opentext™

OpenText™ Documentum™ Content Management

Advanced Workflow Installation Guide

Install and configure Advanced Workflow.

EDCPKLPR250400-IGD-EN-1

OpenText™ Documentum™ Content Management
Advanced Workflow Installation Guide
EDCPKLPR250400-IGD-EN-1
Rev.: 2025-Nov-24

This documentation has been created for OpenText™ Documentum™ Content Management CE 25.4.
It is also valid for subsequent software releases unless OpenText has made newer documentation available with the product, on an OpenText website, or by any other means.

Open Text Corporation

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111
Toll Free Canada/USA: 1-800-499-6544 International: +800-4996-5440
Fax: +1-519-888-0677
Support: https://support.opentext.com
For more information, visit https://www.opentext.com

© 2025 Open Text

Patents may cover this product, see https://www.opentext.com/patents.

Disclaimer

No Warranties and Limitation of Liability

Every effort has been made to ensure the accuracy of the features and techniques presented in this publication. However, Open Text Corporation and its affiliates accept no responsibility and offer no warranty whether expressed or implied, for the accuracy of this publication.

Table of Contents

Chapter 1

Getting started

1.1 Installation environment overview

Installation of OpenText™ Documentum™ Content Management (CM) Advanced Workflow components require installation of both the OpenText Documentum CM components and the additional infrastructure hardware and software prerequisites. For detailed information about hardware and software requirements, see *OpenText Documentum Content Management Release Notes*.

Note: The information in this guide is for application and system administrators who install and administer the Advanced Workflow. It assumes familiarity with basic Documentum functionality, such as repository configuration.

Common OpenText Documentum CM components include:

- Documentum Administrator (DA)
- Deployment Agent (xDA)
- JDK
- OpenText™ Documentum™ Content Management Process Engine
- OpenText™ Documentum™ Content Management Process Integrator
- OpenText™ Documentum™ Content Management Server

1.2 Installation methods

An Advanced Workflow installation environment contains the OpenText Documentum CM components and third-party software needed to run an Advanced Workflow application.

You can build environments only through the manual mode. You can build an Advanced Workflow physical or virtual installation environment by manually installing the OpenText Documentum CM components and third-party software. Use xDA to install Advanced Workflow applications into the manually-built environment. "Manual provisioning" on page 10 provides more details.

The term *OpenText Documentum software components* refers to the Advanced Workflow programs listed in "Installation environment overview" on page 7.

1.3 Managing environments

1. To provision the environments manually, install the components, create application server homes or instances, install xDA, and register the environment with xDA. "Manual provisioning" on page 10 provides details.
2. Install the applications. "Overview of installing an Advanced Workflow application" on page 63 provides details.

1.4 Security configuration

This guide provides information on the following security configurations:

Security configuration	Details or instructions
Password Encryption Utility	Using the Password Encryption Utility
SSL	Importing SSL Certificates to Support Encrypted Connections for Process Activities
Service Endpoint	Service Endpoint Configuration
Process Integrator	Installing Process Integrator Inbound Services
Web Services	Configuring an HTTP Proxy Server for Web Services

Chapter 2

Manually provisioning an Advanced Workflow environment

2.1 Overview

You provision the environments manually and use xDA to install Advanced Workflow applications in those environments. You must have valid authorization through OTDS to provision environment for Advanced Workflow. xDA is the lightweight application used for application installation and endpoint resolution in a manually created environment. The main purpose of xDA is to allow registration of manually provisioned environment and install Advanced Workflow applications to that environment.

To register an environment with xDA, you use **xDA Management Center** or **create-environment** xDA tools command and you can install the Advanced Workflow applications using Advanced Workflow or xDA Tools:

- *xDA Management Center* is a web user interface utility. The xDA Management Center has a set of default environment templates based on module template files, which are same as the blueprint files. You can create additional environment templates to meet your needs. The xDA catalog contains these environment templates.

  You use default or custom environment templates to register manually provisioned environments. The xDA Management Center enables you to configure various manually installed Advanced Workflow components. xDA uses this information to install Advanced Workflow application to these environments.

- *xDA Tools* is a command-line interface (CLI). It connects to xDA to perform validations and install Advanced Workflow application. You can xDA tools to populate the xDA environment initially.

Installing an Advanced Workflow application includes configuring application installation parameters and endpoints, and using xDA to install the application. xDA deploys repository and application server artifacts to its proper locations in the environment. Additionally, it performs necessary application configuration based on rules in the application package.

2.2 Manual provisioning

You provision an Advanced Workflow environment manually.

1. Download and install Documentum components and third-party software into a physical or virtual environment. "Installing and configuring Advanced Workflow components and third-party software" on page 10 provides instructions.
2. Configure the Advanced Workflow application host. The Advanced Workflow application host is the application server that you intend to use for installing the Advanced Workflow applications.
3. Install the xDA. "Installing the xDA" on page 31 provides instructions.
4. Register the environment with xDA. "Registering an environment" on page 41 provides instructions.
5. When the status of the environment is Registered, synchronize the environment. "Synchronizing a manually provisioned environment" on page 48 provides instructions.

After the environment has been successfully registered and synchronized, you can install applications to the environment. "Overview of installing an Advanced Workflow application" on page 63 provides details.

2.3 Installing and configuring Advanced Workflow components and third-party software

1. Install each of the components listed in the following table, in the order listed, into a physical or virtual environment. Download all components from Opentext My Support (support.opentext.com) site, except third-party software.

   Note: Advanced Workflow requires all the components unless otherwise indicated. If you miss to install any one of them, the application installation fails.

Component	Required/ Optional	Source of information
JDK (third-party component) **Note:** JDK 17 is supported at design time.	Required	JDK documentation
OpenText Documentum Content Management (CM) Server and OpenText™ Documentum™ Content Management Foundation Java API	Required	*OpenText Documentum Content Management - Server and Server Extensions Installation Guide (EDCSY250400-IGD)*

Component	Required/ Optional	Source of information
Documentum Administrator (DA)	Required	*OpenText Documentum Content Management - Server and Server Extensions Installation Guide (EDCSY250400-IGD)*
Process Engine	Required	"Installing Process Engine" on page 12
Application server (third-party component)	Required	Respective third-party component documentation
Process Integrator (BPS)	Required **Note:** Process Integrator is an optional service that is required only if you are using inbound activities.	"Installing Process Integrator" on page 16
xDA	Required	"Installing the xDA" on page 31
xDA Tools	Required	
Advanced Workflow	Required	"Installing Advanced Workflow" on page 62

2. Record the following environment information:

   - Application server URL with username and password
   - Docbroker administrator username and password
   - Docbroker name
   - Docbroker port
   - Docbroker host (IP address or host name)
   - DA URL

For more information to deploy applications using xDA, see "Application installation using xDA tools" on page 64. Alternatively, provide this information to developers using Advanced Workflow to design applications.

2.4 Advanced Workflow deployment scenarios

You can deploy Advanced Workflow components across one or more physical or virtual server machines. The two-machine deployment scenario is the preferred development setup to maintain optimal performance but you can use fewer application server instances or hosts as per your own requirements.

This section describes a two-machine deployment scenario that serves as a development environment. In this example environment, the application server is Tomcat server.

Machine	Install the following	On the Application server
Server Machine	• JDK • Documentum CM Server and its components • Application server • xDA	Create two application server instances: • Instance 1 – deploy AppHost application • Instance 2 – deploy Process Integrator
Client Machine	• Advanced Workflow	

2.5 Installing Process Engine

Before installing Process Engine, make sure that the Java Method Server (JMS) service is stopped and the Documentum CM Server repository is running.

2.5.1 Starting the connection broker and all repositories

1. Select **Start** > **Programs** > **Documentum Server Manager**.
2. In the **Documentum Server Manager** dialog box, click **DocBroker**, select a connection broker, and click **Start**.
3. Click **Docbase**, select a repository, and click **Start**.
4. Click **OK**.

2.5.2 Preparing non-Windows Platforms for Process Engine installation

Before you install Process Engine on Linux platforms, source the script dm_set_server_env.csh or script dm_set_server_env.sh (depending on the shell in which you run), in the owner environment for the Documentum CM Server installation. This sets the environment variables required by the installer. The script is located at `$DM_HOME/bin`. The following example shows the script for a C shell:

source ./dm_set_server_env.csh

The *OpenText Documentum Content Management - Server and Server Extensions Installation Guide (EDCSY250400-IGD)* provides more information on manually configuring environment variables.

2.5.3 Process Engine installation

2.5.3.1 Prerequisites

If you are upgrading Process Engine from previous releases to up to 25.4 release, complete the following steps before proceeding with Process Engine upgrade:

Note: This section is applicable only for Process Engine upgrades to releases prior to 25.4.

1. Stop the Documentum Java Method Server.
2. Backup the existing `\\Documentum\tomcat\webapps\bpm` folder for future reference.
3. Install the Process Engine.
4. Restart the Java Method Server.

2.5.3.2 Installing Process Engine with backward compatibility

To ensure backward compatibility with Advanced Workflow versions 24.4 and 25.2, complete the following steps:

1. Go to `\\Documentum\tomcat x.x.x\webapps\` folder and back up the existing `bpm` folder to a temporary location.
2. Delete the `\\Documentum\tomcat x.x.x\webapps\bpm` folder.
3. Proceed to install the Process Engine version 25.4.

2.5.3.3 Installing Process Engine

1. Download the Process Engine software from the OpenText MySupport (https://support.opentext.com) site.
2. Locate the file for Windows, `Process_Engine_win.zip` and extract the file.
3. Browse to the extracted folder.

   For example, open the `Process_Engine_win` folder.
4. Run the installation file, `peSetup.exe`.
5. Read the Software License and Maintenance agreement page.
6. To continue with the installation, click **I accept the terms of the license agreement** and click **Next**.
7. The installer displays a list of repositories where you want to install Process Engine. Click **Next**.

   Make sure you have started all the repositories where you want to install Process Engine. If you do not start the repositories, the installation fails when the installer unpacks the DAR files and tries to install them.
8. The system prompts you for the application server credentials. Enter your credentials, confirm, and click **Next**.
9. The installer displays the installation location. To continue with the installation, click **Install**.

   The installer installs Process Engine on all repositories that are served by Documentum CM Server. This installs the `bpm.war` file on Java Method Server (JMS) and installs the DAR files on each repository.

   The Process Engine installer extracts the contents of the `bpm.war` file to `%DOCUMENTUM%\tomcat\webapps\` on the Microsoft Windows platform and creates the following folder structure:

   - `bpm.war` folder containing the web application files

   On the Windows platform, the installer creates the `PE` folder at `%DOCUMENTUM%\`.
10. Click **Done** to close the Process Engine Installation wizard.
11. To verify that the Process Engine installation is successful, open the following URL in a browser:

    `http://<host>:<port>/bpm/servlet/DoMethod`
12. To check the Process Engine version, open the following URL in a browser:

    `http://<host>:<port>/bpm/modules.jsp`

    where <*host*> is the host name or IP address of the JMS and <*port*> is the port number of the JMS.

Note: If the installation fails, view the installation error details in the `logs` folder created inside the Process Engine installation folder.

2.5.3.4 Installing Process Engine in silent mode

To install the Process Engine in silent mode, perform these steps:

1. Specify the following parameters—username, password, and domain name in the `test.properties` file.

```
INSTALLER_UI=silent
PROCESS_ENGINE.GLOBAL_REGISTRY_ADMIN_USER_NAME=AdminUserName
PROCESS_ENGINE.GLOBAL_REGISTRY_ADMIN_DOMAIN=domainName
PROCESS_ENGINE.SECURE.GLOBAL_REGISTRY_ADMIN_PASSWORD=AdminPassword
```

2. Specify the port number and password for the application host server.

```
APPSERVER.SERVER_HTTP_PORT=9080
APPSERVER.SECURE.PASSWORD=jboss
```

3. Specify the location of Documentum or IJMS, domain name, administrator name, and password:

```
\\Used for Default JMS / Independent Java Method Server \\ Location of DCTM
PE.INSTALL_TARGET= C:\\Documentum
\\Location of IJMS
PE.INSTALL_TARGET= C:\\IJMS
PE.FQDN= fully_qualified_domain_name
PE.DOCBASES_ADMIN_USER_NAME=username
PE.DOCBASES_ADMIN_USER_PASSWORD=password
```

4. If the vault service is enabled, add the following properties:

```
\\ DSIS_URL consists of the ip and port on which the dsis service is running.
\\ DSIS_TOKEN is the token generated from dsis service
IS_VAULT_ENABLED=true
DSIS_URL=http://localhost:8200/dsis
DSIS_TOKEN= <vault_token>
```

5. Invoke the Process Engine silent installation using the following command:

```


//For Windows
peSetup.exe -f c:\test.properties
//For Linux
peSetup.bin -f /home/test.properties
```

2.5.4 Adding Process Engine to a new repository

If you configure a new repository on Documentum CM Server after installing Process Engine, the repository configuration program automatically installs the DAR files. You do not have to install Process Engine for this new repository.

2.5.5 Installing Process Engine in an environment with multiple Documentum CM Servers

1. Make sure all the connection brokers, Documentum CM Servers, and repositories for which Process Engine must be installed are in the running state.

   "Starting the connection broker and all repositories" on page 12 and the *OpenText Documentum Content Management - Server Administration and Configuration Guide (EDCCS250400-AGD)* explain how to start connection brokers and all the repositories.

2. Perform the following for each Documentum CM Server:

   a. Install Process Engine on all Documentum CM Servers.

      "Process Engine installation" on page 13 provides more information.

   b. Configure the Workflow Agent threads according to your sizing requirements.

      Set the Workflow Agent threads to zero for other Documentum CM Servers that are not participating in processing activities.

   c. Configure the Workflow Agents.

      The Workflow Agent controls the execution of automatic activities in a workflow. The *OpenText Documentum Content Management - Server Administration and Configuration Guide (EDCCS250400-AGD)* explains how to configure Workflow Agents.

      For the distributed Documentum CM Server setup, it is recommended that you manually approve the Foundation Java API privileged client on the secondary Documentum CM Server.

2.6 Installing Process Integrator

This section provides instructions for installing Process Integrator into an application server.

2.6.1 Understanding Process Integrator inbound services

Process Integrator provides inbound messaging capabilities to applications based on the Documentum platform. This capability enables the applications to exchange documents and information with people outside the organization.

You can install and configure the `bps.war` file into an application server to enable use of the delivered Process Integrator inbound messaging process activities. When you extract the `bps.war` file, you find these two files available in the `bps/WEB-INF/classes/` folder:

- `bps_template.xml`

  Note: If the vault service is enabled, add the following properties to `bps_template.xml`:

```
</xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<polling_interval>40</polling_interval>
<message_store_home_dir>C:/BPS/message_store<message_store_home_dir>
<instance_name>bps</instance_name>
<ha_enabled>FALSE</ha_enabled>
<config_properties>
<property name="mail.imap.partialfetch" value="false"/>
<property name="mail.debug" value="false"/>
</config_properties>
<connections>
<docbase-connection>
<docbase>docbasename</docbase>
<user>username</user>
<password>vault_secret|vault_key</password>
<domain/>
</docbase-connection>
</connections>
</config>
```

- `dfc.properties`

Note: If the vault service is enabled, add the following properties to `dfc.properties`:

```
dfc.dsis.enabled=true
dfc.dsis.daemon.url=http://localhost:8200/dsis
dfc.dsis.daemon.token=<VAULT TOKEN ID>
dfc.globalregistry.repository=<docbasename>
dfc.globalregistry.username=<dm_bof_registry>
dfc.globalregistry.password=<VAULT GLOBALREGISTRY PASSWORD/docbasename>
dfc.session.allow_trusted_login=false
```

Configure the `bps_template.xml` and `dfc.properties` files to use the inbound services. The following sections describe how to configure these files.

2.6.2 Installing Process Integrator inbound services

1. Download the Process Integrator software.

   You must have received instructions through email regarding how to download products from My Support (support.opentext.com) site.

2. Install the application server and create an instance. The application server documentation provides installation instructions and information on creating an application server instance.

3. Locate the `bps.war` file and extract it to a temporary folder for configuration using the following command:

   ```
   jar -xvf bps.war
   ```

4. Edit the `bps_template.xml` file.

   a. Locate `bps_template.xml` within `bps.war\WEB-INF\classes\`. You edit this file to specify the repository connection information. This template contains sample information that you can use as a guide when adding the connection information for your specific repository.

b. You can encrypt the password in the `bps_template.xml` file using the Password Encryption utility. "Using the password encryption utility" on page 22 provides information on encrypting the password.

c. Remove the comments in the `bps_template.xml` file after you modify it. Make sure the `bps_template.xml` file is a well-formed XML file.

Example:

```
<?xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <polling_interval>300</polling_interval>
   <message_store_home_dir>C:/BPS/message_store</message_store_home_dir>
   <instance_name>bps</instance_name>
   <ha_enabled>FALSE</ha_enabled>
   <config_properties>
         <property name="mail.imap.partialfetch" value="false"/>
         <property name="mail.debug" value="false"/>
   </config_properties>
   <connections>
         <docbase-connection>
               <docbase> BPM_Inbound_D6</docbase>
               <user> tuser1</user>
               <password> tuser1</password>
               <domain/>
         </docbase-connection>
   </connections>
</config>
```

Polling_interval is the time interval (in seconds) after which the Process Integrator framework polls for new activity endpoints (inbound listeners). The default specified in the file is 300 seconds (5 minutes). Use a large value such as 3600 in a production environment where new endpoints are not created frequently.

5. Edit the `dfc.properties` file.

The `dfc.properties` file is part of the extracted `bps.war` file. Edit the `dfc.properties` file to specify the correct connection broker information and the correct `dfc.data.dir\`. Add any other properties to `dfc.properties`.

a. Add the fully qualified host name for the connection broker to the following key. You can add backup hosts by incrementing the index number within brackets.

`dfc.docbroker.host[0]=`*`<host_name>`*

b. If you want to use a port for the connection broker other than the default of 1489, add a port key to the dfc.properties.

`dfc.docbroker.port[0]=`*`<port_number>`*

c. Add the global registry repository name to the following key:

`dfc.globalregistry.repository=`*`<repository_name>`*

d. Add the key for the location of Foundation Java API configuration files:

`dfc.data.dir=`*`<C|:Documentum>`*

e. Add the username of the dm_bof_registry to the following key:

`dfc.globalregistry.username=`*`<dm_bof_registry_user_name>`*

The global registry user, who has the username of dm_bof_registry, has read access to objects in the /System/Modules and /System/NetworkLocations only.

f. Add an encrypted password value for the following key:

```
dfc.globalregistry.password=encrypted_password
```

Copy the username and encrypted password from the `dfc.properties` file on the global registry Documentum CM Server host. You can also select another global registry user and encrypt the password using the following command from a command prompt (assumes the directory containing `javaw.exe` is on the system path):

```
java -cp dfc.jar com.documentum.fc.tools.RegistryPasswordUtils
<password_to_be_encrypted>
```

6. Save the `bps_template.xml` and `dfc.properties` files.

7. Remove the unmodified `bps.war` file from the temporary folder in step 3 and package `bps.war` using the following command:

```
jar -cvf bps.war *
```

8. Deploy the WAR file into the application server.

The first time `bps.war` is installed, the system reads the `bps_template.xml` file and copies it to `<dfc.data.dir>/config/bps/` folder as `bps.xml`. The repository password is encrypted in the `bps.xml` file in `<dfc.data.dir>/config/bps/` folder.

The system uses the `bps.xml` file for retrieving configuration details for message processing. If there are any changes to configuration details, update the `bps.xml` file with current configuration details. Next, restart Process Integrator.

2.6.2.1 Installing Process Integrator Inbound Services on Tomcat application server

Add the following code to `catalina.bat` file for Tomcat application server on Window environment:

```
set JAVA_OPTS=-Xms1024m -Xmx2048m -XX:MaxPermSize=512m -XX:+UseParallelOldGC -Xdebug -
Xnoagent -Xrunjdwp:transport=dt_socket,server=y,suspend=n

set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.lang=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.lang.invoke=ALL-
UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.net=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.lang.ref=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.provider=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/sun.security.pkcs=ALL-
UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/sun.security.x509=ALL-
UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/sun.security.util=ALL-
UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.tools.keytool=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.naming/
com.sun.jndi.toolkit.url=ALL-UNNAMED"
```

```
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.xml.cypto/
com.sun.org.apache.xml.internal.security=ALL-UNNAMED"
```

Add the following code to `catalina.sh` file for Tomcat application server on Linux environment:

```
export JAVA_OPTS='--add-opens=java.base/java.lang=ALL-UNNAMED
--add-opens=java.base/java.lang.invoke=ALL-UNNAMED"
--add-opens=java.base/java.net=ALL-UNNAMED"
--add-opens=java.base/java.lang.ref=ALL-UNNAMED"
--add-exports=java.base/sun.security.provider=ALL-UNNAMED"
--add-exports=java.base/sun.security.pkcs=ALL-UNNAMED"
--add-exports=java.base/sun.security.x509=ALL-UNNAMED"
--add-exports=java.base/sun.security.util=ALL-UNNAMED"
--add-exports=java.base/sun.security.tools.keytool=ALL-UNNAMED"
--add-opens=java.naming/com.sun.jndi.toolkit.url=ALL-UNNAMED"
--add-opens=java.xml.cypto/com.sun.org.apache.xml.internal.security=ALL-UNNAMED'
```

2.6.2.2 Installing Process Integrator inbound services on WildFly application server

Download the latest BPS WAR files for WildFly application server from My Support (support.opentext.com) site. Follow these steps to configure BPS on WildFly Server:

1. Create a folder structure as follows:

   <application_server_home>\modules\system\layers\base*<Subfolder1>**<subfolder2>*\main

2. Create a `module.xml` file in the *<WildFly-Home>*\modules\system\layers\base*<Subfolder1>**<subfolder2>*\main\ folder location:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <module xmlns="urn:jboss:module:1.1" name="com.bps">
   <resources>
   </resources>
   </module>
   ```

3. Add the module created in the previous step as global-module in `<application_server_home>\standalone\configuration\Standalone.xml` file as follows:

   ```
   <subsystem xmlns="urn:jboss:domain:ee:4.0">
   <global-modules>
   <module name=”com.bps”/>
   </global-modules>
   </subsystem>
   ```

4. Add this content in the `/sun/jdk/main/module.xml` file:

   ```
   <path name="com/sun/rowset/internal"/>
   ```

5. Add the following Java Options properties in the `standalone.conf.bat` file. For Linux, update the `standalone.conf` file:

   ```
   set "JAVA_OPTS=-Xms4096m -Xmx4096m -XX:MetaspaceSize=2048M -
   XX:MaxMetaspaceSize=2048m -Djava.awt.headless=true
   --add-opens=java.base/java.lang=ALL-UNNAMED
   --add-opens=java.base/java.lang.invoke=ALL-UNNAMED"
   --add-opens=java.base/java.net=ALL-UNNAMED"
   --add-opens=java.base/java.lang.ref=ALL-UNNAMED"
   --add-exports=java.base/sun.security.provider=ALL-UNNAMED"
   --add-exports=java.base/sun.security.pkcs=ALL-UNNAMED"
   --add-exports=java.base/sun.security.x509=ALL-UNNAMED"
   ```

```
--add-exports=java.base/sun.security.util=ALL-UNNAMED"
--add-exports=java.base/sun.security.tools.keytool=ALL-UNNAMED"
--add-opens=java.naming/com.sun.jndi.toolkit.url=ALL-UNNAMED"
--add-opens=java.xml.cypto/com.sun.org.apache.xml.internal.security=ALL-UNNAMED"
```

6. (Optional) Update the `dfc.properties` of Application.war by adding the following key:

```
dfc.security.keystore.file=<Any Physical Directory Location>\dfc.keystore
```

Note: The `keyStore` folder is not created automatically, you have to create it manually.

2.6.2.3 Installing Process Integrator inbound services on WebSphere Liberty application server

Download the latest BPS WAR files for WebSphere Liberty application server from My Support (support.opentext.com) site. Follow these steps to configure BPS on Liberty Server:

1. Install the WebSphere Liberty application server.
2. Create a server and configure `server.xml` file as described:

```
<?xml version="1.0" encoding="UTF-8"?>
 <server description="new server">
 <!-- Enable features -->
 <featureManager>
 <feature>webProfile-11.0</feature>
  </featureManager>
....
....
....
....
 <remoteFileAccess>
 <writeDir>${server.config.dir}/dropins</writeDir>
 </remoteFileAccess>
 <library id="awfpropeties">
 <fileset dir="${server.config.dir}/lib/global" includes="*.properties" />
 </library>
 <!-- Automatically expand WAR files and EAR files -->
 <applicationManager autoExpand="true"/>
 </server>
```

   a. Create a folder, `lib\global\`, at the application server home directory. For example:

      <<application_server_home>\lib\global\>

   b. Create the `dfc.properties` file for the application to reference the repository and copy the `dfc.properties` file from the `%Documentum%\Config\` folder on the Documentum CM Server to the specified location.

   c. (Optional) Update the `dfc.properties` file by adding the following key:

```
dfc.security.keystore.file=<location of dfc.keystore file>\dfc.keystore
```

3. Copy the `log4j2.properties` file to the `application_server_home>\lib\global\` folder.

4. Before starting the WebSphere Liberty application server in a Windows environment, add the following jvm.options properties in the *<application_server_home>**<server_name>*\bin\server.bat:

```
-Xms512m
-Xmx1024m
-XX:MaxPermSize=512m
-Dlog4j2.configurationFile=<application_server_home>\lib\global\log4j2.properties
set JAVA_HOME="<java_17_home>\jdk-17.0.12"
set JAVA_ARGS="-Dcom.ibm.jsse2.overrideDefaultTLS=true"
set "JAVA_OPTS=-Dprogram.name=%PROGNAME% %JAVA_OPTS%"
set "JDK_JAVA_OPTIONS=-Dprogram.name=%PROGNAME% %JDK_JAVA_OPTIONS%"
set "JAVA_OPTS=%JAVA_OPTS% --add-opens=java.base/java.lang=ALL-UNNAMED"
set "JAVA_OPTS=%JAVA_OPTS% --add-opens=java.base/java.io=ALL-UNNAMED"
set "JAVA_OPTS=%JAVA_OPTS% --add-opens=java.base/java.util=ALL-UNNAMED"
set "JAVA_OPTS=%JAVA_OPTS% --add-opens=java.base/java.util.concurrent=ALL-UNNAMED"
set "JAVA_OPTS=%JAVA_OPTS% --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.net=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.lang.ref=ALL-
UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.naming/
com.sun.jndi.toolkit.url=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.provider=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.pkcs=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.x509=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.util=ALL-UNNAMED"
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-exports=java.base/
sun.security.tools.keytool=ALL-UNNAMED
set "JDK_JAVA_OPTIONS=%JDK_JAVA_OPTIONS% --add-opens=java.base/java.lang=ALL-UNNAMED
```

5. Download and deploy the WebSphere specific `bps.war` file by copying the WAR file to *<application_server_home>*\dropins\ folder.

6. At the *<application_server_home>*\bin\ command prompt and start the WebSphere Liberty application server:

7. Browse to the Process Integrator Server URL to verify that the process integrator server is up and running:

```
http://<hostname>:<port>/bps/lsnrs.jsp
```

2.6.3 Using the password encryption utility

If your security policies require encrypting the authentication password found in `bps_template.xml`, use the delivered script located in the `bpsInboundTools\` archive folder.

1. After editing the `bps_template.xml` file to include the clear-text username and password, save and close the file.

2. Locate e`ncryptBPSPassword batch/shell` script file.

   Navigate to the directory into which you unpacked the bpsInboundTools archive.

   For example, if you extracted `bpsInboundTools.zip` to the C drive, you can find the file in `C:\bpsInboundTools\bin\` and the required JAR files in `C:\bpsInboundTools\lib\`.

3. Open the `encryptBPSPassword batch/shell` script file and edit the properties to set the Foundation Java API configuration directory, the Java path, and the `bps_template.xml` file path.

   For example:

   ```
   set DFC_CONFIG_DIR=C:\Documentum\config
   set JAVA=C:\Program Files\Java\jdk<version>\bin\java
   set BPS_TEMPLATE_XML_FILE_PATH=C:\Tools\bps_template.xml
   ```

4. Execute the encryptBPSPassword batch/shell script.

   The script processes the `bps_template.xml` file and replaces the password with the string of the encrypted password.

2.6.4 Configuring inbound listeners

Inbound listeners for inbound process activities cannot use the same endpoint configuration. Avoid creating multiple inbound activities that point to the same endpoint configuration. To prevent listeners from looking for messages from the same endpoint, the system does not create listeners for multiple activities that point to the same endpoint.

Inbound activity type	Endpoint configuration parameters
Java Messaging Service (JMS)	Provider class, provider URL, connection factory, and queue/topic name.
Email	Email server host name, port, user, email server type, and folder.
HTTP	HTTP URL suffix.
FTP	FTP server host name, port, server type, base directory, inclusion and exclusion criteria.
Web Services	Target Name space Uniform Resource Identifier (URI), Port Type, and Operation Name.
Java Database Connectivity (JDBC)	JDBC URL and SQL Query.
Data Query Language (DQL)	Local or remote repository (true or false), repository name, and DQL query.

2.6.5 Configuring Process Integrator to support high availability

The high availability feature ensures the messaging system continues working without interruption in the event of a hardware or software failure.

Protocol listeners are assigned an active Process Integrator instance with other passive instances available, as well. When the active system fails, one of the other instances provides services for that listener ensuring continual inbound messaging services. The high availability feature must be enabled for each instance of Process Integrator.

1. Open the `bps_template.xml` file in the `bps.war` file.
2. Set the element `ha_enabled` to TRUE. For example:

   ```
   <ha_enabled>TRUE</ha_enabled>
   ```

3. Specify a unique name for the instance in the `bps_template.xml` file. For example:

   ```
   <instance_name>bps1</instance_name>
   ```

   If you do not specify an instance name, the system creates a default name, BPS_Instance when the Process Integrator inbound listener starts. If you are enabling high availability, each instance needs a unique instance name.

4. Repeat step 2 and step 3 for each instance of Process Integrator on the application server that uses high availability.

   The listeners do not automatically fail back to the first server when it comes up.

2.6.6 Setting the listener table attributes

The system creates an entry in the **dmc_bps_listener** table for every process with an inbound activity that is installed in the repository. For example, you may want the FTP listener to run on bps1 and the email listener to run on bps2. Either server can act as the failover server for the other listener.

There are two ways to set the <*preferred_instance name*> and <*run_in_all instances*> (boolean) attributes in the listener table: using DQL or using a script that is provided for Microsoft Windows systems.

2.6.7 Using DQL to set the listener table attributes

1. Find the r_object_id of the inbound activity located in the **dmc_bps_listener** table.
2. Run the following DQL command to set preferred_instance:

```
update dmc_bps_listener object set
preferred_instance='<preferred instance name>' where
listener_act_id='<objid of inbound activity>';
```

   For example:

```
update dmc_bps_listener object set
preferred_instance='bps1' where
listener_act_id='4c23cb538003f1b4'
```

   The **listener_act_id** in the table is the activity ID of the inbound activity.
3. Run the following DQL command to set <*run_in_all_instances*> to `TRUE`.

   By default, this Boolean attribute is false and so the listener can run only in one running Process Integrator instance.

```
update dmc_bps_listener object set run_in_all_instances=true
where listener_act_id='4c23cb538003f1b4';
```

2.6.8 Using a script to set listener table attributes

Note: Do not run listeners for asynchronous protocols on multiple Process Integrator instances. The run_in_all_instances value must remain 0 for asynchronous protocols listeners such as FTP, email, DQL, DB, and JMS. Two listeners must not pick up the same message.

1. Locate the setValuesInListenerTable script in the `bin` folder of the bpsInboundTools ZIP or TAR file.

   The script contains the following text:

```
echo off

REM ** Point the DFC_CONFIG_DIR to the same location
as the one pointed to by Inbound Framework **
REM ** Point JAVA to the java executable **

set DFC_CONFIG_DIR=
set JAVA=

echo on

"%JAVA%" -classpath %DFC_CONFIG_DIR%;..\lib
[[bpm_infra.jar]|[bpm_infra.jar]];..\lib[[bpm-commons.jar]|
[bpm-commons.jar]];..\lib[[bpmutil.jar]|
[bpmutil.jar]];..\lib\dfc[[dfc.jar]|[dfc.jar]]
[com.documentum.bps.utils.SetAttributesInListenerTable] %1 %2 %3 %4
```

2. In this script, set DFC_CONFIG_DIR to the directory containing the [dfc.properties]. Set JAVA to the Java executable directory.
3. Create the listener text file.

The template file `setValues_listener_template.txt`, located in the bin directory must be edited to contain the settings for the *<process name>*, *<activity name>*, *<preferred instance name>*, and *<run_in_all_instances>*.

The *<preferred instance name>* attribute is the name of the instance that is designated as the preferred instance for the listener. The *<run_in_all_instances>* attribute is set to `true` or `false`. If specified as `true`, the listener can run in any running instance. If not specified, the default is `false` and the listener can run only in one inbound instance.

The format for specifying the preferred instance and the run in all instances settings are:

```
#ProcessName::ActivityName::<preferred instance name>
::<run_in_all_instances>
#<process name>::<activity name>::<preferred instance name>
#<process name>::<activity name>::<preferred instance name>
::<run_in_all_instances>
#ex:
```

In each line of the text file, specify each inbound activity for which the preferred instance needs to be set.

For example:

```
#Process1::Activity1Inbound::bps1
#Process2::Activity2Inbound::bps1::true
```

Update the Preferred Instance column. The other columns are reserved for internal use only and must not be edited.

4. Save the `setValues_listener_template.txt` file and record the file path.

   The file path is used in the script as a parameter.

5. Complete the four parameters of the script.

   Set values for the repository name, the user, the password, and the full file path to the file specifying the attributes to be set.

6. Run the setValuesInListenerTable script.

   ```
   setValuesInListenerTable<docbase><user><password>
   <full path to the setValues_listener_template text file>
   ```

   Running the script populates the listener table for each activity with the values specified in the file.

2.6.9 Validating inbound services and listeners

After installing the web archive (WAR) file, use the validation URL to check that the inbound services and listeners are running. The validation URL provides access to the following information:

- Installed product versions
- Inbound listener details
- Inbound web services

1. Type the following URL into the browser:

   ```
   http://<host_name>:<port_number>/bps/inbound_details.jsp
   ```

   If the system shows an error page rather than the **Inbound Details** page, ensure that you have correctly entered the host name and port number in the URL. If the error persists, you may need to reinstall the WAR file.

2. On the **Inbound Details** page, click the link for the appropriate page:

   - **Click Here to View Versions**

     This page lists the versions of the different JAR files used in the inbound WAR file.

   - **Click Here to View Inbound Listener Details**

     This page lists the framework details including the names of any repositories to which the host machine is listening and the names and versions of installed inbound runtime modules. The names and status details of all running inbound listeners on the inbound host are shown after they are installed and are running. You configure and install inbound listeners using the integration activities in the process model from Advanced Workflow. The inbound_details.jsp page also lists the HTTP URL suffix for HTTP listeners.

   - **Click Here to View Inbound Web Services**

     This page lists the Web Service Definition Languages (WSDLs) that are available to Process Integrator. "Viewing a list of available web services" on page 28 provides more information about this page.

2.6.10 Viewing a list of available web services

Whenever an inbound web services listener is running, Process Integrator hosts the corresponding web services. You can view the list of all inbound web services hosted by Process Integrator using the web services validation URL.

The validation URL provides access to the following information:

- Name of the process with a link to the WSDL
- Port type
- Operations that the web service supports

1. Type the following URL into the browser:

   http://<*host_name*>:<*port*>/bps/webservice

   The page can also be accessed by clicking **Click Here to View Inbound Web services** on the inbound_details.jsp page or on the direct jsp page:

   ```
   http://<host_name>:<port>/bps/webservices.jsp
   ```

   "Validating inbound services and listeners" on page 27 provides instructions on using the .jsp pages.

2. To view the WSDL, click the **(wsdl)** link.

2.6.11 Validating outbound messaging configuration

When the Java Method Server is installed along with the Documentum CM Server, use the following URL to view details of the outbound messaging configuration. The validation URL provides access to the following information:

- System properties
- Installed product versions
- Repository information and runtime modules

1. Type the following URL into the browser after ensuring that the host_name and port_number in the URL reference the Java Method Server and not the server hosting the `bps.war` file:

   ```
   http://<host_name>:<port_number>/bpm/outbound_details.jsp
   ```

   If the system shows an error page rather than the **Outbound Details** page, ensure that you have correctly entered the host name and port number in the URL. If the error persists, there may be a problem with the JMS installation.

2. On the **Outbound Details** page, click the link for the appropriate page:

   - **View Versions**

     This page lists the details for the Java system properties on the host machine.

   - **View Docbases and Runtime Modules**

This page lists the framework details including the names of any repositories the host machine is listening to, the names and versions of installed outbound runtime modules, and the BPM Servlet URI.

2.6.12 Importing SSL certificates to support encrypted connections for process activities

To support SSL encryption, you must have an SSL certificate in your environment. Most of the certification authority (CA) issued certificates are validated by the system since the Java truststore contains most of the popular root certificates.

For self-signed certificates, the imported certificate must reside in the following locations:

- Java truststore used by Advanced Workflow
- Java Method Server bundled Java truststore to make it available during runtime outbound execution
- Application server bundled Java truststore where Process Integrator is installed to make it available to Process Integrator during runtime
- xDA application host bundled Java truststore

1. Generate or export the SSL certificate and note its location.
2. Use the following keytool command to import the certificate from the command prompt:

   ```
   keytool -import -noprompt -trustcacerts -alias <alias> -file <certificate-location>
    -keystore <java_truststore> -storepass <password>
   ```

   Where:

   - *<alias>* is the unique name for the alias
   - *<certificate-location>* is the location of the `brokercert` and `servercert` certificate
   - *<password>* is the password of the truststore (default is changeit)
   - *<java_truststore>* is the Java home location of Advanced Workflow, the bundled Java truststore of the method server, or the Java home location of the application server on which Process Integrator runs:
     - For Advanced Workflow:

       `<java location used by awfProcess>\lib\security\cacerts\`

     - For the Method Server:

       `%Documentum%\java64\JAVA_LINK\lib\security\cacerts\` (Microsoft Windows)

       `$DOCUMENTUM_SHARED/java64/JAVA_LINK/lib/security/cacerts/` (Linux)

– For the application server on which Process Integrator is running:

  Import the certificate into the Java truststore (cacerts) to which the application server points. For example:

  `%JAVA_HOME%\lib\security\cacerts\` (Microsoft Windows)

  `$JAVA_HOME/lib/security/cacerts\` (Linux)

– For the xDA host:

  For example:

  `%JAVA_HOME%\lib\security\cacerts\` (Microsoft Windows)

  `$JAVA_HOME/lib/security/cacerts/` (Linux)

3. Verify that the certificate imported successfully.

   The system shows a message that the certificate was added to the keystore after importing the certificate to the trust store. To list the content imported use the command:

```
keytool -v -list -alias <alias> -keystore <java_truststore> -storepass <password>
```

2.6.13 Configuring an HTTP Proxy server for web services

To access a web service that is located outside the firewall through an HTTP proxy server, configure the HTTP proxy parameters. Adding HTTP proxy parameters to the Documentum CM Server Java Method Server startup parameters enables web service access at application runtime.

1. To configure the Java Method Server startup parameters, open the `StartMethodServer.cmd` file from $DOCUMENTUM.

2. Add the HTTP proxy parameters for basic authentication or HTTP over SSL to the JAVA_OPTIONS.

   For example, add the following lines to support HTTP basic authentication:

```
-Dhttp.proxyHost=<proxy_host> -Dhttp.proxyPort=<port_number> -
Dhttps.nonProxyHosts="<non_proxy_hosts>"
```

   Where:

   - <*proxy_host*> is the host name of the proxy server.
   - <*port_number*> is the port number of the proxy server. The default value is 80 for HTTP Basic authentication and 443 for HTTP over SSL.
   - <*non_proxy_hosts*> is the list of hosts that can be reached directly, bypassing the proxy server. This list contains regular expressions separated by '|'. Any host matching one of these regular expressions is reached through a direct connection instead of through a proxy.

2.7 Installing the xDA

The prerequisites for installing the xDA are:

- Install the 64-bit Java Development Kit (JDK) and set JAVA_HOME (an operating system environment variable) to the JDK installation directory. Set environment variable <*PATH*> to `%JAVA_HOME%\bin\`.
- A client system or a VM for xDA Tools.

2.7.1 Setting up xDA and xDA tools

To install xDA:

1. Download and extract the contents of the `xDA_win64.zip` or `xDA_linux64.tar` to your local machine. Throughout this guide, `${xda-home}` refers to the root directory of xDA.
2. Edit the `xda-config.properties` in `${xda-home}\config`. The following table describes the value for each parameter:

Parameter	Description
xda.data.dir	(Optional) Specify the path to alternate location on the xDA machine where you want xDA to maintain catalog information and log files. Ensure that the directory name does not contain spaces. The default location is `${xda-home}\`.
logging.config	The path of the `log4j2.properties` file. Do not change the value.
server.port	Specify the port where you want the xDA application to run. Default port is 7000.
wildfly.libs.home	(Optional) Specify the path to the WildFly client directory on the xDA machine while deploying an application in the WildFly application server.
jboss.libs.home	(Optional) Specify the path to the JBoss client directory on the xDA machine while deploying an application in the JBoss application server.
websphereliberty.libs.home	(Optional) Specify the path to the WebSphere client directory on the xDA machine while deploying an application in the WebSphere application server.

Parameter	Description
redirectToHTTPS [1]	(Optional) Set this variable to true if the LoadBalancer is configured in HTTPS and the proxy is configure in HTTP mode. When set to true, HTTP response is redirected to HTTPS protocol.
redirectPort [1]	(Optional) If redirectToHTTPS is enabled Default redirection port is set to 443. Should be changed if HTTPS is configured on different port.
[1] If xDA is not accessible when we have proxy layer between LoadBalancer and xDA service in Cloud. Use the following properties to retrieve the protocol from LoadBalancer and route the response using the same protocol.	

3. For `log4j2.properties`, set logs for various parameters to debug the configuration.
4. Add these entries to the `xda-config.properties` file to enable SSL mode on xDA as follows:

```
server.port=8443
server.ssl.enabled=true
server.ssl.key-store=keystore.jks
server.ssl.key-store-password=<keystore-password>
server.ssl.key-alias=<certificate-alias>
server.ssl.key-password=<key-password>
server.ssl.trust-store=truststore.jks
server.ssl.trust-store-password=<truststore-password>
```

   This looks for the TrustStore and KeyStore at the `${xda-home}\xDA\bin\` folder.
5. While using strong encryption, use the appropriate java.policy and jce JAR along with the following properties:

```
server.ssl.ciphers=<ssl cipher used to create certificate>
server.ssl.protocol=<Compatible ssl protocol>

For example: If a certificate is created with "256-bit AES encryption with
SHA-1 message authentication and RSA key exchange",
set the properties as follows:

server.ssl.ciphers=TLS_RSA_WITH_AES_256_CBC_SHA
server.ssl.protocol=TLS
```

6. Configuring Repository service.

   By default, Foundation Java API searches the value of <*dfc.security.ssl.truststore*> (property for certificate) in `dfc.properties` file for SSL communication. If found, Foundation Java API uses the certificate-based SSL communication. Otherwise, Foundation Java API defaults to the `cipher-based` anonymous SSL supported by Java.

   JDK 17.x and later versions do not support anonymous ciphers for SSL or TLS communication and results in the handshake failure exception while xDA connects with repository. You can use one of these methods to connect with the repository:

   a. Anonymous SSL communication:

OpenText strongly recommends to use certificate-based SSL communication. However, if you want to use anonymous SSL communication, perform the following steps:

i. In JDK 17.x and later, open the `<Java_Home>\conf\security\java.security` file.

ii. Find the line starting with `jdk.tls.disabledAlgorithms`.

iii. Remove *anon* from the list of disabled algorithms.

iv. Save your changes.

b. Certificate-based SSL communication:

- Create a Truststore and add Documentum CM Server certificates (server, connection broker, and exdocbroker) to it.
- Add these arguments in `xda.bat` and `xDA-service-installer.xml` (Windows) or `xda.sh` (Linux) file of xDA as follows:

```
-Djavax.net.ssl.trustStore=<trust-store>
-Djavax.net.ssl.trustStorePassword=<trust-store-password>

Examples:

xda.bat : "%JAVAHOME%"\bin\java.exe -Djavax.net.ssl.trustStore=<trust-
store>
-Djavax.net.ssl.trustStorePassword=<trust-store-password>

xda-service-installer.xml : <arguments> -Xmx1024m
 -Djavax.net.ssl.trustStore=<trust-store>
-Djavax.net.ssl.trustStorePassword=<trust-store-password>

xda.sh : "$JAVA_HOME"/bin/java -Djavax.net.ssl.trustStore=<trust-store>
-Djavax.net.ssl.trustStorePassword=<trust-store-password>
```

- Add the Documentum CM Server certificates (server, connection broker, and exdocbroker) in Truststore specified in the *<server.ssl.trust-store>* property of the `xda-config.properties` file.

7. To start xDA application:

- For Windows: Run `%xda-home%\bin\xda.bat` using administrator role.

  Note: A separate command prompt is opened for BaseX server when you run the `xda.bat` command. Do not close either of the two command prompts.

- For Linux:

  a. Open located at `${xda-home}\xDA\bin\xda.sh` file.

  b. Append `-Djava.security.egd=file:/dev/./urandom` at the end of XDA_OPTS as follows:

```
XDA_OPTS="-Xmx4096m……
……/log4j2.properties -Djava.security.egd=file:/dev/./urandom"
```

c. Run `xda.sh` in ${xda-home}\bin to start the xDA application.

8. To verify that the xDA installation is successful, open the following URL in a web browser:

```
https://<host>:<port>/xda

<host> is the host name or IP address of the machine where xDA is installed.
<port> is the port number on which xDA is listening.
```

The installation is successful if you can see the login page.

Note: When you log in to xDA through GUI or xDA-Tools utility for the first time, the system prompts you to change the default password.

9. To install xDA tools on the same machine where you have installed xDA, complete the following steps:

a. Install the 64-bit JDK on a machine that meets the requirements for xDA Tools. Set JAVA_HOME (an operating system environment variable) to the directory where the 64-bit JDK is installed.

b. Download and extract `xda-tools_win64.zip` or `xda-tools_linux64.tar` to this machine.

c. The extracted xDA Tools folder structure is described in the following table:

Folder	Contents
bin	Executable files, including `xda.bat/sh` and `configure-xda.bat/sh`.
blueprints	Module-template files that describe the structure of environment templates used to register manually-provisioned environments.
config	Configuration files, including `xda.properties` to connect to xDA.
lib	Libraries needed to deploy the Advanced Workflow application and libraries required by the xDA API plug-ins.
logs	Log files.
reports	Reports generated by the xDA CLI commands.

Throughout this guide, `${xda-tools-home}` refers to the root directory of xDA Tools.

You can set up xDA Tools on any number of machines. Your organization might want to have xDA Tools installed on additional machines so that each member of a team can run xDA commands on a separate machine.

10. To enable SSL communication between xDA Tools and xDA, append these entries to the `XDA_OPTS` parameter in the `xda.bat` or `xda.sh` file located at `${xda-tools-home}\bin`:

```
-Djavax.net.ssl.trustStore=<path-to-xdatoolsTrustStore>
-Djavax.net.ssl.trustStorePassword=<truststore-password>
```

11. On an xDA tools machine, run the xDA configuration script:
    a. Connect to xDA: Set up xDA Tools and open the `${xda-tools-home}\config\xda.properties` file.
       - Specify the IP address of the machine where the xDA is running and the port number on which the xDA is listening.
       - For xda-schema, specify the xDA host protocol as HTTP or HTTPS.

       These entries point xDA Tools to the running xDA.
    b. Set up a password for the xDA admin user and run xDA configuration script: Double-click `configure-xda.bat` (for linux, run `configure-xda.sh`) in `${xda-tools-home}\bin\` folder. At the prompt, type the username and password. If you have not changed the default xDA password, you will be prompted to change it.

       Note: If you change login password for the first time using xDA tool or UI, you must reopen the xDA tool and run **configure-xda** command to synchronize the updated password.

    The system imports the following configuration information:

    - The `bootstrap-dctm-configuration.xml` file from `${xda-tools-home}\config\` folder.
    - Default environment templates from `${xda-tools-home}\blueprints\module-templates`.
    - For Advanced Workflow, only the developer environment template, `xCP-25.4-Developer-Environment-Template`, is supported.

12. To enable SSL communication between Advanced Workflow and xDA, add these entries to the `DocumentumAdvancedWorkflow.ini` file:

```
-Djavax.net.ssl.trustStore=<path-to-xcpTrustStore>
-Djavax.net.ssl.trustStorePassword=<truststore-password>
```

You can now use the xDA Management Center to register an environment. "Using the xDA management center to register an environment" on page 39 provides instructions.

2.7.2 Installing xDA as a Windows service

The prerequisites for installing xDA as a Windows service are as follows:

- Install the 64-bit Java Development Kit (JDK).
- Install the Microsoft .NET framework 3.5 version.

Installing xDA Windows service

1. Download and extract the contents of the `xDA_win64.zip` to your Windows machine. Throughout this guide, `${xda-home}` refers to the root directory of xDA.
2. Log in to the Windows machine as an administrator user or a user with administrative privileges.
3. At the `${xda-home}\bin` prompt, run the `manage-xda-services.bat` command and enter `I` when prompted to install and create xDA as a service:

```
xDA\bin>manage-xda-services.bat
Choose from the following options for xDA services
I to Install
U to Uninstall
S to Start
T to Stop
C to Check status
E to Exit

Enter your choice : I
```

4. Click **Start** > **Programs** > **Administrative Tools** > **Services**.
5. Open Windows Services application to verify if the xDA service has been successfully installed. The name of the service appears as **Documentum xCP Deployment Agent** and **Documentum xCP Deployment Agent Basex Service**.

Starting or stopping xDA Windows service

1. Log in to the Windows machine.
2. There are two ways to start or stop xDA windows service:
   - Using `create-xda.bat` file.
   - Using the Windows Services option
3. Using `manage-xda-services.bat`:
   - To start xDA service, run the `manage-xda-services.bat` command and enter `S` when prompted:

```
xDA\bin>manage-xda-services.bat
Choose from the following options for xDA services
I to Install
U to Uninstall
S to Start
T to Stop
C to Check status
E to Exit
```

```
Enter your choice : S
```

- To stop xDA service, run the `manage-xda-services.bat` command and enter `T` when prompted:

```
xDA\bin>manage-xda-services.bat
Choose from the following options for xDA services
I to Install
U to Uninstall
S to Start
T to Stop
C to Check status
E to Exit

Enter your choice : T
```

4. Using the Windows Services option:
   - To start xDA service:
     a. Click **Start > Programs > Administrative Tools > Services**.
     b. Select **Documentum xCP Deployment Basex Agent** service and click **Start service**.
     c. Select **Documentum xCP Deployment Agent** service and click **Start service**.
   - To stop xDA service:
     a. Click **Start> Programs** > **Administrative Tools** > **Services**.
     b. Select **Documentum xCP Deployment Agent** service and click **Stop service**.
     c. Select **Documentum xCP Deployment Basex Agent** service and click **Stop service**.

Uninstalling xDA Windows services

1. Log in to the Windows machine as an administrator user.
2. At the `${xda-home}\bin` prompt, run the `manage-xda-services.bat` command and enter `U` when prompted to uninstall xDA services:

```
xDA\bin>manage-xda-services.bat
Choose from the following options for xDA services
I to Install
U to Uninstall
S to Start
T to Stop
C to Check status
E to Exit

Enter your choice : U
```

Upgrading xDA

While installing any newer version of xDA or installing the xDA patch, if your `${xda-home}` path changes, the user must stop and remove the existing xDA Service.

1. Remove the existing xDA Service by following instructions in the **Uninstalling xDA Windows Service** section.
2. Reinstall the newer xDA service by following instructions in the **Installing xDA Windows Service** section.

Troubleshooting

While starting a xDA service, in case you experience any errors, check the `xDA-installer-service.err.log` file located in `${xda-home}\logs\` folder.

2.7.3 Connecting to xDA from xDA tools

After you have set up xDA Tools and installed xDA, you can connect to xDA from xDA Tools.

1. On each machine that you intend to use for connection to the xDA, set up xDA Tools. "Setting up xDA and xDA tools" on page 31 provides details.
2. On each machine where you have set up xDA Tools, edit the `xda.properties` file in `${xda-tools-home}\config`.

Example 2-1:

The following is an example of the `xda.properties` file:

```
xda-host = 192.0.2.0
xda-port = 8080
xda-schema = http
```

The following table describes each property in this file:

Property	Description
xda-host	The IP address or host name of xDA.
xda-port	The port of the xDA.
xda-schema	Specify the protocol for access to the xDA: • `HTTP` • `HTTPS`

You can point xDA Tools to a different xDA by replacing or revising the `xda.properties` file.

2.7.4 Logging In to xDA from xDA tools

1. To login to xDA, double-click **xda.bat** in `${xda-tools-home}\bin`.
2. At the prompt, type your username and password.

   Three failed attempts end the session.

To end a session, type `exit`.

2.8 Using the xDA management center to register an environment

The xDA Management Center lets you register manually-provisioned environments. This process includes configuring various manually-installed Advanced Workflow components.

The statuses for a manually-provisioned environment, a service, or an endpoint are similar. The following table describes these possible statuses:

Status	Manual environment	Service	Endpoint
Registered	The system has all required information for all host groups and services.	The service has all required information.	The endpoint has all required information.
Registration Incomplete	Some required information for host groups or services is missing.	Some required information for the service is missing.	Some required information for the endpoint is missing.

Note: xDA can only have repository and application server endpoints.

2.8.1 Logging In to the xDA management center

Before you log in to the xDA Management Center:

- Install the xDA. "Installing the xDA" on page 31 provides details.

1. Open the following URL in a browser:

   `http://<host>:<port>/xda`

   where <*host*> is the host name or IP address of the machine where the xDA is running and <*port*> is the port number on which the xDA is listening. If this server is the same computer as the browser, you can use:

   `http://localhost:<port>/xda`

2. Type xDA username and password. Click **Login**.

2.8.2 Configuring an environment template

The system uses an environment template as a base when you register your environment. Default environment templates are available for this purpose. However, if you need to customize an environment template to use the supported Application servers, or for any reason, use the following steps:

1. Start xDA and log in to xDA Management Center. "Logging In to the xDA management center" on page 39 provides instructions.
2. Click **Catalog**. A list of environment templates appears.
3. Clone an environment template:
   a. Select an environment template and click **Clone Template**. The system creates a copy of the selected environment template.
   b. Type a unique name for the new environment template.
   c. Type a unique key for the environment template.
   d. (Optional) Type a description for the new environment template.
   e. Click **Done**. The system saves the information to the database.
   f. Click **OK**. The new environment template appears in the list.
4. Update the new environment template:
   a. Select the environment template and click **Update Template**.
   b. Optional In the **Properties** tab, type a key and a description for the environment template. These fields contain the same values as when you cloned the environment template.
   c. On the **Prerequisites** tab, specify on which application server you intend to deploy the application.
   d. In the **Host Groups** tab, select each host group and click **Edit**. The **Edit Host Group** dialog box appears. For each host group, specify value for the field described in the following table:

Field	Description
Description	(Optional) Type a description for the host group.

   In **Edit Host Group** dialog box, click **Done**. The system saves any new information into memory, but does not save the information to the database.
   e. In the **Services** tab, select each service and click **Edit**. The **Edit Service** dialog box appears. Configure each tab as follows:
      - On the **Properties** tab, specify values for the fields described in the following table:

Field	Description
Enable Service	Indicate whether you want to enable the service for the environment template. Some services, such as the repository service, are required. You cannot disable these services.
Host Group	Specify which host group you want to associate with the service.

- On the **Service Components** tab, indicate which optional service components you want to disable for the service.

Click **Done**. The system saves any new information into memory, but does not save the information to the database.

f. Click **Finish**. The system saves all information (any information in memory and any new information) to the database.

2.8.3 Registering an environment

1. Start xDA and log in to the xDA Management Center. "Logging In to the xDA management center" on page 39 provides instructions.
2. Click **Environments**.
3. Click **Create Environment**. The **Environment Template Selector** appears.
4. (Optional) Click **View Details** next to each environment template name for more details.
5. Select an environment template. After you finish creating the environment, you cannot choose a different environment template for this environment.
6. Click **Done**. The system saves any new information into memory, but does not save the information to the database.
7. Configure each tab:
   - "Configuring general properties" on page 42
   - "Configuring accounts" on page 42
   - "Configuring host groups" on page 42
   - "Configuring hosts" on page 43
   - "Configuring services" on page 43
8. At any point, you can click **Apply** to save to the database any properties you have specified so far.
9. When you have finished configuring the environment, click **Finish**. The system saves all information (any information in memory and any new information) to the database.

2.8.4 Configuring general properties

In the **Properties** tab, specify environment properties as described in the following table:

Field	Description
Environment Mode	Select either Production or Development as the mode of the environment. After creating an environment, you can change the environment mode only from Production to Development. When you change the environment mode to Development, you need to create an environment again with environment mode as Production.
Environment Name	Type a name for the environment, unique within the xDA Management Center.
Description	(Optional) Type a description for the environment.

2.8.5 Configuring accounts

In the **Accounts** tab, provide credentials to serve as the default account for service endpoints:

Field	Description
Username	Type a username to access the relevant systems.
Password	Type the password for accessing the relevant systems.
Domain	(Optional) Type the domain for accessing the relevant systems. For clustered environments, the domain is required.

2.8.6 Configuring host groups

The system automatically populates the **Host Groups** tab with information from the selected environment template.

In the **Host Groups** tab, configure each host group as a container for one or more hosts:

1. Select the host group that you want to modify and click **Edit**. Type a description for the host group and click **Done**. The system saves any new information into memory, but does not save the information to the database.
2. If you want to add a host group, click **Add**. Type a name for the host group, unique within the environment. After you finish creating the host group, you cannot type a different name for this host group. Type a description for the host

group. If you want to add another host group, click **Save and Add Another**. Otherwise, click **Done**. The system saves any new information into memory, but does not save the information to the database.

Delete is not available for a default host group, or for a host group associated with a service.

2.8.7 Configuring hosts

Use the **Hosts** tab to specify each host in your environment:

1. Click **Add**.
2. Specify an IP address or a host name.
3. Select a host group.
4. If you want to add another host, click **Save and Add Another**. Otherwise, click **Done**. The system saves any new information into memory, but does not save the information to the database.
5. If you want to modify the properties for a host, select the host from the list and click **Edit**. When you have finished modifying the host, click **Done**. The system saves any new information into memory, but does not save the information to the database.

2.8.8 Configuring services

In the **Services** tab, configure each service in your environment:

1. Select the service from the list and click **Edit Service**.
2. Specify the general properties of the service as described in the following table:

Field	Description
Enable Service	(Optional) If you do not need this service in your environment, clear the checkbox to disable the service. The repository service is always enabled.
Host Group	Select the host group on which the service is running. If you change the host group associated with a service, remember to update endpoint properties to choose one or more hosts from that host group.

Field	Description

Version	Type the version of the service, using the following format:
	`mm.mn.xxxx.yyyy`
	Where:
	• `mm` is the major version number
	• `mn` is the minor version number
	• `xxxx` is the update number
	• `yyyy` is the build number
	For example, the service version can be:
	`7.0.0001.0005`

3. On the **Endpoints** tab for each service, configure each service endpoint. Endpoints enable you to configure communication pathways between systems. Select the endpoint and click **Edit**.

   a. "Service endpoint configuration" on page 44 describes how to configure the fields unique to each endpoint.

   b. Specify credentials for the service endpoints that require credentials: By default, some services use the credentials that you specified on the **Accounts** tab. If you want the service to use different credentials, clear **Use Default Service Account?**. Specify a username, password, and (optionally) domain.

      For the Tomcat or tc Server application server, specify Tomcat manager user credentials.

   c. Select one or more hosts for the service endpoint. The list of hosts depends on the host group associated with the service endpoint.

   d. When you have finished configuring the service endpoint, click **Done**. The system saves any new information into memory, but does not save the information to the database.

4. When you have finished configuring the service, click **Apply** or **Finish**. The system saves all information (any information in memory and any new information) to the database.

2.8.9 Service endpoint configuration

"Configuring services" on page 43 describes how to specify credentials and hosts for service endpoints. The following table lists the endpoints for each service and describes how to configure each endpoint further:

Note: xDA can have repository and application server endpoints only. By default, AppHostService endpoint fields are not required for repository deployment, but these fields are marked as mandatory, hence you must provide any sample values for the Port, Protocol, and URL fields.

Service	Endpoint	Field	Description
Documentum Administrator	daEndpoint	Port	Type the HTTP port of the Documentum Administrator.
		URL	Type the URL to access the Documentum Administrator. Include the protocol, host, port, and default URL prefix as follows: `http://<host>:<port>/da`
Process Integrator	bpsEndpoint	Port	Type the HTTP port of the Process Integrator component.
		URL	Type the URL to access the Process Integrator component. Include the protocol, host, port, and default URL prefix as follows: `http://<host>:<port>/bps`
Repository	repositoryEndpoint	Repository Name	OTDS user with valid authorization and Superuser privileges.
Connection Broker Port	Type the HTTP port of the machine where connection broker is installed.		
AppHostService	appHostEndpoint	Port	Enter any sample HTTP port of AppHost Service.
		Protocol	Select any sample protocol for AppHost Service.
		URL	Enter any sample URL for AppHost Service.

Notes

- The following service endpoints are optional:
  - BpsService
  - CustomApplicationService
  - DaService
- The following service endpoints are optional and should be disabled:
  - AdtsService ()
  - AvtsService
  - BamService
  - CisService
  - MtsService
  - SearchService

2.8.10 Updating environment configuration

1. Start the xDA Management Center and log in. "Logging In to the xDA management center" on page 39 provides instructions.
2. Select the environment that you want to update. Click **Update Environment**.
3. Configure each tab:
   - "Configuring general properties" on page 42
   - "Configuring accounts" on page 42
   - "Configuring host groups" on page 42
   - "Configuring hosts" on page 43
   - "Configuring services" on page 43
4. At any point, you can click **Apply** to save to the database any properties you have specified so far.

2.8.11 Viewing service component versions

1. Start xDA and log in to the xDA Management Center. "Logging In to the xDA management center" on page 39 provides instructions.
2. Select the environment for which you want to view the versions of the service components.
3. Depending on the option you select to view service component versions, perform these steps:
   a. **View Environment**
   b. **Validate Environment Result**
4. **View Environment**: Perform the following steps:
   a. Click **View Environment**.
   b. Examine the information in the **Component Properties** tab.
   c. View the properties of the service components as described in the following table:

Field	Description
Service Component	Identifies service components that are part of a specific service.
Service	Identifies the services in the environment.
Version	Identifies the version of a specific service component. By default, all default versions appear for the service components. When you synchronize the environment, the system fetches, saves, and displays the actual version for service components. In case services are not available or in down state, the default versions are displayed.

   d. Click **Copy version information** to copy version specific information of the services to the clipboard. On successful operation, you can paste the information to any external text editor.

      Note: On successful copy operation, the system shows the success message.

   e. Click **Export version information** to export the service component information in **CSV** format. The information contains these entities—service component, service, and version.
   f. Click **OK**.

5. Validate Environment Results: Perform the following steps:

   a. Click **Validate Environment Results**.

   b. Examine the information in the **Service Status** screen.

   c. View the properties of the service components as described in the following table:

Field	Description
Service Component	Identifies core service component that is part of a specific service.
Validation URL Details	Identifies the URL of a service component.
Status	Identifies the status of the service component—Failed or Passed.
Version	Identifies the version of the core service component. By default, all default versions appear for the service components. However, when you synchronize the environment, the system fetches and displays the actual version for the core service component. For example, for CTS - Documentsservice, the CTS - Documents, and JDK services are required. On synchronization, the system displays the actual version of the CTS - Documents service.

   d. Click **OK**.

2.8.12 Synchronizing a manually provisioned environment

When you synchronize an environment, the xDA examines your environment and updates the system to reflect the correct services, components, and their versions. Perform synchronization in each of the following scenarios:

- After you register your manually provisioned environment.
- After manually changing any components in your environment.
- Before deploying an Advanced Workflow application to your environment.

Synchronizing an environment requires a connection to the xDA.

2.8.13 Synchronizing an environment

1. Start xDA and log in to the xDA Management Center. A list of environments appears.
2. Select the environment that you want to synchronize. Click **Maintenance**. The **Maintenance** dialog box for the selected environment appears.
3. Click **Synchronize Environment** and then click **OK**.

   Details about the running job are available on the **Operations** tab.
4. (Optional) When the synchronization completes, verify the changes: In the list of environments, select the environment and click **View Environment**. Examine the information on the **Services** tab.

After the environment has been successfully synchronized, you can deploy Advanced Workflow applications to the environment.

2.8.14 Unregistering an environment

You can use the xDA Management Center to unregister an environment, if it is not currently in use. If the system is currently registering the environment, wait until that process completes before attempting to unregister the environment.

1. Start xDA and log in to the xDA Management Center."Logging In to the xDA management center" on page 39 provides instructions.
2. Select the environment or environments that you want to unregister. Click **Unregister Environment** and then click **Yes**.

2.9 Using commands to create an environment

The `environment.yml` file lets you create manually-provisioned environment and it includes setting up values for various manually-installed Advanced Workflow components.

As a prerequisite, make sure that xDA is installed and configured with Advanced Workflow environment templates. Installing the xDA provides details about installing xDA.

2.9.1 Creating an environment

1. Double-click **xda.bat** in `${xda-tools-home}\bin` and log in to xDA.
2. At the xda> prompt, type:

   ```
   create-environment --yml-file <Location of YML file>>
   ```

   The sample `environment.yml` is located at the `${xda-tools-home}\config\` folder. Edit this file to provide your environment endpoint details.
3. Resolve any errors that may occur and run the command again.

   On successful execution of the command, the system displays this successful message. For example:

   ```
   Environment "xyz" created successfully with "Registered" status.
   ```

   All values for password related properties mentioned in the `environment.yml` file can be entered as plain text, encrypted or can be left blank. The system prompts you to enter password on the console when password is left blank in the `environment.yml` file.
4. Configure each section in the `environment.yml` file:

   - Encrypting Passwords
   - “Configuring general properties” on page 51
   - “Configuring accounts” on page 52
   - “Configuring host groups and hosts” on page 52
   - “Configuring services” on page 53

2.9.2 Synchronizing an environment using xDA tools

To set the environment status to `Provisioned`, use the `discover-environment` command.

1. Double-click **xda.bat** in `${xda-tools-home}\bin` and log in to xDA.
2. At the xda> prompt, type

   ```
   discover-environment --name <Env_name>
   ```
3. Log in to xDA UI and verify that the status of the environment is set to **Provisioned**.

- The command encrypts the passwords mentioned in plain text in the `passwords.properties` file and creates an encrypted counterpart text file at `${xda-tools-home}\config`. For example, when you specify location of `passwords.properties` file, the system encrypt the passwords and creates an encrypted counterpart text file as the `encrypted_passwords.txt` file.
- If you do not specify the location of password properties file, the `encrypt-passwords` command, by default, picks up the `${xda-tools-home}\config\properties-file\passwords.properties` file to encrypt passwords.

2.9.3 Encrypting passwords

For security purpose, use the command to encrypt your system account passwords. You may specify these encrypted passwords in the `environment.yml` file while creating environment using the `create-environment` command.

1. Double-click **xda.bat** in `${xda-tools-home}\bin` and log in to xDA.
2. At the xda> prompt, type

```
encrypt-passwords [--properties-file <Location of Password Properties File>]
```

- The command encrypts the passwords mentioned in plain text in the `passwords.properties` file and creates an encrypted counterpart text file at `${xda-tools-home}\config`. For example, when you specify location of `passwords.properties` file, the system encrypt the passwords and creates an encrypted counterpart text file as the `encrypted_passwords.txt` file.
- If you do not specify the location of password properties file, the `encrypt-passwords` command, by default, picks up the `${xda-tools-home}\config\properties-file\passwords.properties` file to encrypt passwords.

2.9.4 Configuring general properties

The system uses an environment template as a base when you register your environment. Default environment templates are available for this purpose.

1. Locate the `environment.yml` file in the `${xda-tools-home}\config\` folder and open it with any external text editor.

```
# environmentTemplate, environmentName are mandatory fields
environmentTemplate: xCP-25.4-Developer-Environment-Template
environmentName: myEnvironmentName12345
# environmentMode is mandatory and can be 'Development' or 'Production'
environmentMode: Development
environmentDescription: My Environment Description
```

2. For each property, specify value in the field described in the following table:

Fields	Description
environmentTemplate	Name of the environment template. Configuring an Environment Template provides instructions.
environmentName	Unique name for the environment.
environmentMode	Select either Production or Development as the mode of the environment.
environmentDescription	(Optional)Description for the environment.

2.9.5 Configuring accounts

1. Edit the `environment.yml` file in the `${xda-tools-home}\config\` folder with any external text editor.

```
defaultSystemAccount:
  username: myDefaultUsername
  password:
  domain:
```

2. In the `defaultSystemAccount`section, specify value in the field described in the following table:

Fields	Description
username	Type a username to access the relevant systems.
password	Type the password for accessing the relevant systems.
domain	(Optional) Type the domain for accessing the relevant systems. For clustered environments, the domain is required.

2.9.6 Configuring host groups and hosts

1. Edit the `environment.yml` file in the `${xda-tools-home}\config\` folder with any external text editor.

```
hostGroups:
    # 'name' is mandatory field for hostGroup
  - name: xCPVM1
    description: Developer Environment Host Group 1
    # Provide either 'hostname' or 'ipAddress' for a host
    hosts:
    - hostname: MYHOST1
      ipAddress:
     # uncomment below to add multiple hosts to host group
#    - hostname:
#      ipAddress: 192.168.1.1
```

2. In the `hostGroups` section, configure each host group as a container for one or more hosts:

Fields	Description
name	Type a name for the host group, unique within the environment.
description	(Optional) Type the description for the host group.

3. In the `hosts` section, specify one or more hosts for your environment:

Fields	Description
hostname	Type the host name associated with a host group.
ipaddress	Type an IP address associated with a host group.

Note: Specify either an IP address or host name.

You can add multiple host groups in a similar manner.

2.9.7 Configuring services

1. Edit the `environment.yml` file in the `${xda-tools-home}\config\` folder with any external text editor.
2. In the `services` section, configure each service for your environment.

Fields	Description
serviceName	Displays the name of the service. Do not change the name of the service.
enabled	If you do not need this service in your environment, set this value as `false` to disable the service.

version	Type the version of the service, using the following format:
	`mm.mn.xxxx.yyyy`
	`Where:`
	`· mm is the major version number`
	`· mn is the minor version number`
	`· xxxx is the update number`
	`· yyyy is the build number`
	`For example, the service version can be:`
	`25.4.0000.0005`

hostGroup	Type the existing host group on which the service is running. It is a mandatory field for service and it must be existing HostGroup name from the `hostGroups` section.

3. "Service endpoint configuration" on page 54 describes what values to provide for service endpoint properties.
4. If your service uses different credentials, set *<useDefaultSystemAccount>* as `false`. Otherwise, the service uses `defaultSystemAccount` credentials.
5. Select one or more `hosts` for the service endpoint. The list of hosts depends on the host group associated with the service endpoint.
6. Save the file.

2.9.8 Service endpoint configuration

Endpoints enable you to configure communication pathways between systems. "Configuring services" on page 53 describes how to specify credentials and hosts for service endpoints. The following table lists the endpoints for each service and describes how to configure each endpoint further:

Service	Endpoint	Property	Description
Documentum Administrator	daEndpoint	port	Type the HTTP port of the Documentum Administrator.
		protocol	Type the HTTP or HTTPS protocol.
		url	Type the URL to access the Documentum Administrator. Include the protocol, host, port, and default URL prefix as follows: `<protocol>://<host>:<port>/da`
Process Integrator (BPS)	bpsEndpoint	port	Type the HTTP port of the Process Integrator component.
		url	Type the URL to access the Process Integrator component. Include the protocol, host, port, and default URL prefix as follows: `<protocol>://<host>:<port>/bps`
Repository	repositoryEndpoint	repositoryName	OTDS user with valid authorization and Superuser privileges.
Connection Broker Port	Type the HTTP port of the machine where connection broker is installed.		

2.9.9 Deleting an environment

Use the command to delete an environment if it is not currently in use.

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the xda> prompt, type:

```
delete-environment --environment-name <Environment Name>
```

Specify an existing environment name for deletion.

2.10 Managing xDA users

You can use the xDA Management Center or a set of commands to manage xDA users.

The following points apply to user management:	• The same guidelines for username and password apply to all xDA users, including the default `admin` user. "Guidelines for username and password creation" on page 56 provides details. • You cannot delete the default `admin` user. • Only the `admin` user can create user, change user password, and delete user. • Users cannot delete themselves. • There is no password retrieval in xDA. • Usernames are case-sensitive. • You cannot change the username after you finish creating an account.
The following additional points apply when you are using the xDA Management Center to manage users:	• All users can access My Profile, where they can update only their own user account. • Only the default `admin` user can access the Users tab, to add, update, or delete user accounts. • Each user account requires a username, password, first name, and last name. You can also specify an email address.

2.10.1 Guidelines for username and password creation

The following table provides guidelines for username and password creation:

Guideline	Username	Password
Minimum length	4 characters	8 characters
Maximum length	12 characters	(No maximum)
Character usage	You can use alphanumeric characters, underscore (_), and one period (.). Do not use spaces or any other special characters.	Use at least one non-alphabetic character.

2.10.2 Using the xDA Management Center

The xDA Management Center lets `admin` user to add, update and delete users. All users can access **My Profile**, where you can update only your own user account.

Creating a user

1. Start xDA and log in to xDA Management Center with `admin` user. "Logging In to the xDA management center" on page 39 provides instructions.
2. Click **Users**.
3. Click **Add User**.

   Note: You cannot change the username after you finish creating an account.

4. In the **Add User** screen, type the following details:
   - Username
   - Password
   - Confirm password
   - First name
   - Last name
   - (Optional) Email Id
5. Click **Finish**. The system saves the new user information in the database.

Updating a user

The `admin` user can update these items— first name, last name, email ID and change the password.

1. Start xDA and log in to xDA Management Center with `admin` user. "Logging In to the xDA management center" on page 39 provides instructions.

2. Click **Users**.
3. Click **Update User**.
4. In the **Update User** screen, type the following details:
   - First name
   - Last name
   - (Optional) Email Id
5. If you intend to change the password, select the **Change Password** checkbox, type the new password to change the password.
6. Click **Finish**. The system saves the updated user information in the database.

Deleting a user

1. Start xDA and log in to xDA Management Center with `admin` user. "Logging In to the xDA management center" on page 39 provides instructions.
2. Click **Users**.
3. Click **Delete User**. You cannot delete the default `admin` user.
4. Click **Yes** to confirm the deletion of the user. The system deletes the user information from the database.

2.10.3 Using commands

The following table lists the commands to manage xDA users:

Command	Description
`change-password`	Changes xDA user password.
`create-user`	Creates a new xDA user.
`delete-user`	Deletes an existing xDA user.
`show-users`	Shows xDA users list.

2.10.4 Creating a user

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the **xda>** prompt, run the following command:

```
create-user -username <username> [-password <password>]
```

Notes

- Only the `admin` user can create other users using this command.
- If the user provides only username, the system prompts you to enter the password in a masked (*) format.

3. If the new user needs access to xDA from a different machine, give the new user a copy of the `xda-tools.zip` file and the `xda.properties` file.

2.10.5 Viewing a list of users

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the **xda>** prompt, run the following command:

   ```
   show-users
   ```

   The information for each user includes:

   - Username
   - The user creation date
   - Name of the user who created this user
   - Last modification date for the user
   - Name of the user who last modified this user

2.10.6 Changing a user password

The following principles apply for changing the user password:

- Only the `admin` user can change the password of other users.
- You can change your own password.
- There is no password retrieval in xDA.

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the **xda>** prompt, run the following command:

   ```
   show-users
   ```

3. At the **xda>** prompt, run the following command:

   ```
   change-password -username  <name>  [-newpassword <password>]
   ```

   where <*password*> is the new password for the named user.

   Note: If the user provides only username, the system prompts you to enter the password in a masked (*) format.

2.10.7 Deleting a user

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the **xda>** prompt, run the following command:
   ```
   show-users
   ```
3. At the **xda>** prompt, run the following command:
   ```
   delete-user -username <name>
   ```
   where <*name*> is the name of the user that you want to delete.

   Note: Only the `admin` user can delete any user.
4. When the system prompts you to confirm, enter:
   ```
   y
   ```

2.11 Viewing environments

After you have created an environment, you might need to view a list of environments or view the properties of the environment, as described in the following sections.

2.11.1 Viewing a list of environments

You can use the xDA Management Center or a command to view a list of environments.

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the **xda>** prompt, run the following command:
   ```
   show-environments [-report <reportname>]
   ```
   []

   If you do not specify parameters, the system shows a list of all environments in the system. The information for each environment includes:

   - Environment name
   - Environment template used to register the environment
   - Environment label (description)
   - Provisioning date of the environment
   - Status of the environment
   - Name of the user who created the environment

   If you specify a parameter, the system shows information or creates a report as described in the following table:

Parameter	Description
`--report`	The system creates a report containing the information shown with the `show-environments` command. The default location is `${xda-tools-home}\reports\`. You cannot save reports outside the `reports` folder. The report name depends on the following: • If you do not specify a name, the system creates a report with the name *<yyyy-mm-dd>*`-show-environments.txt` file and saves it in the default location. • If you specify a name, the system creates a report with that name and .txt extension, and puts it in the default location. The system converts any spaces in the name to an underscore (_). The system appends a .txt extension to the name even if you specify a different extension, for example, report.doc becomes report.doc.txt. The system converts a .TXT extension to a .txt extension.

2.11.2 Viewing properties of a specific environment

You can use the xDA Management Center or a command to view the properties of a specific environment.

1. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in to xDA.
2. At the **xda>** prompt, run the following command:

```
show-environment-details --name <name> [--report <reportname>] [--services] [--applications] [--endpoints]
```

where the only required parameter is `--name` *<name>*.

The system shows information or creates a report as described in the following table:

Parameter	Description
`--name`	The environment name. The system shows basic information, including the environment name, the environment description, the template name, the provisioning date of the environment, and the mode of the environment. Additionally, it shows a list of services, endpoints, and Advanced Workflow applications deployed on that environment.
`--report`	The system creates a report containing the information shown with the `show-environment-details` command. The default location is *<${xda-tools-home}\reports\>*. You cannot save reports outside the `reports` folder. The report name depends on the following: • If you do not specify a name, the system creates a report with the name *<yyyy-mm-dd>*`-show-environment-details.txt` and places it in the default location. • If you specify a name, the system creates a report with that name and .txt extension, and places it in the default location. The system converts any spaces in the name to an underscore (_). The system appends a .txt extension to the name even if you specify a different extension, for example, report.doc becomes `report.doc.txt`. The system converts a .TXT extension to a .txt extension.
`--services`	The system shows basic environment and services information. Services include, for example, VM information and IP addresses.
`--applications`	The system shows basic environment and application information. Application information includes a list of all the Advanced Workflow applications deployed on the specified environment. For each application in the list, the system shows the application name, the application version, and the deployment date of the application.

Parameter	Description
`--endpoints`	The system shows basic environment and endpoint information. Endpoint information includes endpoint name, service name and application endpoint details, for example, host, port, protocol and URL.

2.12 Installing Advanced Workflow

If you want to use multiple instances of Advanced Workflow on your computer, use the same version of Advanced Workflow for each instance. The first instance of Advanced Workflow has an internal application server that subsequent instances use. The first instance needs to stay open to use the other instances.

1. Download the Advanced Workflow ZIP file.
2. Extract the contents of the file to your computer.

   The file contents are stored in an `DocumentumAdvancedWorkflow` folder.
3. Double-click **DocumentumAdvancedWorkflow.exe** to open Advanced Workflow.

Chapter 3

Installing an Advanced Workflow application

3.1 Overview of installing an Advanced Workflow application

The following table lists the application installation methods, connections, and data policies that you can use with each environment mode:

Environment mode	Application installation method	Application installation connection	Application installation data policy
Development	Advanced Workflow only	xDA only	Clean, Minimal, or Maintain
Production	Advanced Workflow or xDA Tools	xDA only	Maintain only

For the prerequisites for installing an application, see "Registering an environment" on page 41.

Best practice:

- To avoid disk space issues, clear the cache before deploying applications.

3.2 Application installation using Advanced Workflow

In Advanced Workflow, you typically install an application to a Development environment or a test environment. You can install an application in secure mode as well. The prerequisites for installing an application using Advanced Workflow are:

- Information registered in Advanced Workflow. The required information includes:
  - The name of the installation environment.
  - The IP address and port number of xDA so that the system can connect to the xDA.
  - xDA credentials (username and password).
  - The name of the target environment registered in xDA on which you intend to install the application.
- For installing large and complex applications using Advanced Workflow, you need to increase the heap size of XMX to 1024 in the `.ini` file of the Advanced Workflow. Then, restart the Advanced Workflow.

When you click **Run Application** on the Advanced Workflow toolbar, the system installs the application to the environment.

If you make a change to an installed business object that requires data to be destroyed, running the application again forces the changed business object definitions to be reinstalled. In the following scenarios, data associated with the business object is deleted:

- Removing an attribute from a type
- Renaming an attribute on a type
- Shortening the length of a string attribute
- Changing an attribute from single to repeating
- Changing an attribute from repeating to single

Business object instances are also deleted when the business object is deleted from the application.

If you have test data in your environment, any of the preceding changes to your data model causes some data to be lost. As a best practice for these situations, build loader scripts using API, DQL or Foundation Java API code to load the data, and periodically use the clean option to clean your data and then reload it.

As a best practice, if you make a change to an application that is already in production, use the maintain data policy to ensure that you do not run into issues during application upgrade, since maintain is the only supported data policy in a production environment.

3.3 Application installation using xDA tools

A developer or consultant packages an application created in Advanced Workflow when it is ready to be installed to a staging or production environment. Packaging is the automated process of validating an application, compiling it, and putting the application in the correct format for an environment. The packaging process produces two files: a WAR file and an application configuration file in XML format. You could receive these files in various ways, for example, after an upload to a staging area, by FTP, in email, or on physical media. After you receive these files, save them to a desired location.

When you deploy an application for the first time, you can enter or overwrite endpoint and parameter values in the application configuration file. When you redeploy the same application to an environment in the production mode, the system deploys only new parameter or endpoint values. You can deploy an application in secure mode as well.

The prerequisites for deploying an application using xDA Tools are listed in "Overview of installing an Advanced Workflow application" on page 63.

3.3.1 Deploying Advanced Workflow application with xDA tools

1. Go to the Advanced Workflow application package location.
2. Open the application configuration file in a text editor.
3. Enter or overwrite endpoint or parameter values as necessary. For example, you could enter a value for the property name WSDL URL in the ABC Web Service endpoint as shown in the following code sample:

```
<configuration type="Web Service (SOAP) endpoint" name="ABC Web Service">
   <property name="WSDL URL">http://www.someurl.com</property>
   <property name="User name"></property>
   <property name="Password"></property>
</configuration>
```

   Use the Manage Application Parameters page in Advanced Workflow to update application parameter values after you have deployed an application.
4. Save the file.
5. Double-click **xda.bat** in `${xda-tools-home}\bin\` and log in with xDA username and the password of that user.
6. At the **<xda>** prompt, run the following command:

```
deploy-xcp-application --environment <name> --war-file <filename> --configuration-
file <filename> [--deployment-method <method>] [--data-policy <policy>] [--
validateonly <validateonlyflag>] [--repositoryservicesonly
<repositoryservicesonlyflag>]
```

   Specify the parameters `--war-file`, `--configuration-file`, and —`environment`, either on the command line or in the `deploy-xcp-application.properties` file.

   The following table describes the parameters:

Parameter	Description
war-file	Specify the path, including the file name, to the WAR file in the application package. For model-only deployment, specify the path, including the file name, to the **artifact_bundle JAR** file and set the **repositoryservicesonly** parameter value as `true`. If the path to the file contains a space, put double quotation marks around the path.
configuration-file	Specify the path, including the file name, to the application configuration file in the application package. If the path to the configuration file contains a space, put double quotation marks around the path.

Parameter	Description
environment	Specify the environment name where you want to deploy the application.
deployment-method	(Optional) Specify the deployment method. The method can be: • Incremental: Deploys only application components that changed since the last deployment or new application components (for example, an updated business object model or a new application page). This is the default. • Full: Deploys the full application.

Parameter	Description
data-policy	(Optional) Specify the data policy. The data policy defines what happens to application data in the environment when you deploy a new version of the application. Examples of application data include runtime instances of business objects, content objects, processes, and reports. The policy can be: • Clean: Deletes application data from the environment. • Minimal: Updates application data associated with application data that changed since the last deployment. This is the default if you set the environment mode to Development. • Maintain: Preserves application data. Deploying the updated application does not affect existing application data in the environment. This is the default if you set the environment mode to Production. Best practice: Use **Maintain** when there is an existing production deployment of the application you are working on and you want to change the application. Setting the deployment to use **Maintain** mode ensures you capture any data upgrade issues before you deploy to the Production environment. Use a representative set of test data in your Development environment to accurately catch any issues that might occur in production when you try to redeploy the upgraded application. An environment set to Development mode supports all the data policies. An environment set to Production mode supports only the Maintain policy. If you have test data that is important, create test loader scripts using API, DQL, or Foundation Java API code to load the data. If your data policy is set to **Minimal** or **Clean**, you could lose the test data, in which means that you need an automated way to repopulate it.

Parameter	Description
validateonly	(Optional) Indicate whether the entire deployment must be performed or only the validation step. The options are: • `False`: Validates the application and deploys it if it is valid. This is the default. • `True`: Validates the application and logs information about full-text indexing considerations that helps you decide if you want to skip full-text indexing during the deployment.
repositoryservicesonly	(Optional) Indicates whether to deploy the complete application or only the repository artifact bundle. The options are: • `False`: The system accepts the WAR file for the *`--war-file`* parameter. This is the default. • `True`: The system accepts the artifact bundle JAR file for the *`--war-file`* parameter.

The system looks for manifest files in subfolders of `${xda-home}\installs\` on the machine where xDA is running. After successful application deployment, the system shows a message that it executed the command.

When an application is deployed, system creates temporary files in `${xda-home}\temp\work`. System automatically delete these temporary files post application deployment.

3.4 Deploying Advanced Workflow application with Headless Composer

1. In Composer, add the new plugin `com.emc.ide.installer.xcpdarinstaller`under the `plugins` folder.
2. Backup and then replace the shared plugin `com.emc.ide.installer.darinstaller_1.0.0` under the plugins folder.
3. Clear the cache folders:
   a. Delete all folders from `darinstallerconfiguration` folder except the `config.ini` .
   b. Delete the `org.eclipse.core.runtime` and `org.eclipse.osgi`folders from `configuration` folder.
   c. Delete all the contents from `darinstallerworkspaces` folder.

4. Deploy the artifact bundle JAR. Copy the `artifact bundle.jar` from `target\application.war` and `target\application.config` file from Advanced Workflow into Headless Composer.

3.5 Licensing OpenText Documentum CM

OpenText Documentum CM uses OpenText Directory Services (OTDS) to apply licenses for all the OpenText Documentum CM components. For more information about procuring the license file and configuring OTDS and license, see *OpenText Documentum Content Management - Server and Server Extensions Installation Guide (EDCSY250400-IGD)*.

3.6 Integrating the GHS into Advanced Workflow

Global Help Server (GHS) with Advanced Workflow Process is now available by default. All the help information is retrieved from the help server. You need to ensure that Internet connectivity is available to access help information from GHS.

3.7 Integrating the PHS into Advanced Workflow

You can install Private Help Server (PHS) into Tomcat server using three different methods:

- Using the Installer file
- Using the Silent mode
- Using the Command line

See *PHS - Administration Guide* for detailed steps on private help installation on your system. PHS is useful for environments with no or limited access to Internet connectivity. To integrate private help server into Advanced Workflow complete the following steps:

1. Install and configure OpenJDK version 11.0.14 or later on the server.
2. Set up a server in your environment for deploying the help system.
3. Setup the PHS on the server. See *Installing the OpenText Private Help Server* section in the *PHS - Administration guide*. Save the PHS root URL (http://<<*host-name*>>:<<*port-number*>>) for future reference.
4. Test the PHS configuration on the help server. See Testing a Private Help Server installation section in the *PHS - Administration guide*.
5. Deploy the Advanced Workflow help files to the server as described in the Adding product online help to the Private Help section in the *PHS - Administration guide*.
6. Update the `DocumentumAdvancedWorkflow.ini` file with the PHS root URL noted earlier. For example:

```
-Doffline.help.server.url=http://host_ip:port
```

where `http://host_ip:port` is the PHS root URL.

For detailed information about setting up PHS, see *PHS - Administration* guide.

Chapter 4

Migrating Applications

4.1 Migrating an application from an earlier version to a current version of Advanced Workflow

You can migrate an Advanced Workflow application created using previous release to the latest Advanced Workflow. This section describes how to migrate applications to the latest Advanced Workflow version.

Before you upgrade, perform all the relevant tasks described in "Licensing OpenText Documentum CM" on page 69.

To migrate an application, you must first export the application using the previous version of Advanced Workflow and then import it into the latest Advanced Workflow instance.

When you upgrade from an existing environment with Advanced Workflow to the latest environment and create an Advanced Workflow application and deploy it, the existing artifacts in the repository from previous Advanced Workflow deployment are deleted leading to loss of data. To avoid this issue, you should first export the application using the old version of Advanced Workflow and import the application to the latest version of Advanced Workflow.

4.2 Migrating an Advanced Workflow application

1. In the older version of Advanced Workflow, navigate to **Home** page, select an application and click **Export**.
2. In the **Export** dialog box, click **Browse** and select a destination folder.
3. Select **Include all dependent projects** option and click **Finish**.
4. Import the exported application to the current version of Advanced Workflow.
    a. From the Advanced Workflow **Home** page, click **Import Application**.
    b. Click **Browse** and select the source folder.
    c. Select the required options for the application and all the project.
    d. Select **Import a copy** to copy the selected projects to the root folder of your new application.
    e. Click **Next**.
    f. Click **Finish**.

Chapter 5

Advanced Workflow language packs

5.1 Deploying Advanced Workflow language packs

The following procedures describe how to download and deploy language packs for Advanced Workflow.

1. Download the `DocumentumAdvancedWorkflow_<version>.zip` file from the OpenText MySupport (https://support.opentext.com). Extract the contents of the `DocumentumAdvancedWorkflow_<version>.zip` file to a folder in any directory.
2. Download one or more language pack ZIP files from the OpenText MySupport (https://support.opentext.com). Extract the contents of each language pack ZIP file to the same folder as in the previous step.
3. Do one of the following:
   - If Advanced Workflow is installed on a localized operating system, double-click **DocumentumAdvancedWorkflow.exe** in the `DocumentumAdvancedWorkflow` folder to launch the localized Advanced Workflow.
   - If Advanced Workflow is not installed on a localized operating system, in the command prompt window, change the current working directory to the directory where you extracted the contents of the Advanced Workflow ZIP file.

     a. Type the following command:

     ```
     cd DocumentumAdvancedWorkflow
     ```

     to point to the folder that has the executable file for Advanced Workflow.

     b. Type the following command:

     ```
     DocumentumAdvancedWorkflow -nl <lang>
     ```

     to launch the localized Advanced Workflow, where <*lang*> is the abbreviation of the language pack.

     For example, the command `xcpdesigner -nl zh` launches a localized Advanced Workflow which in this case is in the Chinese language interface.

     The display language for Advanced Workflow depends on the regional setting of the operating system. Launching Advanced Workflow using the command option overrides the regional setting and displays Advanced Workflow in the language you specify in the command.

     For example, if you install a French language pack in an operating system with English as the base language but with the French regional setting, double-click **DocumentumAdvancedWorkflow.exe** to launch Advanced

Workflow in French. However, if you want to run Advanced Workflow in English, launch it using the command `DocumentumAdvancedWorkflow -nl en` from the command prompt window.

Chapter 6

Uninstalling an Application

6.1 Manually uninstalling application data

Applications cannot share namespaces in the same environment. When you finish testing an application in a development environment, uninstall the application so the namespace can be reused. If you move an application from one production environment to another production environment, uninstall the application from the original environment.

Uninstalling an application involves removing application data from the OpenText Documentum CM repository.

6.2 Removing data from the repository

Removing data from the OpenText Documentum CM repository consists of retrieving a list of artifact bundles in the application and then removing the following artifacts:

- Business objects, folders, and content objects
- Relations
- Java modules
- Java Services
- Processes and activities
- Groups
- Parameters and endpoints
- Permission sets
- Historical data
- Type fragment

When you remove data from the repository, you use the Documentum Administrator DQL Editor and API Tester to run commands that retrieve and remove the data. The *OpenText Documentum Server Administration and Configuration Guide* contains more information on Documentum Administrator.

6.2.1 Retrieving a list of application artifact bundles

Use the Documentum Administrator DQL Editor to run the commands in this procedure.

1. Retrieve a list of all instances of a deployed application by running the following command:

   ```
   Select r_object_id, r_modify_date from xcp_artifact_bundle (all) where
   object_name='<$appname>' order by r_modify_date asc
   ```

   where <*$appname*> is the name of the application.

   The query returns a list of object IDs for each deployed instance of the application. The most recently deployed instance is at the bottom of the list. When you remove application data, you remove the data associated with each instance.

2. Retrieve a list of bundle libraries for an application instance by running:

   ```
   Select bundlelibrary from xcp_artifact_bundle where r_object_id='<$bundleobjectid>'
   ```

   where <*$bundleobjectid*> is an object ID for an instance of application or library or project returned in the previous step.

3. Repeat the previous step for each library bundle for an instance until you have retrieved the object IDs for all library bundles associated with the instance.

   For example, after you run the command with the object ID for an application instance, the query returns library bundles 080004d180005e47 and 080004d180005e46. Run the command again using 080004d180005e47 as the value for <*$bundleobjectid*> and then run the command again using 080004d180005e46 as the value for <*$bundleobjectid*>.

4. For a library bundle, retrieve the object IDs for the artifacts that have been translated into repository objects.

   Do not retrieve the IDs for bundles shared between applications, the xcpcore bundle, and the xcpcommons bundle. Retrieve the IDs by running the following command:

   ```
   Select installedobjectid from xcp_artifact_bundle where r_object_id='<
   $bundleobjectid>'
   ```

   The remaining sections in this chapter explain how to delete data associated with the object IDs that you retrieve in this step.

5. Repeat step 2 through step 4 for each application instance in the query results from step 1. The resultant of this step provides object IDs for all bundle libraries.

6.2.2 Removing application data

6.2.2.1 Removing relation data

1. Run the following command from the Documentum Administrator DQL Editor to retrieve the application namespace:

```
Select namespace from xcp_artifact_bundle where r_object_id='<$bundleobjectid>'
```

2. Run the following command from the Documentum Administrator DQL Editor to retrieve the dm_relation_type object ID for the namespace:

```
Select r_object_id,relation_name from dm_relation_type where relation_name like
'<namespace>_%'
```

3. Run the following command from the Documentum Administrator API Tester to remove the objects associated with the dm_relation_type:

```
destroy,c,<object_ID>
```

4. Run the following command from the Documentum Administrator DQL Editor to retrieve the dm_type that corresponds to the relation:

```
Select r_object_id,name from dm_type where name like '%<namespace>_%'
```

5. Run the following command from the Documentum Administrator API Tester to delete dm_type objects from the repository:

```
query,c,delete <name_of_type> objects
```

6. Run the following command from the Documentum Administrator API Tester to remove the type definition:

```
query,c,drop type <name_of_type>
```

6.2.2.2 Removing Java module and Java service data

1. Run the following command from the Documentum Administrator DQL Editor to retrieve a list of dmc_module object names and object IDs:

```
Select r_object_id, object_name, r_folder_path from dmc_module where r_object_id in
(select installedobjectid from xcp_artifact_bundle where r_object_id='<
$bundleobjectid>')
```

2. Run the following command from the Documentum Administrator DQL Editor to retrieve a list of dmc_jar objects for a dmc_module object:

```
Select r_object_id, object_name from dmc_jar(all) where FOLDER('/System/Modules/
<module_object_name>'
```

3. Run the following command from the Documentum Administrator API Tester to unlink each dmc_jar object from the dmc_module object:

```
unlink,c,<object_id>,/System/Modules/<module_object_name>'
```

4. Run the following command from the Documentum Administrator API Tester to remove each dmc_jar object from the repository:

```
destroy,c,<object_id>
```

5. Repeat step 2 through step 4 for each dmc_jar object.
6. Run the following command from the Documentum Administrator API Tester to remove each dmc_module object from the repository:

```
destroy,c,<object_id of dmc_module>
```

6.2.2.3 Removing process data

1. Run the following command from the Documentum Administrator DQL Editor to retrieve a list of processes:

```
Select r_object_id, object_name from dm_process where r_object_id in (select
installedobjectid from xcp_artifact_bundle where r_object_id='<$bundleobjectid>')
```

2. Run the following command from the Documentum Administrator API Tester to abort workflows for a process:

```
uninstall,c,<process_id>
```

3. Run the following command from the Documentum Administrator API Tester to re-install the process with the option to abort halted workflows:

```
install,c,<process_id>,T,F
```

4. Run the following command from the Documentum Administrator DQL Editor to retrieve a list of activities for the process:

```
Select r_act_def_id from dm_process where r_object_id='<process_id>'
```

5. Run the following command from the Documentum Administrator API Tester to uninstall the process from the repository:

```
uinstall,c,<process_id>
```

6. Run the following command from the Documentum Administrator API Tester to invalidate the process:

```
invalidate,c,<process_id>
```

7. Run the following command from the Documentum Administrator API Tester to remove the process from the repository:

```
destroy,c,<process_id>
```

8. Run the following command from the Documentum Administrator API Tester to uninstall each activity:

```
uninstall,c,<act_id>
```

9. Run the following command from the Documentum Administrator API Tester to invalidate each activity:

```
invalidate,c,<act_id>
```

10. Run the following command from the Documentum Administrator API Tester to remove each activity:

```
destroy,c,<act_id>
```

11. Repeat step 2 through step 10 for each process returned in step 1.
12. Remove aborted workflows and orphaned structured data type (SDT) objects by using Documentum Administrator to run a dm_DMClean job with the following argument:

```
'-clean_aborted_wf TRUE' and '-clean_wf_template TRUE'
```

6.2.2.4 Removing group data

1. Run the following command from the Documentum Administrator DQL Editor to retrieve a list of groups for application bundles:

```
Select r_object_id, group_name from dm_group where r_object_id in (select
installedobjectid from xcp_artifact_bundle where r_object_id='<$bundleobjectid>')
```

2. Run the following command from the Documentum Administrator API Tester to remove each group from the repository:

```
destroy,c,<group_id>
```

where <*group_id*> is the object ID for a group.

6.2.2.5 Removing parameter and endpoint data

1. Run the following command from the Documentum Administrator DQL Editor to retrieve the namespace of the application:

```
Select namespace from xcp_artifact_bundle where r_object_id='<$bundleobjectid>'
```

2. Run the following command from the Documentum Administrator API Tester to remove all objects related to the namespace:

```
query,c,delete dmc_xcp_app_config objects where namespace='<namespace>'
```

6.2.2.6 Removing permission set data

1. Run the following command from the Documentum Administrator DQL Editor to retrieve a list of dm_acl objects:

```
Select r_object_id, object_name from dm_acl where r_object_id in (select
installedobjectid from xcp_artifact_bundle where r_object_id='<$bundleobjectid>')
```

2. Run the following command from the Documentum Administrator API Tester to remove each dm_acl object:

```
destroy,c,<acl_id>
```