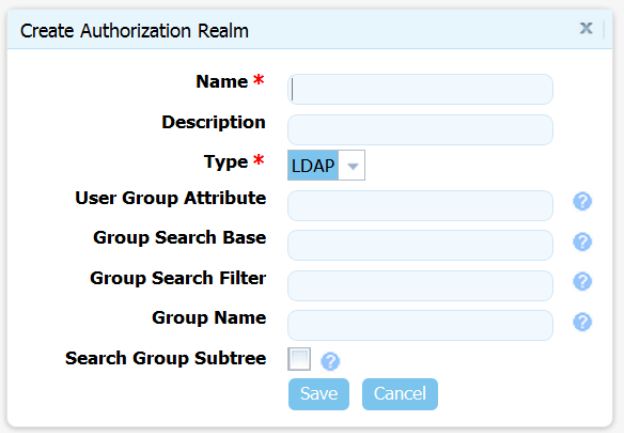
Creating an LDAP Authorization Realm

An LDAP authorization realm uses an external LDAP server for authorization.

To create an LDAP authorization realm:

1. Display the Create Authorization Realm dialog (*`Settings > Security > Authorization > Create Authorization Realm [button]`*).

Figure 3. Create Authorization Realm Dialog



2. Ensure that `LDAP` is selected in the Type list box, then specify the following:.

Table 10. LDAP Authorization Realm Properties

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	Searches the subtree for the roles if checked.

Groups

Groups are logical containers that serve as a mechanism to grant permissions to multiple users; members automatically share a group's permissions. Default 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 as well.

To create a group:

1. Display the Create Group dialog (*`Settings > Security > Authorization > Groups > Create Group [button]`*).

2. Provide a name for the group. The name appears in the Default Permissions pane.

3. Select an authorization realm. Groups are only valid for the selected realm.

IBM uDeploy provides several default groups and users, which are listed in the following table. The default groups and users are part of the internal security authorization realm.

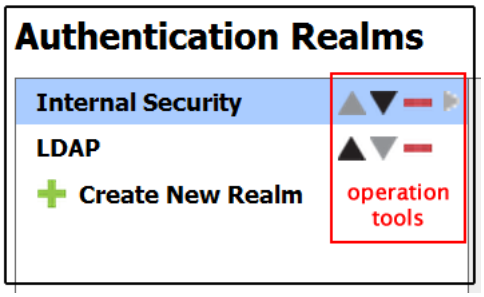
Table 11. Default Groups

Group	Users
Admin Group	admin
Approve Group	approve
Configuration Group	config
Deploy Group	deploy

Authentication Realms

The Authentication Realms pane is used to create authentication realms and users. Authentication realms determine a user's identity within an authorization realm. Authentication is determined following the hierarchy of realms displayed on the Authentication Realms pane. In the example below, authentication will first be determined in the Internal Security realm followed by the LDAP realm. A user listed in the LDAP realm may have different authorizations from those in the other realms.

Figure 4. Authentication Realms Precedence



If you have a number of authentication realms, you can reorder them using the operation tools. Each realm can be moved up to a higher priority, moved down, or deleted by using the operation tools.

Creating an Authentication Realm

1. Display the Create New Authentication Realm (*`Settings > Security > Authentication > Create New Realm`*).
2. Enter a name and 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. Selecting `Internal Storage` completes the process.

Creating an LDAP Authentication Realm

If you selected `LDAP`, provide information about your LDAP installation:

Table 12. LDAP Authentication Realm Properties

Field	Description
Context Factory	Context factory class used. 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.
Use DN Pattern	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.

Field	Description
User Search Base	Base directory used to execute group searches, such as ou=employees,dc=mydomain,dc=com.
User 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).
Search User Subtree	If the LDAP user names are case sensitive, check the box to treat different-case names as different users.
Search Connection DN	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 Connection Password	Password used when connecting to LDAP to perform searches.
Name Attribute	Contains the user's name, as set in LDAP.
Email Attribute	Contains the user's email address, as set in LDAP.

Once configuration is complete, when a new user logs on using their LDAP credentials, they will be listed on the Authentication Realm Users pane. It is best practice not to manage user passwords nor remove users from the list. If an active user is removed from IBM uDeploy, they will still be able to log onto the server as long as their LDAP credentials are valid.

Creating Users

When adding a new user, the user name and password is what the individual will use when logging into IBM uDeploy. 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.

Importing LDAP Users

Unless using LDAP authorization realm, valid LDAP users can log on but will have no permissions. To provide permissions, import them first and define their permissions before they log on. You can import users from existing LDAP systems into IBM uDeploy-managed authentication realms.

To Import LDAP Users

1. Display the Create User dialog(*`Settings > Security > Authentication Realms > [select LDAP realm] > Import User [button]`*).
2. Enter the name of the user.

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

Tokens

Tokens provide authorization for agents and users. Agents use tokens when performing process steps and communicating with the IBM uDeploy server and external services. Users can use tokens with the CLI client, and instead of supplying a user name and password in certain situations.

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

To create a token:

1. Display the Create New Token dialog (*`Settings > Security > Tokens > Create New Token [button]`*).
2. From the User drop-down list box, select the user who will use the token.
3. Specify the expiration date and time.

Tokens can be used immediately after being created.

User Interface Security

These roles determine which parts of the IBM uDeploy web application users can access. Each tab, such as Reports, on the web application's home page can be restricted. Available permissions are:

Table 13. Web UI Permissions

Permission	Description
Resources	Access the Resources tab.
Applications	Access the Applications tab.
Components	Access the Components tab.
Configuration	Access the Configuration tab.
Reports	Access the Reports tab.
Deployment Calendar	Access the Calendar tab.
Work Items	Access the Work Items tab.
Settings	Access the Settings tab.
Dashboard	Access the Dashboard tab.

To add users to Web UI roles:

1. Display the System Security tab (*`Settings > Security > Security)`*. (For resource groups: *`Resources > Resource Groups > [Edit Group action] > Security`*).

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides the following roles:

Table 14. Default Web UI Roles

Role	Description
Deployment Engineer	Access the Reports, Calendar, Work Items, and Dashboard tabs.

Role	Description
Approver	Access the Reports, Work Items, and Dashboard tabs.
Administrator	Access all tabs.
Configuration Engineer	Access all tabs except *Calendar* and *Work Items*.

System Security

These roles define the functions users can perform with the IBM uDeploy server (also referred to as system security). Available permissions are:

Table 15. Server Permissions

Permission	Description
Security	Manage security configuration; users without this permission cannot access or change the security functions.
Manage Plug-ins	Grants users the ability to install new plug-ins.
Create Subresources	Ability to create subresources.
Create\Manage Resource Roles	Create and delete resource roles.
Create Components	Create components.
Create Applications	Create applications.
Create Component Templates	Create component templates.
Manage Licenses	Add and remove licenses.

To add users to system security roles:

1. Display the System Security tab (*`Settings > Security > Security)`*.

   All defined roles are displayed.

2. Use the Add Role Member action for a specific role, then select the user.

   All users are available. As shipped, IBM uDeploy provides `Configuration Manager` and `System Administrator` roles; the latter has all configured permissions granted. By default, `System Administrator` role has a single group—`Admin Group` (with user admin), and the `Configuration Manager` role also has a single group—`Configuration Group` (with user config).

System Settings

Installing Plug-ins

Plug-ins can be installed at any time. Download plug-ins from UrbanCode, an IBM Company's plug-in page:

http://plugins.urbancode.com

To install a plug-in:.

1. Download the plug-in from the UrbanCode, an IBM Company plug-in page using the link supplied above. Plug-ins are provided in compressed format (ZIP files). There is no need to decompress the file.

   You can also load your own plug-ins. For information about creating plug-ins, see ???.

2. From the Automation Plug-ins pane, display the Load Plug-in dialog (*`Settings > Plugins > Load Plugin [button]`*).

3. Enter the path to the compressed plug-in file then use the Submit button.

If the plug-in loaded successfully, it will be listed on the Automation Plug-ins pane as soon as the process finishes. Once installed, plug-in functionality is available immediately.

Locks

Alock is a routinely used 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 on the **Dashboard** tab with a status of *`Running`*. If a locked process takes longer to complete than expected, you can cancel the process from the **Dashboard**, or investigate it fully with the *`Settings`* tab.

1. Display the **Lock** pane by clicking the *`Locks`* link on the **Settings** tab (*`Home > Settings > Locks`*).

   The **Lock** pane displays the following information:

   Table 16. Lock Fields

Field	Description
Name	The name identifies the lock. The displayed name is a concatenation of the component or application name (depending on type) + process name + resource name.
Type	Indicates whether the process creating the lock is a component- or application-type. Locks can

Field	Description
	only be applied to component or application processes.
Component/Application	Displays the name of the component or application containing the lock. Clicking an item displays (depending on the type) the **Component** pane, or **Application** pane, where you can investigate the lock.
Resource/Environment	Displays the name of the resource or environment containing the lock. Clicking an item displays (depending on the type) the **Resource** pane, or **Environment** pane.
Process	Displays the name of the process containing the lock. Clicking an item displays the process in the process editor.
Actions	Lists the available actions.

2. Resolve the lock by selecting an action:

Table 17. Lock Actions

Action	Description
View Request	Displays the process log for the process containing the lock. You can use the *Actions* field on the displayed pane 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 IBM uDeploy server and or agents go down while a locked process is running, IBM uDeploy will automatically restore any interrupted processes along with any locks they might contain once service is restored.

Post-Processing Scripts

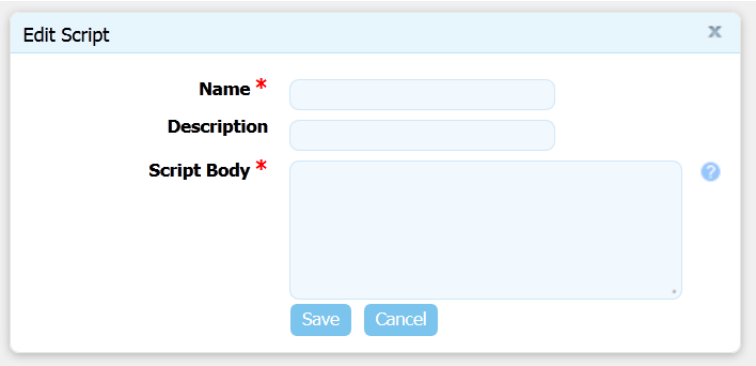
IBM uDeploy component processes perform post-processing whenever a plug-in step finishes execution. Typically, post-processing scripts ensure that expected results occurred. You can use your own JavaScript script instead by instructing IBM uDeploy to use your script when you define the step. See ???.

When a step finishes, the agent performing the step will run your script (the script must be written in JavaScript). 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.

To create a script:

1. Display the **Edit Script** dialog (*Settings > Post Processing Scripts*).

Figure 5. Edit Script Dialog



2. 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 ???.

3. Enter or paste the 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.

The IBM uDeploy server log file is normally found in the following location: `IBM uDeploy_root \var\log\deployserver.out`.

Inventory and Component Statuses

Statuses can be used to track component version and inventory states. Inventory statuses can track component versions in environments and resources. You can create inventory statuses for any requirement —for instance, you might copy files to an agent without running them, and apply a `Staged` status to them; when you install the version, you might set the status to `Active`.

Version statuses are used with application gates (see ???) to ensure that only component versions that meet certain criteria are deployed.

Creating Statuses

Component version and inventory statuses are defined with the Statuses tab (*Settings > Statuses*).

To create a status:

1. Use the *Add Status* button for the type—inventory, version—you want to create.

2. Configure the status using the Add Status dialog box:

Table 18. Status Parameters

Parameter	Description
Name/Description	The name identifies the status and is used in process steps and in the UI.
Color	Displayed in the UI.
Unique	When checked, only one instance of the status can be used for the component. For inventory

Parameter	Description
	statuses, an application will remove the status from any existing version in the environment or resource inventory. For version statuses, the status can only be used by one version at a given time.
Required Component Role	User role required before a user can add the status to a component.

3. Save your work when finished.

Statuses are stored in the IBM uDeploy database.

Using Statuses

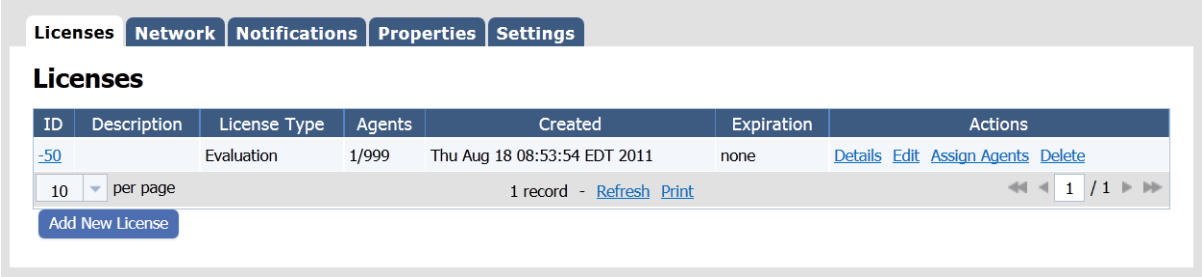
Version statuses can be added to components on the Versions tab (*`Components > [selected component] > Versions > [selected version]`*).

Inventory statuses can be added to a component with the Inventory Update application process step or the Add Inventory Status plug-in step.

Licenses

The **Licenses** pane is where you manage user licenses--adding or deleting licenses, and assigning agents to them. Display the **Licenses** pane by clicking the *`Licenses`* link on the **Settings** window (*`Home > Settings > License`*). You can also access the pane through the **Resources** tab (*`Resources > Agents > License`*).

Figure 6. Licenses Pane



Adding a License

To add a license:

1. Display the **Add New License** dialog by clicking the **Add New License** button.
2. Paste the license text supplied by UrbanCode, an IBM Company into the *`License`* field.
3. Optionally, add a description.
4. Click **Save** when you are done.

To see information about a license, display the **License Details** pop-up by clicking the *`Details`* link.

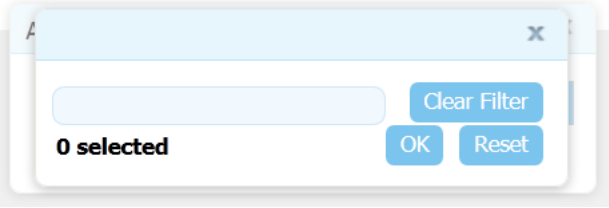
Adding Agents to a License

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 checked. See the section called “System Settings”.

To add an agent to a license manually:

1. Display the **Assign Agents to License** pop-up by clicking the *Assign Agents* link for the intended license.
2. Select an agent by clicking the *Agents* field. A selection-type pop-up is displayed listing any agents not already assigned to the selected license.

Figure 7. Assign Agents to License Pop-up



3. Select the agent or agents you want to add to the license.
4. Optionally, you can filter the listed agents by entering search text into the text field.
5. After select agents, click **OK** to close the selection pop-up.
6. If you want to restart the selection process, click **Reset**.
7. When you are finished, click **Save**.

Modifying or Deleting a License

To modify or update an existing license:

1. Display the **Edit License** dialog by clicking the *Edit* link for the license you want to change.
2. Edit the information shown in the *License* field.
3. Click **Save** when you are done.

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

Log Settings

Logging is done with the Log4j logging framework from Apache. Log4j logging is configured with the `server_install_directory\conf\server\log4j.properties` file, and can be configured at run time.

The IBM uDeploy server log file can be found in the following location: `server_install_directory\var\log\deployserver.out`.

Network Settings

Network Relay

A network relay is used in conjunction with an agent relay. The network relay reverses the direction of communication through a firewall between the IBM uDeploy server and agent relay. A network relay is only used when you want the server to connect to the relay instead of the reverse (which is default). To create a network relay an agent relay must be created. (See the section called “Installing Agent Relays” to create an agent relay)

Creating a Network Relay

To create a network relay, display the network pane (`Home>Settings>System>Create New Network Relay`).

1. Enter the name of the network relay.
2. Identify the Host and Port.
3. Indicate if the Network Relay will be Active by checking the box.

Notifications

IBM uDeploy can send email notifications whenever user-defined trigger events occur. Notifications can be sent when a deployment finishes or an approval is required, for example. Notification recipients are defined with the security system's (see *IBM uDeploy Security*) LDAP integration. If you have not already done so, set up LDAP prior to configuring notifications. IBM uDeploy relies on LDAP and an associated e-mail server to send notifications.

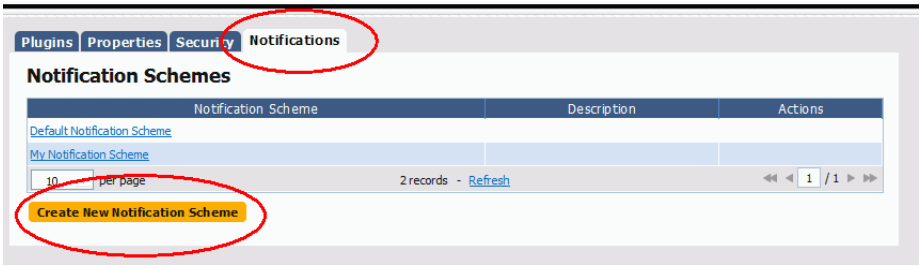
Note

IBM uDeploy requires an external SMTP mail server to send notifications. For information about configuring a mail server, see the section called “System Settings”.

When setting up notifications, you select both the triggering events and the role, which is inherited from the security system, to determine which users will receive notification. 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. The default notification scheme, which sends notifications to the application and admin default roles (see *IBM uDeploy Security*), can be edited or you can create your own scheme.

To set up your own notifications, display the Notifications pane (`Settings > Notifications`).

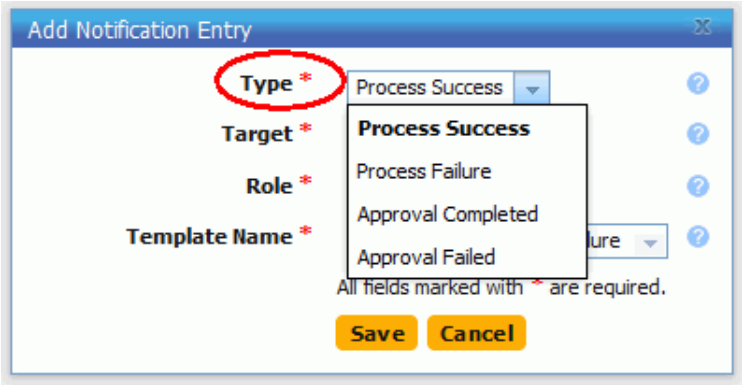
Figure 8. Notification Schemes



Configure the new Scheme. Here, you will be setting up the who/when for notifications. Once configured, you can come back add additional Entries to the Scheme or edit an existing one.

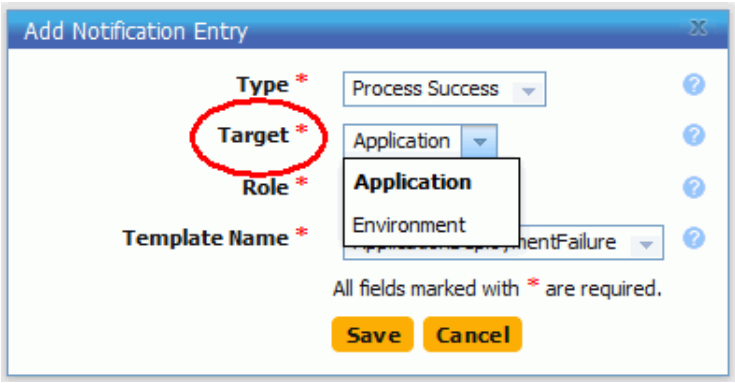
Notification Type. The process type is determined mainly by the type of recipient. For example, a deployment engineer would be interested in being notified about a failed deployment.

Figure 9. Notification Type



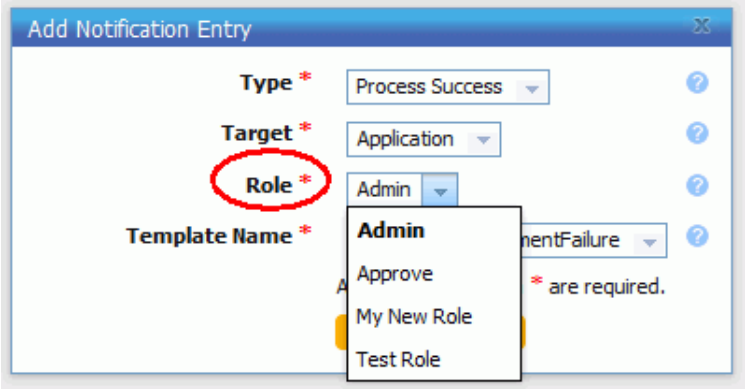
Notification Target. When setting the target, the application option will only send out notifications when the event selected above corresponds to an Application. For example, the "Process Success" event, when pared with the "Application" Target would trigger a notification when a Process (an application deployment) is successful. Similarly, the same event type, when used with the "Environment" target would instigate a notification when a successful deployment has been run in an Environment (e.g., SIT, PROD).

Figure 10. Notification Target



Notification Role. The Role corresponds to those set in the Security System. Any individual assigned the Role you select will receive an e-mail.

Figure 11. Notification Role



Template Name. The available templates are provided by default and should suffice for all your needs; they format the e-mail being sent. Which template you use is based on why you want to set up a notification and the recipients of the notification. However, if the default templates do not suit your needs, you can create your own.

Application deployment failure/success. Sends notifications about a specific application to the specified users, based on the role setting.

Task readied/created/completed. This template is used to report back on the state of manual tasks.

Deployment readied. A specialized e-mail template for letting people know a deployment has been prepared.

Approval created/failed. These templates are used to notify the status of an approval.

Once you have the entry done, add others using the same process. If you want to use the new notification scheme with existing applications, modify the application settings.

Creating Notification Templates

Notification Templates are XML files located on the server'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 new Notification Template:

1. Start a new XML file.
2. Enter Script. (Notification templates only supports Velocity Reports)
3. Save file in the server's `conf/server/notification-template` file folder.
4. Restart the server.

Output Log

Server output is written to the `server_install_directory/var/log/deployserver.out` log file. You can open the file directly or access it from the UI (*`Settings > System > Output Log`*).

The information written to the log file is determined by the settings in the `log4j.properties` file which is found at `server_install_directory\conf\server\log4j.properties`. You can edit the file directly or through the UI, see the section called "Log Settings "

System Properties

System properties are global variables. System properties are available on the Settings tab (*`Settings > Properties`*).

System properties are referenced like this:

`${p:system/propertyName}`

If you create system variable `SUCCESS`, for example, you would reference it like this:

```
echo ${p:system/SUCESS}
```

Output in this case:

```
SUCESS
```

System Settings

Table 19. System Settings Field

Field	Description
External URL*	URL used by agents and users to connect to the IBM uDeploy server.
Only Groups in Security Roles	When checked, privileges are assigned to user groups, not individual users.
Automatic Version Import Check Period (seconds)*	The number of seconds between when IBM uDeploy polls components for new versions. If changed, the server must be restarted before the change becomes effective. UrbanCode, an IBM Company recommends that the value be set no lower than 15 seconds.
Mail Server Host	Host name of the mail server used for notifications. IBM uDeploy can send notifications to users based on user-configured trigger events (to set up notifications, see the section called “Notifications”). IBM uDeploy requires an external SMTP mail server to send messages. To disable notifications, leave the field blank.
Mail Server Port	SMTP port used by the notifications mail server.
Secure Mail Server Connection	Specifies whether the SMTP connection is secure. The default value is unchecked--not secure.
Mail Server Sender Address	Sender address for email notifications.
Mail Server Username	User name for sending email notifications. Some e-mail servers and firewalls treat e-mails with different sender and user names suspiciously--you might want to use the same name for both fields.
Mail Server Password	User password for sending email notifications.
Hour to Clean Versions*	Time of day when versions are cleaned. Value must be an integer between 0 (midnight) and 23 (11 pm).
Days to Keep Versions*	Number of days component versions are kept. A value of `-1` means they are kept indefinitely.
Number of Versions to Keep*	Number of component versions to keep. A value of `-1` means all are kept.
Archive Path	Path where the compressed file containing archived component versions is written. If blank, the compressed file is not written (and no archive kept).
Automatic License Management	Determines whether new agents are assigned to a licenses automatically. If checked, agents

Field	Description
	are assigned to the license with the most time remaining before it expires. The default value is checked--assign agents automatically.

* = required

Preview Version Cleanup

To preview the component versions that will be archived the next time an archive file is created, click the *Preview Version Cleanup* link. Using the link displays the **Version Cleanup Preview** dialog, which lists the to-be-archived component versions.

Configuration

The IBM uDeploy Configuration tool enables you to directly manage application, component, and environment configuration data.

Configuration data is manipulated at the application, component, and environment levels:

- **Component**

  A component refers to any file that you want to include in the build process; components are associated with the configuration data required to deploy them.

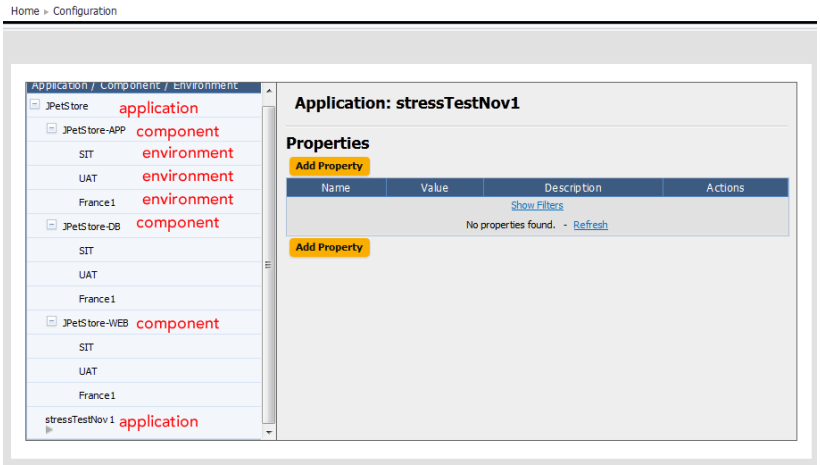
- **Application**

  Applications represent a group of components deployed together by component version and environment. Applications also map the hosts and machines (called resources) components require within every environment.

- **Environment**

  An environment is a collection of resources that host an Urban Deploy application.

Figure 12. Configuration Tab

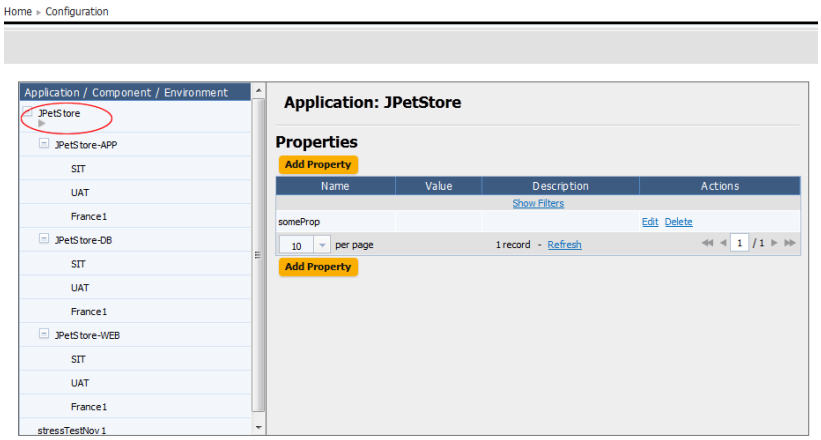


Access the Configuration Tool by clicking on the Configuration tab.

Application Configuration

You attach properties to an application by using the Configuration Tool's *`Application: Add Property`* button. Typical application-level properties include items that are the same in all environments, such as base-install paths.

Figure 13. Application Properties panel



Access the Configuration Tool Application panel by clicking on an application in the *Application/Component/Environment* list box.

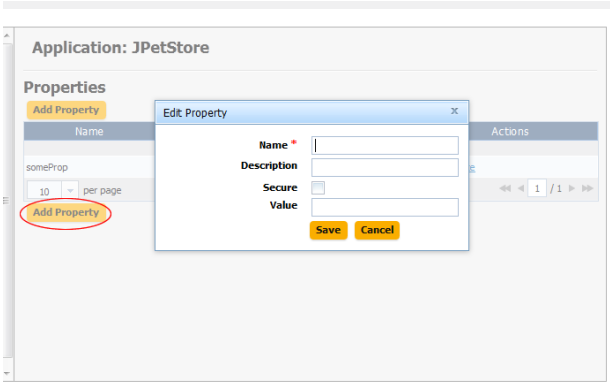
Adding Application Configuration Properties

To add a property to the selected application:

1. Use the *Add Property* button.

   The Edit Property pop-up is displayed.

Figure 14. Edit Property pop-up



2. Enter the property's name in the *Name* field.

   While component fields can be of any size, configuration properties are restricted to 4,000 characters.

3. Enter a description of the property in the *Description* field.

4. Specify whether the property is secure by using the *Secure* check box.

   Secure properties are stored encrypted and displayed obscured in IBM uDeploy's user interface.

5. Enter a value for the property in the *Value* field.

6. Save the property by using the *Save* button.

7. To discard your work, use the *Cancel* button.

Modifying and Deleting Application Configuration Properties

Modifying Application Configuration Properties

To modify a previously created property, use the *Edit* link in the Action column to display the Edit Property pop-up.

Deleting Application Configuration Properties

To delete a property, use the *Delete* link in the Action column.

Component Configuration

The Urban Deploy Configuration tab enables you to configure applications and their components from a single location. Configuration data is manipulated at the application, component, and environment levels:

- component

  A component refers to any file that you want to include in the build process; components are associated with the configuration data required to deploy them.

- application

  Applications represent a group of components deployed together by component version and environment. Applications also map the hosts and machines (called resources) components require within every environment.

- environment

  An environment is a collection of resources that host an Urban Deploy application.

Access the Configuration Tool by clicking on the Configuration tab.

Environment Configuration

The Urban Deploy Configuration tab enables you to configure applications and their components from a single location. Configuration data is manipulated at the application, component, and environment levels:

- component

  A component refers to any file that you want to include in the build process; components are associated with the configuration data required to deploy them.

- application

  Applications represent a group of components deployed together by component version and environment. Applications also map the hosts and machines (called resources) components require within every environment.

- environment

An environment is a collection of resources that host an Urban Deploy application.

Access the Configuration Tool by clicking on the Configuration tab.

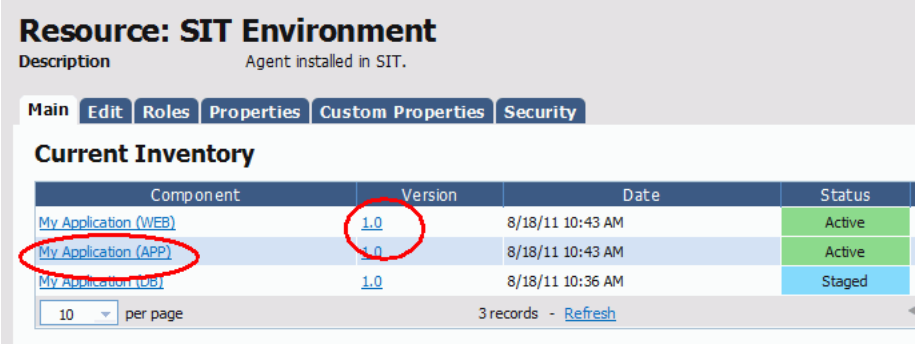
Inventory

The Inventory shows what Applications and Components have been deployed, including the current Versions that are running on the Resource within an Environment. The inventory provides complete visibility into the different Versions of your Applications which can be tracked back to the original artifacts imported into IBM uDeploy. There different views of the current inventory, depending on where in IBM uDeploy you are. Inventory information is available on the individual Components, for every Application Environment, as well as for each Resource (agent).

Resources Inventory

If you want to see what Components are sitting on the SIT Environment, go to Resources and select the agent that is running in the Environment. From here, selecting either the Component or its Version will take you to the Component's page if you need more information.

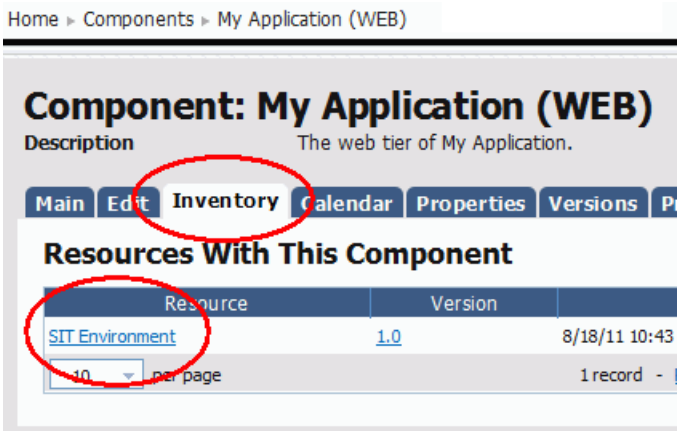
Figure 15. Resource inventory



Component Inventory

Unlike the Resource Inventory, the component inventory tells you what Version of the Component is running on a Resource. For example, if the Component is currently deployed to multiple machines, they would all be displayed. For here, you can go navigate to the Resource.

Figure 16. Component inventory



Environment Inventory

For any given Application Environment, the environment inventory tells you both what version of any given Component is running on a particular Resource. If multiple Versions are running on different Resources, they will all be listed.

Glossary

A

Agent	Agents are light-weight Java processes that run on the agent machine. Agents allow the distribution of tasks for performance and multi-platform support. The agent contacts the server whenever the agent process is started. Since the agent communicates with the central server, it need not be on the same network as the central server. However, the agent must be able to open a socket connection to the server. By default, all communication between the central server and the agent is not secure. Communication may be secured using SSL. Agents are light-weight Java processes that run on the agent machine. Agents allow the distribution of tasks for performance and multi-platform support. The agent contacts the server whenever the agent process is started. Since the agent communicates with the central server, it need not be on the same network as the central server. However, the agent must be able to open a socket connection to the server. By default, all communication between the central server and the agent is not secure. Communication may be secured using SSL.
agent	An agent is a lightweight process that runs on a host and communicates with the IBM uDeploy server. Agents manage the resources that are the actual deployment targets.
Agent Filters	Agent Filters are used to select one or more agent(s) at run time and to monitor the quiet period.
agent pools	An agent pool helps you organize and manage agents installed in different environments.
Agent Relays	The new communication relay acts as a proxy for agents (which perform work on behalf of the AnthillPro server) that are located behind a firewall or in another network location. Available under separate download.
agent relays	An agent relay is used to communicate with remote agent. An agent relay requires that only a single machine in the remote network contact the server.
AHPTool	AHPTool is a command-line interface for AnthillPro that provides communication between agent-side commands, scripting and the AnthillPro central server. AHPTool is focused on setting and retrieving information from the AnthillPro server in the context of a running workflow: It can be used to look up or set properties at the system, step, request, job, Build Life and agent levels. AHPTool can upload and retrieve Test, Coverage, Analytics, or Issue data in the form of an XML document to AnthillPro -- making it an excellent integration point for writing your own plug-in or script.
Any Agent Filter	An any agent filter selects the first available agent. Also, it returns all online agents in the environment, ordered by a combination of current load and throughput metric.
applications	An application is the mechanism that initiates component deployment; they bring together components with their deployment target and orchestrates multi-componet deployments. An Application must have one component.

application process	An application process can run automatically, manually, or on a user-defined schedule. An application process can orchestrate the entire process including putting servers on-and-off line for load-balancing as required.
Application Security Report	The application security report provides information about user roles and privileges defined for IBM uDeploy-managed applications.
Artifact	For any artifact associated with this project, the files produced by a workflow published as the artifact.
artifacts	Artifacts are files, images, databases, configuration materials, or anything else associated with a software project.
Artifact Retention Policy	Manages how long you will keep the artifacts in order to save disk space.
Artifact Set	An Artifact Set is a label for a collection of build artifacts. Artifact Sets are used to define dependency relationships: Any project that another project depends on generates one or more collections of files used by the dependent project.
Authentication Realm	Authentication Realms are used to determine a user's identity within an Authorization Realm.
Authorization Realm	Authorization reals are used to associate Users with Roles and to determine user access to AnthillPro.

B

Bar Chart	Displays the data in a bar chart embedded in HTML. Data must all be numeric.
blackout	Blackouts are set per-environment, per-application. Once set, no deployments (nor Snapshots) can be scheduled to occur in that Environment. Any previously scheduled deployments to the Environment will fail if they fall within the blackout date you set.
Build-Farm	The default environment containing all agents used for pure-build processes.
Build Life	represents all the transformations a build has gone through, and the processes such as deployments and testing that the artifacts have undergone.
Build Life Links	Build Life Links are used to pass the URL of resources outside of AnthillPro to the Build Life.
Build Life Originating Workflow Request Property	When set as a workflow property, the value will get pushed to the Build Life originating workflow before the build begins.
Build Life Properties	typically used to hold variable data for builds or to create an audit trail. Once the property is set, and the build has run, the Build Life Property will be visible on the Build Life page.
Build Request Property	Build Life Tools- Build Life Tools are available through the UI to help you diagnose problems and mange Build Lives.
Build Request Property	The property will get pushed to the build request before the build begins.
Build Workflow	A build workflow defines the process for running jobs.

C

Caching Proxy	Could reduce bandwidth and improve AnthillPro’s response time. This is especially helpful in distributed development environments: the proxy can improve performance for users at off-site locations because commonly used pages are loaded from the locally stored cache.
Cleanup	Cleanup configures AnthillPro to periodically cleanup old Build Lives, build request and miscellaneous jobs generated by a project. Cleanup is on a per-project basis, so every project that uses the same Life-Cycle Model will have the same policy. To clean up the Build Life, the user can delete, inactivate, or archive the Build Life.
Cleanup Build Lives	determines when to cleanup build lives. The cleanup of build lives is base on the status the build life has achieved.
Cleanup Policy	Specifies when to delete information about old Build Lives and other task associated with the project.
Cleanup Schedule	A Cleanup Schedule kicks-off the Cleanup.
Cleanup Build Request	A Cleanup Build Request determines when build request are cleaned up.
Clone Instance	Cloning an instance of your AnthillPro server will allow you to change scripts, optimize processes, improve build performance, move the server, etc., without having to experiment on the production server.
CodeStation	CodeStation is the name of AnthillPro’s artifact repository management and access tool set. It provides support of dependencies of third-party tool kits and software libraries. CodeStation is also responsible for bringing the dependency management lookup and retrieval utilities to the individual developer.
CodeStation	CodeStation is IBM uDeploy's artifact repository. It provides secure and tamper-proof storage. It tracks artifact versions as they change and maintains an archive for each artifact.
CodeStation Artifact Time-to-Live	This feature determines how long unused artifacts remain in the cache.
components	A component represents deployable items along with user-defined processes that operate on it. Components are deployed to a resource by agents.
component inventory	A component inventory tells you what version of the component is running on a resource.
component process	A component process is a series of user-defined steps that operate on a components artifacts.
Component Security Report	A component security report provides information about user roles and privileges defined for components.
component template	Component templates enable you save and reuse component processes and properties. Components based on templates inherit the template's properties and process.

Conflict Strategy	A Conflict Strategy allows you to control how AnthillPro acts when a conflict occurs within the dependency tree.
Cron Schedule	A cron schedule may be configured to consider more complex criteria than a simple interval.

D

Defined-value property	Basic property type. Displays a secured field for the user (either the property name or default value) when running a build.
Dependencies	Specifies the specific artifacts, including version, from other projects that should be retrieved in order to support the workflow. Also specifies the artifacts from other projects and the directory in the checked-out source that those artifacts should be placed in.
deployment	A deployment is the process of moving software through various preproduction stages to final production.
Deployment Average Duration Report	A deployment average duration report provides the average deployment time for applications executed during a user-specified reporting period.
Deployment Count Report	A deployment count report provides information about the number of deployments executed during a user-specified reporting period.
Deployment Detail Report	The deployment detail report provides information about deployments executed during a user-specified reporting period.
Deployment Reports	A deployment report provides information about user roles and privileges defined with the IBM uDeploy security system.
Deployment Total Duration Report	A deployment total duration report provides total deployment for applications executed during a user-specified period.
design space	The process editor's work area, where plug-in steps are configured and process flows defined.
Dev-kit	The Dev-kit provides comprehensive documentation on AnthillPro scripting, remoting, using SOAP-based web services with AnthillPro, the AHPTool (command-line tool) and creating your own Plug-in for AnthillPro.
digital certificate	A digital certificate is a cryptographically signed document intended to assure others about the identity of the certificate's owner.
Distributed Servers	Distributed Servers is a complimentary product to AnthillPro designed to reduce the reliance on WAN connections by performing all the heavy lifting (builds, etc.) on local networks. There are three main components to Distributed Servers: Distributed Web Interface, Codestation 2.0, and Agent Relay.
Distributed Web Interface	The interface mirrors the AnthillPro server, providing the same information and functionality as the main server, for AnthillPro users.

E

Empty-value properties	enable users to set a standard for the available attributes a user can provide when executing a resource. Using an empty-value property gives users the option to

	define a required value or simply pass a blank value that AnthillPro will effectively ignore.
environment	An environment is a user-definded collection of one or more resources that host an application. At least one environment must be associated with the process before the process can be executed.
Environment	An environment is a partition grid of agents that is specific for different stages of a project life-cycle.
Environment Group	determines the set of environments that project workflows may be executed on. Each environment group must contain at least one environment.
environment inventory	An environment inventory tells you both what versions of any given component is running on a particular resource.
Environment Property	Environment Properties are used to set default values for a particular environment.
Environment Security Report	An environment security report provides information about user roles and privileges defined for environments.
Event Triggers	Event Triggers use scripts to create an Event Filter that listens to events passing through the AnthillPro Event Service. When the Event Script detects an event, it can then trigger another action by the AnthillPro server.

F

Fixed Agent Filter	Fixed Agent Filters always select a specific agent within the environment. If the requested agent is locked or can't receive more work, the request will be queued until the agent frees up.
full version	A full version contains all component artifacts.

G

Guest User Account	The Guest User Account gives anonymous access to AnthillPro, and does not require a user name or password at login.

I

IDE Plug-ins	The plug-ins enable developers to view the current activity and state of projects, start new builds, map their projects to projects in AnthillPro, and resolve the project's dependency artifacts.
incremental version	An incremental version contains only artifacts that have been modified since the previous version was created.
Independent Scheduling	Related projects do not need to know about each other for independent scheduling. A dependent project starts each build by gathering the most recent artifacts from its dependencies.
Internal Projects	Allows for dependency relationships between concurrently developed AnthillPro projects.

Interval Schedule	Regularly fires after a fixed interval of time.

J

Job	A job is a series of steps detailing how to get something done.
Job-execution Property	Use the Job-execution property type if the value, default value or options are to be generated by a job that executes on an agent.
Job Iteration Property	Job iteration properties are configured after iterating a job. They are typically used to set parameters for a single job that is run many times, with a slightly different parameter each time.
Job Iteration	Job Iterations are used to run the same job multiple times.
Job Wizard	A Job Wizard assists in the creation of build jobs. The Wizard takes you through the steps to configure the builder and the publisher required by the job, and is the best way to ensure your build job will be configured successfully. The Build Life is a map of (1) what occurred during the build; (2) the processes that were performed on the generated artifacts; (3) where the build artifacts ended up.

L

Library Jobs	A Library Job is a way for you to create template jobs. Once a Library Job has been configured, it can be used by multiple projects or can be turned into a project-specific job.
Library Workflow	The Library Workflow can only use library jobs as part of its definition.
Life-Cycle Build	The Life-Cycle Build provides a wealth of information and visibility into the build and release cycle. It is defined by its Life-Cycle Model, Environments, and Build workflow.
Life-Cycle Model	The Life-Cycle Model provides a template for managing the dependencies, artifacts, deployments, etc, associated with every build of the project. They are reusable, allowing you to apply the same standards to any similar project.
LDAP	A lightweight directory access protocol (LDAP) is a widely used protocol used to access distributed directory information over the internet protocol (I.P) networks.
lock	A lock is routinely used to ensure that processes do not interfere with one another.
Lockable Resource	A Lockable Resource specifies a resource that might be used by multiple workflows, gives it a name and forces arbitrary workflows to run one at a time or in small groups.

N

Network Settings	Network Settings are used to configure communication between the AnthillPro server and agents.
Notification	Notifications inform the recipient of the status of a CI build.
notifications	Notifications play a role in approving deployments: IBM uDeploy can be configured to send out an e-mail to either a single individual or to a group or people

	(based on their security role) notifying them that they need to approve a requested deployment.
Notification Scheme	Notification Schemes determine who to notify of build status, what conditions to notify them on, and what mechanism to notify them with. sets rules determining what groups of users are sent which kind of notification about specific events.
notification scheme	A notification scheme enables IBM uDeploy to send out notifications based on events. For example, the Default Notification scheme will send out an e-mail when an Application Deployment fails or succeeds.
Notification Template	Notification Templates are Velocity templates that take information about the build and produce a document.

O

Operational Project	Operational Projects provide a simplified interface that allows AnthillPro users to execute ancillary tasks not easily run during a build, deployment, etc., workflow.

P

Permission	Permissions associate the role, resource, and an action that may be performed on the resource. Typical actions include the ability to read or view the resource, the ability to write to or modify the resource, the ability to modify the security settings for a resource, or the ability to execute the resource.
plug-ins	A plug-in is the integration with third-party tools.
Preflight Build	Preflight Builds allow developers to run a test build in the authoritative build environment before committing their changes to source control.
process	Processes play a coordination role. They are authored using a visual drag-n-drop editor.
process editor	A process editor is a visual drag-and-drop editor that enables you to drag process steps onto the design space and configure them as you go.
processing property	A processing property is a way to add user-supplied information to a process.
Project Environment Property	Project Environment Properties are automatically placed as environment variables for all commands run in the target environment.
Project Property	Project Properties are used for all workflows regardless of the target environment.
Property	Properties identify various properties that must be specified when the workflow is executed. Properties are used to manage variables passed into commands, agent filters, and custom stamping algorithm templates.
proxy resource	A proxy resource is a resource effected by an agent on a host other than the one where the resource is located.
Pull Build	A Pull Build will build every dependency that is out of date prior to building the top level of a project.
Pure Build	Pure builds take the source as input and transforms it into artifacts. In AnthillPro, it generates the Build Life.

Push Build	Push Builds perform the minimum number of builds, in the correct order, to ensure that a change to a component does not break anything that depends on it directly or transitively. A push build can become resource intensive.
Push Scheduling	Push Scheduling are used when an originating workflow builds, the workflows that depend on it automatically build.

Q

Quiet Periods	Quiet Periods are configured on the project, and play an integral part in ensuring that the source code AnthillPro obtains from the SCM contains a consistent set of changes. The quiet period is a measure of time that must pass without anyone committing changes to the source.

R

Read Permission	Read permissions allow a user to resolve/download the artifacts associated with a Build Life.
relay servers	A relay server enables network-to-network communication.
remote agents	A remote agent is an agent that will communicate with the server via an agent relay.
Report	Reports provide information about system activity.
Repository Trigger	Repository Triggers are used for Continuous Integration builds. Once the trigger is active for a workflow, every commit of source changes will initiate a build.
resource	A resource is a user-defined construct based on IBM uDeploy's architectural model. A resource represents a deployment target.
resource group	A resource group is a group of resources used to help organize and manage the agent installed in a different environment.
Resource Security Report	A resource security report provides information about user roles and privileges defined for resources.
role	A role enables you to further refine how a resource is utilized, and is similar to subresources.

S

Schedule	A Schedules determines when events such as builds, cleanups, backups, etc., are automatically run by the system. Once a schedule has been created, it may be used by many different AnthillPro resources.
Schedule Trigger	A Scheduled Trigger runs on its own timer that pools the SCM for changes. If changes are detected, the build is registered with the schedule and kicked off when the schedule fires. Scheduled Triggers give the option to force a build regardless of whether source changes have occurred.
schema	A schema is a visual representation of the different parts of IBM uDeploy that may be secured. Each Schema interacts with users indirectly, through the role.

Script Library	the script library is used to create, organize, and provide security around often-used AnthillPro scripts. The Script Library is most helpful for large organizations, allowing them to ensure that only the appropriate team members can modify a script.
Scripted Property	Scripted value properties are properties where the value, default value or options are generated by a script that is executed before workflow execution.
Scripted Agent Filters	A scripted agent filter selects agent based on an agent filter script.
SSL	A secure socket layer (SSL) enables clients and servers to communicate securely by encrypting all communications.
Security Permission	Security Permissions allows users to determine who can set security for the artifact set.
Security Reports	A security report provides information about user roles and privileges.
Security Setting	A Security Setting is used to configure access to the Anthill Server setting for configuration and artifact management.
Server Proxy	Server proxies enable you to access servers/repositories that are on external networks.
snapshot	A snapshot is a collection of specific component versions, usually versions that are known to work together.
Source Configuration	Source Configuration identifies exactly which source code should be retrieved from a repository.
Stamp	A Stamp is a configurable name for the build number, build identifier, or version number used to identify a build. This allows you to mimic your strategies.
Stamp Context Script	Stamp Context Scripts generate values for variables in the stamp context when creating a stamp (for builds, etc.).
Stamp Mapping	For every stamp style in the project's stamp style group, one must specify what stamping strategy will be used. A typical stamp style group might specify for development builds to a version number and strategy for incrementing that number.
Stamp Style	Stamp Styles are used to apply different stamps to a single build, and allow you to help track a build throughout its life-cycle.
Stamp Style Group	A Stamp Style Group creates common names for types of stamps.
Status Group	A Status Group is a set of common names for statuses.
stateless	Stateless means the server retains little session information between requests, and each request contains all information needed to handle it.
subresource	A subresource enables you to apply logical identifiers or categories within any given group.
switch step	A switch step enables you to create conditional processes.
System Tray Monitor	The System Tray Monitor provides feedback directly to the desktop, without having to open or refresh a browser.

System Property	The System property is used to set default values for a particular property for all workflows and project system wide.

T

Template Reports	Template Reports are good for generating tabular data in one or more types of output.
Trigger	A Trigger is an automated mechanism for kicking off a workflow.

U

uncontrolled environment	A uncontrolled environment is an environment that does not contain approvals approval gates.
user impersonation	IBM uDeploy can use user impersonation when an agent must execute a command for which it might not otherwise have permission.

V

Velocity Report	Velocity Reports are good for customizing the AnthillPro UI.
version	A version is set each time a component changes. There are two types of versions a full version and an incremental version.

W

Workflow Definition	A Workflow Definition defines which job should be run, species the order of jobs, as well as the elements to be run in parallel.
Workflow Priorities	Workflow Priorities allow you to determine the order in which workflows run. Use workflow priorities to determine which workflow will run first.
Workflow Property	Workflow properties are used to send a build to a particular platform when writing native code for multiple platforms.
Workflow Request Context	A workflow request context is a collection of requests for workflows that are processed together.
Workflow Task	A Workflow Task allow you to set up manual gates, etc., that must be performed.
Workflow Tool	There are two major categories of work flow tools: priorities and requests.
Workflow Request	The Workflow Request is the first action taken by the server when executing an originating or secondary workflow.

Index

Symbols

A

B

C

D

E

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W