software AG

Deploying to webMethods Integration Cloud

Version 5.0

October 2018

WEBMETHODS

Table of Contents

Deploying to webMethods Integration Cloud Documentation

This document explains how to deploy user-created packages and configuration assets that reside within on-premises runtimes or repositories to webMethods Integration Cloud.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies service names and locations in the format *folder.subfolder.service*, APIs, Java classes, methods, properties.
Italic	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
Monospace font	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at "http://documentation.softwareag.com". The site requires credentials for Software AG's Product Support site Empower. If you do not have Empower credentials, you must use the TECHcommunity website.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to "empower@softwareag.com" with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at "https://empower.softwareag.com/".

You can find product information on the Software AG Empower Product Support website at "https://empower.softwareag.com".

To submit feature/enhancement requests, get information about product availability, and download products, go to "Products".

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the "Knowledge Center".

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at "https://empower.softwareag.com/public_directory.asp" and give us a call.

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at "http://techcommunity.softwareag.com". You can:

■ Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.

■ Access articles, code samples, demos, and tutorials.

■ Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.

■ Link to external websites that discuss open standards and web technology.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

Deploying to Integration Cloud

Cloud deployment is the process of deploying user-created packages and configurations that reside within on-premise runtimes or repositories to webMethods Integration Cloud. Using Software AG Designer you can seamlessly deploy your on-premise Integration Server packages and configuration assets to solutions present on Integration Cloud.

> **Note:** In the cloud deployment context, configuration assets are limited to Integration Server and Universal Messaging configurations.

Software AG Designer allows you to deploy the Integration Server packages or configuration assets that you have created, verified, and tested on on-premise Integration Server or Universal Messaging to Integration Cloud. When you initiate the deployment from Designer, Integration Server packages and configuration assets are built from Integration Server and Command Central respectively, and are published to an asset repository present in Designer.

After performing variable substitutions to make the on-premise configuration data compatible for cloud deployment, you can publish the packages and configurations from the asset repository to a remote repository provisioned for the tenant on Integration Cloud.

Platform Manager plug-ins for Integration Server and Universal Messaging continuously monitor the remote repository on Integration Cloud. When a fresh deployment is detected, Platform Manager plug-ins deploy the Integration Server packages and configuration assets to respective runtime instances on Integration Cloud.

> **Note:** Deploying assets to Integration Cloud using Designer eliminates the previous need to install or invoke webMethods Deployer on cloud for the purpose of cloud deployment.

The following figure provides a high-level basic overview of the process involved in deploying on-premise Integration Server packages and configuration assets to Integration Cloud.
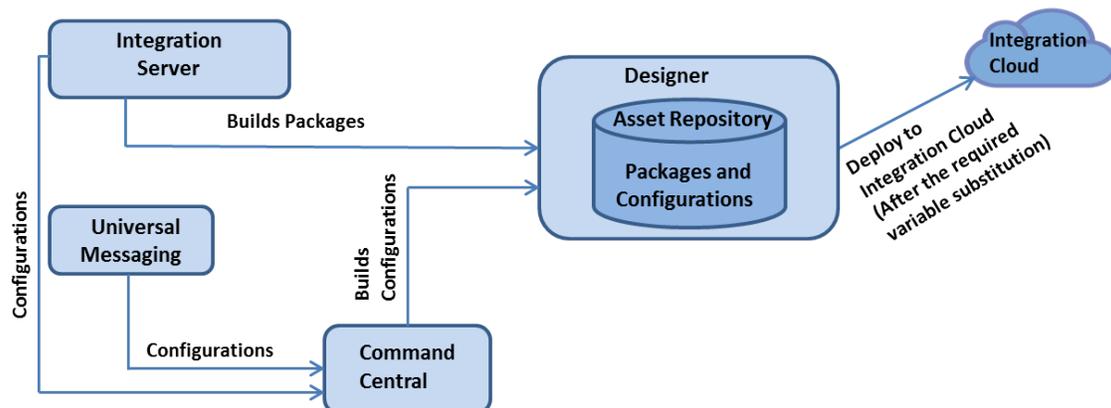
Getting Started

The following table briefly describes the high-level tasks to get started with Cloud Deployment.

Tasks	References
What is Cloud Deployment?	See the *Deploying to Integration Cloud* section. **Note:** See this "video" for information on how to perform Cloud Deployment.
License and installation information.	Install the relevant products as mentioned in the "Installation Information" on page 17 section. Further, you must have a tenant in webMethods Integration Cloud and a license for Cloud Deployment. Ensure that you download the latest version of Software AG Designer to use the new features available for Cloud Deployment.
Assets and configurations that can be deployed to Integration Cloud.	See "Deployable Assets and Configurations" on page 13.
Create a Solution in Integration Cloud.	See the *Cloud Deployment* section in the *webMethods Integration Cloud Help* for information on how to create a Solution.
Connect to Integration Cloud.	See "Connecting to Integration Cloud for Cloud Deployment" on page 21.
View and publish assets and configurations to the solution created in Integration Cloud using any of the Cloud Deployment approaches.	See "Cloud Deployment Approaches" on page 19.
View published assets and configurations in Integration Cloud.	See the *Cloud Deployment* section in the *webMethods Integration Cloud Help*.

Tasks	References
Promote the assets and configurations to different stages in Integration Cloud.	See the *Cloud Deployment* section in the *webMethods Integration Cloud Help*.
View and run services in Integration Cloud.	See the *Cloud Deployment* section in the *webMethods Integration Cloud Help*.
View the solution landscape, configure webMethods Integration Server service access settings, administer, and restart webMethods Integration Server in Integration Cloud.	See the *Cloud Deployment* section in the *webMethods Integration Cloud Help*.
Monitor the health and availability of the deployed solutions and run-time instances, alerts, and alert statuses in Integration Cloud.	See the *Cloud Deployment* section in the *webMethods Integration Cloud Help*.
Best Practices and General Information.	See "Best Practices and General Information" on page 55.

Deployable Assets and Configurations

The following tables provide information on the deployable assets and configurations.

Table 1. Integration Server

Configuration/Asset Type	Asset/Configuration
ACLs	Configuration
Groups	Configuration
Consumer Web service endpoint alias	Configuration
Extended Settings * (See "Best Practices and General Information" on page 55)	Configuration
Global Variables	Configuration
JMS aliases	Configuration
JNDI aliases	Configuration
Keystores	Configuration
Provider Web service endpoint alias	Configuration
Resources Settings *	Configuration
SMTP settings *	Configuration
Truststores	Configuration
Users	Configuration
webMethods messaging	Configuration
Flow Service	Asset

Configuration/Asset Type	Asset/Configuration
Java Service	Asset
Map Service	Asset
OData Service	Asset
Adapter Service	Asset
.NET Service	Asset
C Service	Asset
Adapter Notification	Asset
Document Type	Asset
Flat File Dictionary	Asset
Flat File Schema	Asset
JMS Trigger	Asset
REST API Descriptor	Asset
REST Resource	Asset
Schema	Asset
Specification	Asset
Web Service Descriptor	Asset
webMethods Messaging Trigger	Asset
XML Document Type	Asset
XSLT Service	Asset

* - Restart Required

Table 2. Universal Messaging

Configuration Type	Asset/Configuration
Channels	Transient
	Simple
	Reliable
	Persistent
	Mixed
	Off-Heap
	Paged
Cluster	UM Cluster
JNDI Connection Factories	Connection Factory>Shared Durable Type
	Connection Factory> Non Shared Durable Type
	TopicConnection Factory>Shared Durable Type
	TopicConnection Factory>Non Shared Durable Type
	Queue Connection Factory
	XA Connection Factory
JNDI Destinations	Topic
	Queue
Memory *	JAVA_MIN_MEM JAVA_MAX_MEM
Queues	Transient
	Simple

Configuration Type	Asset/Configuration
	Reliable
	Persistent
	Mixed
	Off-Heap
	Paged
Users	All Custom Users
Zones	UM Zone

* - Restart Required

Installation Information

Install Software AG Command Central using the Command Central bootstrapper or provision a Command Central container from the Docker Store. For instructions, see the Command Central documentation.

Install the following products into a different directory or on a different machine, using the Software AG Installer or Command Central:

- webMethods Asset Build Environment

- Service Development in Software AG Designer with Local Version Control System Integration feature and webMethods Unit Test Framework.

- webMethods Integration Server with webMethods Unit Test Framework.

- Software AG Universal Messaging. All child nodes except Enablement for Cloud Foundry and webMethods Broker to UM Migration Utility.

If you want to use an external RDBMS for webMethods Integration Server, install webMethods Database Component Configurator on any machine on the same network as your database server, create the database components for webMethods Integration Server, and then connect webMethods Integration Server to the database components.

For detailed instructions, see the following documents:

- *Installing Software AG Products* document.

- If you are using Software AG Installer to install, see the *Using Software AG Installer* document.

- If you are using Command Central to install, see the *Software AG Command Central Help*.

Cloud Deployment Approaches

Choose the following approaches to publish user-created packages and configurations to a solution created in webMethods Integration Cloud. See the *Cloud Deployment* section in the *webMethods Integration Cloud Help* for information on how to create a solution in webMethods Integration Cloud.

Using Software AG Designer

■ "Package Navigator" on page 25: Use the **Package Navigator** view in Software AG Designer to deploy webMethods Integration Server packages and configuration assets to webMethods Integration Cloud.

■ "Asset Repository" on page 30: Use the **Asset Repository** view in the Service Development perspective of Software AG Designer to deploy webMethods Integration Server packages and configuration assets to webMethods Integration Cloud.

■ "Landscape Navigator" on page 36: Use the **Landscape Navigator** view in the Service Development perspective of Software AG Designer to deploy customized configuration templates to webMethods Integration Cloud. "Export the configurations" on page 43 into a Software AG Designer configuration project and then build and deploy this project to webMethods Integration Cloud. For continuous integration purposes, these projects can be stored in a Version Control System (VCS) location, built through webMethods Asset Build Environment, and then deployed to webMethods Integration Cloud.

Using "Continuous Integration and Continuous Deployment" on page 49.

Connecting to Integration Cloud for Cloud Deployment

Using Designer you can deploy your on-premises Integration Server packages and configuration assets to Integration Cloud. Before deploying the packages and configuration assets, you must configure a connection to Integration Cloud.

Before you configure a connection to Integration Cloud, ensure that the following criteria are met:

- A valid URL exists to connect to Integration Cloud.

- A valid user account is created on Integration Cloud.

Configuring a Connection to Integration Cloud

You need to provide specific information in Designer for initiating a connection to Integration Cloud. Designer saves this information in a connection configuration. You can add, edit, and update connection configurations in Designer.

Adding a Connection Configuration for Integration Cloud

Using Designer you can add connections to different instances of Integration Cloud.

To add a connection configuration for Integration Cloud

1. In Designer, select **Window > Preferences**.

2. In the preferences navigation tree, select **Software AG > Integration Cloud**.

3. Click **Add**.

4. In the **Add connection configuration** dialog box, enter the following information:

Field	Description
Name	The name to use for the Integration Cloud connection configuration.
	Note: The name cannot contain control characters, special characters, and characters outside of the basic ASCII character set, such as multi-byte characters.
URL	URL of the Integration Cloud host to which Designer is to connect. For example, `https://<sub-domain>.<domain-name>.<domain-suffix>`, for example, https://mysubdomain.webmethodscloud.com.
User	The user name for an account on Integration Cloud.
Password	The password for the specified **User**.

Field	Description
Save password (in the Eclipse secure storage)	Indicates whether the password for the specified user account should be saved in Eclipse secure storage. Integration Cloud uses this password from the Eclipse secure storage whenever user authorization is required. If you want to save the password in Eclipse secure storage, select this check box.
	If you decide not to save the password in Eclipse secure storage, you must specify your password each time your user authorization is required for connecting to Integration Cloud.

5. To verify whether Integration Cloud can be accessed by using the specified information, click **Test**.

6. To store the connection configuration details, click **OK**.

   A connection configuration is added to the Connections page with the specified details. The first connection configuration that you create is automatically marked as default. This default configuration is indicated with a check mark on the Connections page. Designer always uses the default connection configuration for Integration Cloud. If there are multiple connections configured, you can select the connection that you want to make it as default.

Editing a Connection Configuration for Integration Cloud

You can edit a connection configuration for Integration Cloud if there are any changes in the configuration values.

To edit a connection configuration for Integration Cloud

1. In Designer, select **Window > Preferences**.

2. In the preferences navigation tree, select **Software AG > Integration Cloud**.

3. Click **Edit**.

4. Enter new values in the connection configuration fields you want to change.

5. In the **Edit Connection Configuration** dialog box, click **OK**.

6. In the **Connections** page, click **OK**.

Removing a Connection Configuration for Integration Cloud

You can remove connection configurations for Integration Cloud one at a time from the **Connections** page.

To remove a connection configuration for Integration Cloud

1. In Designer, select **Window > Preferences**

2. In the preferences navigation tree, select **Software AG > Integration Cloud**.

3. Select the Integration Cloud connection configuration you want to remove.

4. Click **Remove**.

   Designer prompts you to confirm that you want to remove the selected connection configuration.

5. Click **OK**.

Deploying Packages and Configuration Assets to Integration Cloud

Using Designer you can deploy Integration Server packages and configuration assets to Integration Cloud.

Before deploying packages and configuration assets to Integration Cloud, ensure the following:

■ Designer is configured to connect to Integration Cloud using the Integration Cloud preference page. For more information on configuring a connection to Integration Cloud, see Connecting to Integration Cloud for Cloud Deployment chapter.

■ Designer is configured to connect to Command Central. Use the Landscape Navigator view to connect to Command Central. For more information on Landscape Navigator view and connecting to Command Central, see "Using Landscape Navigator to View Runtime Configurations" on page 35.

■ Command Central is connected to Integration Server and Universal Messaging instances.

> **Note:** If you plan to deploy only Integration Server packages and not the Universal Messaging and Integration Server configurations, you do not need to connect to Command Central.

To deploy assets to Integration Cloud

> **Note:** The following procedure explains how to deploy packages and configuration assets only to Integration Cloud. If you want to deploy, refresh,or remove the packages and configuration assets from Integration Cloud, use the Asset Repository view in Designer.

1. In Package Navigator view, select the Integration Server package that you want to deploy to Integration Cloud.

2. Right-click the package and click **Deploy to Cloud**.

3. In the **Publish Assets to Cloud** dialog box, select the package to deploy. You can select multiple packages.

   If you want to publish the configuration assets along with the selected Integration Server package, select **Include runtime configurations**, click **Next**, and go to step 4. Otherwise, click **Next** and go to step 5.

   Designer fetches the configuration data from Command Central.

4. In the **Confirm selected configurations** dialog box, select the configuration that you want to deploy and click **Next**.

5. In the **Integration Server package variable substitution** dialog box, select the package and edit the value of the property that you want to modify before deploying to cloud. Click **Next**.

6. In the **Select Cloud Solution** dialog box, select the solution on Integration Cloud to which you want to deploy the assets and click **Finish**.

   Designer deploys the assets and configurations to a remote repository provisioned for the tenant on Integration Cloud.

Deployment of Adapter for JDBC Assets

To deploy Adapter for JDBC assets to Integration Cloud, do the following:

1. Create a separate package, *<package_Name>*, for deploying the database specific driver jar files that are part of the on-premises installation.

2. Copy the jar files from the on-premises installation `<webM_Home>\`Integration Server`\instances\<instance_Name>\packages\WmJDBCAdapter\code\jars`, and place the driver jar files in the `<package_Name>/code/jars/static` folder.

3. Add the package dependency of the asset package to the package *<package_Name>*, where database specific driver jar files are placed.

   > **Note:** Maintain the package dependency for assets scattered across packages. For more information on package dependency, see the *webMethods Adapter for JDBC Installation and User's Guide*.

4. To enable the asset (connection, polling notification) on cloud, ensure that the **State After Deployment** is enabled during deployment from Software AG Designer. The state of the asset (connection, polling notifications) should be disabled on-premises before deploying to cloud. To deploy Adapter for JDBC to Integration Cloud, see "Deploying Packages and Configuration Assets to Integration Cloud" on page 25.

   > **Note:** ■ When using VPN and connecting to an on-premises database, you have to variable substitute the serverName to IP address of the server, as the host name of the server cannot be resolved.

Limitations

■ You can use only Type 4 JDBC drivers for cloud deployment.

■ Kerberos authentication is not supported.

■ Data type configuration file editing is not supported.

Asset Repository

Asset Repository is a binary repository that provides components for managing user assets . Asset Repository enables the process of repository-based deployment, in which runtime servers pull assets from a repository. You can build Integration Server packages and configuration assets and add to the repository making the assets available for deployment to Integration Cloud. Asset Repository is installed along with Designer using the Software AG Installer.

The Asset Repository view in Designer allows you to build, deploy, refresh, remove, or retract the assets.

Opening the Asset Repository View

To open the Asset Repository view

1. In the Service Development perspective of Designer, select **Window > Show View > Other**.

2. In the **Show View** dialog box, select **Software AG Service Development > Asset Repository**.

3. Click **OK**.

Building Packages and Assets in Asset Repository View

If the Asset Repository view does not already display the list of Integration Server packages and configuration assets, you can build them.

To build packages and configuration assets

1. In Asset Repository view, click the View Menu button ▽.

2. Select **Build**.

   The Asset Repository view displays all the newly created Integration Server packages and configuration assets.

Deploying Packages and Assets from Asset Repository View

You can deploy Integration Server package or configuration assets using the Asset Repository view.

To deploy packages and configuration assets from Asset Repository View

1. In Asset Repository view, select the Integration Server package or configuration assets that you want to deploy to Integration Cloud.

2. Right-click the package and click **Deploy to Cloud**.

3. In the **Publish Assets to Cloud** dialog box, select the package to deploy.

If you want to publish the configuration assets along with the selected Integration Server package, select **Include runtime configurations**, click **Next**, and go to step 4. Otherwise, click **Next** and go to step 5.

4. In the **Confirm selected configurations** dialog box, select the configuration that you want to deploy and click **Next**.

   Designer fetches the configuration data from Software AG Command Central.

5. In the **Integration Server package variable substitution** dialog box, select the package and edit the value of the property that you want modify before deploying to cloud. Click **Next**.

6. In the **Select Cloud Solution** dialog box, select the solution on Integration Cloud to which you want to deploy the assets and click **Finish**.

   Designer deploys the assets and configurations to a remote repository provisioned for the tenant on Integration Cloud.

Retracting Packages and Assets

Using Designer you can retract Integration Server packages, configuration assets, or both from Integration Cloud.

To retract packages and configuration assets

1. In Asset Repository view, select the Integration Server package or configuration assets that you want to retract.

2. Right-click and select **Retract**.

   Designer retracts the packages and configuration assets fromIntegration Cloud.

Deleting Packages and Assets

You can delete Integration Server packages or configuration assets from the Asset Repository view and from Integration Cloud. You can do this to remove packages and configuration assets with incorrect variable substitution or remove old versions of them from Integration Cloud.

To delete packages and configuration assets

1. In Asset Repository view, select the Integration Server package or configuration assets that you want to delete.

2. Right-click and select **Remove**.

   Designer removes the Integration Server packages or configuration assets from Asset Repository view and Integration Cloud.

Deploying Command Central Assets to Integration Cloud

Command Central can export configuration properties for Software AG run-time components as YAML templates. Configuration properties are exported as assets using Software AG Designer, stored in the Landscape Asset Repository, and deployed to product instances, running on webMethods Integration Cloud.

To create Command Central assets and deploy them to Integration Cloud, in Designer:

1. Connect to Command Central.

2. Select and export the configurations to be deployed.

3. Edit the YAML configurations.

4. Deploy the configuration assets to Integration Cloud.

After the assets are deployed to Integration Cloud, a Platform Manager that runs in a container on Integration Cloud listens and pulls assets when it detects changes in an asset repository. Then Platform Manager deploys the assets automatically on product instances in Integration Cloud. When Platform Manager pulls assets from the remote asset repository, it deploys the assets to Integration Cloud by placing dependencies first. For example, if asset "A" has a dependency on asset "B", the dependent asset "B" is deployed before asset "A".

Using Landscape Navigator to View Runtime Configurations

Using Landscape Navigator view you can connect to Command Central and view list of registered installations and runtime instances. For more information about connecting to Command Central see the *Creating Command Central Definitions* section.

What Does the Landscape Navigator View Contain?

You can use Landscape Navigator view and Configuration Browser views to display the list of installations that are registered in Command Central. The available run-time instances are listed in the view for each installation and these available run-time instances are available for deploy to Integration Cloud.

Landscape Navigator view provides an option to connect to a local or remote Command Central server. To view the contents of installations that are registered in Command Central, you must configure a connection to Command Central and then establish a connection.

You can find Landscape Navigator and Configuration Browser views under the **Windows > Show View > Other > Software AG Command Central**.

Creating Command Central Definitions

Command Central definitions create the connection between Designer and Command Central. The definition for a server contains the connection information needed to establish a connection with the Command Central server.

To create a Command Central definition

1. On the Landscape Navigator toolbar, click .

2. In the New Command Central Connection dialog box, enter the details for connecting to the Command Central server as described in following table:

Field	Description
Connection name	The connection name.
Command Central Location	Choose one of the following options to specify the location: ■ **URL:** The URL for Command Central connection. For example, https://<Command_Central_host>:<Command_Central_port>/cce. ■ **Installation Directory:** The local Software AG product installation

Field	Description
	directory where Command Central is installed.

3. Under **Authentication**, enter the details as described in the following table:

Field	Description
Username	Username to use for authentication on the Command Central.
Password	The password to use for authentication on the Command Central.
Connection timeout (s):	The maximum seconds Designer waits for response from Command Central.

4. Click **Save** password.

   Designer stores the password in Eclipse secure storage location.

5. Click **Test Connection** to ping the Command Central server and to verify that the connection is working.

6. Click **Finish**.

   Designer refreshes the Landscape Navigator view and displays the new connection.

Editing Command Central Definition

If there are changes in the associated Command Central server that require you to update the server definition, edit the Command Central connection properties. For example, to add or edit the username and password, you must edit the Command Central Connection page to reflect that change.

To edit a Command Central definition

1. Open the Landscape Navigator view.

2. Select the connection you want to edit.

3. Right-click the connection and select **Edit Command Central Connection**.

4. Edit the connection properties as needed.

Deleting a Command Central Definition

To delete a Command Central definition

1. Open the Landscape Navigator view.

2. Select the connection that you want to delete.

3. Right-click the connection and select **Delete Command Central Connection**.

4. Click **Yes** to confirm the deletion from the view.

What Does the Configuration Browser View Contain?

Designer offers a detailed view of run-time specific configuration in Configuration Browser view. Designer retrieves information about a specific run-time component selected in Landscape Navigator.

The following table describes the runtime component information that is displayed in the Configuration Browser view.

Name	Description
Name	Display name of the configuration instance.
Id	Id of the runtime instance.
Product	Product to which the runtime instance belongs.
Category	Configuration category to which the runtime instance belongs.
Deployable	Whether or not the asset is deployable to Integration Cloud.
Description	Description of the configuration instance.

Browsing Runtime Instances and Configurations

After you connect to Command Central, you can use the Landscape Navigator view to browse the registered installations, for available runtimes and corresponding configurations.

When you expand an installation node, you can view the list of the runtime instances in that installation. You can also view the name and the runtime status as ONLINE, UNKNOWN, FAILED, OFFLINE, and so on. Upon expanding the runtime instances, the list of configurations are displayed. You can view the configuration data for each configuration instance in Properties view, under Data tab.

Deploying Customized Configuration to the Integration Cloud

Using Designer you can deploy customized configuration templates to Integration Cloud. You can use Landscape Navigator view to export the configuration into a Designer configuration project. Designer can be used to build and deploy this project to Integration Cloud. For continuous integration purpose, these projects can be stored in Version Control System location and built through Asset Build Environment and deployed to Integration Cloud.

The following figure depicts the processes involved in exporting deployable configurations.
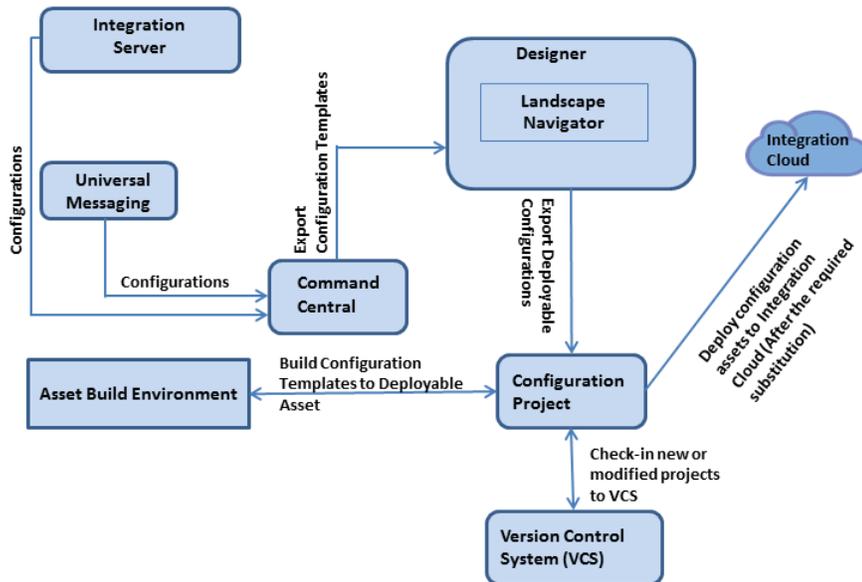
Exporting Deployable Configurations

You can export one or more runtime configurations to a YAML template file or as a deployable asset. You can export only deployable configurations. Deployable configurations are those configurations that can be deployed to Integration Cloud. Configuration Browser view lets you identify these deployable configurations assets in the list with the icon.

You can export deployable configuration from a runtime instance as YAML template within a configuration project. For more information about configuration projects, see "Using Configuration Project" on page 45. This YAML template can be manually edited and build as deployable asset. Alternatively, you can export configurations as deployable asset, which can be readily deployed to cloud solutions.

To export a configuration

1. On the Landscape Navigator view , select one or more configurations from a runtime to export.

2. Right-click and select **Export Configurations**.

3. Under **Location**, provide the following information for the configuration project:

   ■ Project: The project name for configuration project. You can assign a project in either of the following ways:

      ■ Browse and select an existing configuration template project.

      ■ Specify a new configuration project resource. For more information about creating a new project resource, see "Using Configuration Project" on page 45.

4. In the **Name** field, select a template name.

   The template contains the data for the selected configurations.

5. Under **Format**, select any one of the following values:

   ■ YAML source: To export configurations to YAML template. You can manually edit and build YAML template as a deployable asset.

   ■ Deployable Asset : To export configurations that are readily deployable for cloud solutions.

6. Under **Parameterization**, select **Enable Auto Parameterization in template** check box to add parameterization as needed.

7. In the **Parameterization** field, type the name of a parameter used to identify parameterization.

   An identifier can include lowercase a-z, uppercase A-Z, digits, dollar sign, and underscore. An identifier cannot start with a digit and cannot include any other special characters.

8. Click **Next**.

   By default, Designer displays deployable configurations that are included for export. To view all the configurations, clear the **View deployable configurations only** check box.

9. Select the configuration contents to confirm the set of configurations to export.

10. Click **Finish**.

    The exported YAML template file will open in an editor. You can view the exported contents using the Project Explorer view.

Using Configuration Project

The configuration project is used for adding, editing, and building configuration templates exported from Integration Server and Universal Messaging runtimes. This project is pre-configured to build with Asset Build Environment which is used to prepare a deployable configuration asset from YAML source.

Before you configure a project, specify the build path entries to be used as the default path. For more information on specifying build path entries, see the Build Path Preferences page in Designer.

To create a configuration project

1. In Designer, navigate to **File > New > Project > SoftwareAG**.

2. Under SoftwareAG element, select **Asset Builder** and click **Configuration template**.

3. Click **Next**.

   Designer displays New Configuration Project dialog box.

4. In the **Project name** field, type the new name for the project.

5. Clear the **Use default location** check box, to choose a different location.

   By default Designer uses the Workspace root location. For example, (c: \<username> \workspace).

6. Click **Browse** adjacent to the **Location** field. Navigate to and select the location directory.

7. In the **Project Settings**, edit the default project structure preference to use for the source and output of builds.

8. Under **Working sets**, select check box **Add project to working sets** to specify a working sets.

9. Click **Finish**.

   Designer creates a configuration project.

Deploy Configuration Project

To deploy configuration assets to Integration Cloud

1. In **Project Explorer** view, select the configuration project that you want to deploy to Integration Cloud.

2. Right-click the project and click **Deploy Configurations to Cloud**.

3. In the **Configuration Templates** dialog box, select the templates to deploy. Click **Next**.

4. In the **Configuration Template Variable Substitution** dialog box, select the template and edit the values of the property that you want to modify before deploying to cloud. Click **Next**.

5. In the **Select Cloud Solution** dialog box, select the solution on Integration Cloud to which you want to deploy the assets and click **Finish**.

   Designer deploys the configurations assets to a remote repository provisioned for the tenant on Integration Cloud.

Continuous Build and Publish

You can *build* user-created assets and configurations using webMethods Asset Build Environment (ABE), *retrieve* those assets and configurations from a VCS by using ABE or an automated tool like Jenkins, and then *deploy* those assets and configurations to the Integration Cloud*Development Stage* by using ABE.

The following figure depicts the processes involved in the basic build and publish mechanism:

BASIC BUILD AND PUBLISH
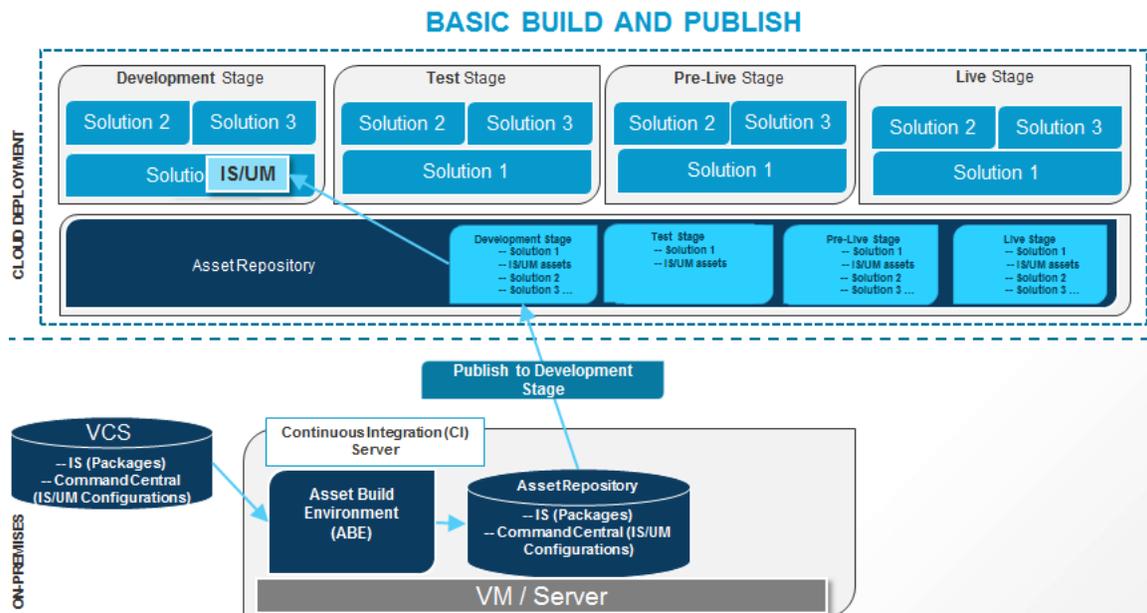
The following high-level steps describe the continuous integration (CI) and continuous deployment (CD) mechanism:

1. Create source packages and store them in a file system or any version control system (VCS). The source directory must be accessible to the build scripts. If the source directory is a file system on a VCS, ensure that the source directory is checked out in the local file system.

2. **Creating and exporting configurations from Software AG Command Central**

   To create Command Central configurations, use the following command: `sagcc exec templates composite generate -i <generate-templatemetadata-file>.xml`

   To export the configuration YAML files to a directory that you intend to use as sources for ABE, use the following command: `sagcc get templates composite export <templateAlias> -o <template-file>.yaml -f application/yaml`. Ensure that the exported YAML files are accessible to the build scripts.

3. Install webMethods Asset Build Environment (ABE). ABE installs the build scripts and the build properties file that you use to build the composites and descriptors.

See the *Building Composites for Repository-Based Deployment* section in the *webMethods Deployer User's Guide* for more information.

4. Set the properties for the build and run the build script in ABE to create the composites and descriptors. See the *Building Composites for Repository-Based Deployment* section in the *webMethods Deployer User's Guide* for more information.

5. Use ABE to push the composites and descriptors to the Integration Cloud Asset Repository. Apart from providing values to the properties in build.properties, specify the following *additional* properties to upload composites and descriptors to the Integration Cloud Asset Repository.

Property	Definition
localStore	Represents the local folder where the Git repository is cloned or the local Git repository. *localStore* must be the same as the *build.output.dir* .

remoteStore	URL of the Integration Cloud Asset Repository. The URL format is:
	
	`https://mysubdomain.webmethodscloud.com/`
	`integration/rest/internal/wmic-git/`
	`<stagename>-<solutionname>-<productaliasname>`
	
	Example:
	
	`https://mysubdomain.webmethodscloud.com/`
	`integration/rest/internal/wmic-git/`
	`development-mysolution-myis`

message	Required. Message which describes the commits.
username	Required. Integration Cloud login user name.
password	Required. Integration Cloud login password.

Run one of the following commands from the <Software AG_directory>\common \AssetBuildEnvironment\bin directory to build and upload the composites and descriptors to the Integration Cloud Asset Repository:

For this platform...	Run the following command...
Windows	`build.bat buildUploadAssets`
UNIX	`build.sh buildUploadAssets`

6. **Promoting assets to the next stage**

---

a. Using an automated tool like Jenkins, invoke the following API to promote the assets to the next stage:

Method: POST

URL: https://{subdomain}.webmethodscloud.com/integration/rest/external/ landscapes/ assets/promote/{solutionName}/{nodeName}

URI parameters

■ *solutionName* - Name of the solution in Integration Cloud

■ *nodeName* - webMethods Integration Server instance where the assets need to be deployed.

> **Note:** Use *basic* as the HTTP authorization scheme.

Request Body: The request body should contain the following:

■ **fromStageName**: In which stage the assets are available in Integration Cloud

■ **toStageName**: To which stage the assets are to be promoted in Integration Cloud

■ **commitMessage**: An optional commit message

■ **acdlComposite**: List of composites, assets, and their properties, which are available in the previous stage.

> **Note:** If you want to retrieve the request body, invoke the *getcomposites* API as provided in *Step b*. The *getcomposites* API will get the composites and asset details from the previous stage.

Sample Request Body

```
{
"integration": {
   "cicdBean": {
     "fromStageName": "Development",
     "toStageName": "Test",
     "commitMessage": "Promoting assets from Development stage to
     Test stage.",
     "acdlComposite": {
        "IS": [{
          "name": "mySqlAssets",
          "targetNamespace": "http://namespaces.softwareag.com/
          webMethods/IS",
          "assets": [],
          "properties": [{
             "name": "activatePkgOnInstall",
             "values": [
                       "true"
             ]
          },
          {
             "name": "archivePkgOnInstall",
             "values": [
                       "true"
             ]
          },
```

```
                {
                  "name": "compilePackage",
                  "values": [
                          "true"
                  ]
                }
            ]
        },
        {
            "name": "JDBCRegression_deployment",
            "targetNamespace": "http://namespaces.softwareag.com/
            webMethods/IS",
            "assets": [{
                    "name": "deploy.notifications.deleteN",
                    "properties": [{
                        "name": "art.deployment.state",
                        "values": [
                            "disable"
                        ]
                    },
                    {
                      "name": "notificationImmediate",
                      "values": [
                          "false"
                      ]
                    },
                    {
                      "name": "notificationInterval",
                      "values": [
                          "10"
                       ]
                    },
                    {
                      "name": "notificationOverlap",
                      "values": [
                          "false"
                      ]
                    }
                ]
            }],
            "properties": [{
                  "name": "activatePkgOnInstall",
                  "values": [
                          "true"
                  ]
            }]
        }
      ]
    }
  }
}
```

b. **getcomposites API**

getcomposites retrieves all the composites and its assets which have properties.

Method: GET

URL: https://{subdomain}.webmethodscloud.com/integration/rest/external/
landscapes/assets/{solutionName}/{nodeName}?stageName={stageName}

URI parameters

- *solutionName*: Name of the solution in Integration Cloud

- *nodeName*: webMethods Integration Server instance where the assets need to be deployed

Query parameters

- *stagename*: The stage from where the composites and assets have to be retrieved

Allowed stage names: development, test, live, preLive

Note: Use *basic* as the HTTP authorization scheme.

Sample Response
```
{
 "integration": {
  "message": {
    "code": 0,
    "description": "Success"
},
"cicdBean": {
    "fromStageName": "Development",
    "toStageName": "Test",
    "commitMessage": null,
    "acdlComposite": {
      "IS": [{
        "name": "mySqlAssets",
        "targetNamespace": "http://namespaces.softwareag.com/
        webMethods/IS",
        "assets": [],
        "properties": [{
           "name": "activatePkgOnInstall",
           "values": [
                    "true"
           ]
         },
         {
           "name": "archivePkgOnInstall",
           "values": [
                    "true"
           ]
         },
         {
           "name": "compilePackage",
           "values": [
                    "true"
           ]
         }
        ]
     },
     {
       "name": "JDBCRegression_deployment",
       "targetNamespace": "http://namespaces.softwareag.com/
       webMethods/IS",
       "assets": [{
                   "name": "deploy.notifications.deleteN",
                   "properties": [{
                       "name": "art.deployment.state",
```

```
                              "values": [
                                         "disable"
                              ]
                    },
                    {
                         "name": "notificationImmediate",
                         "values": [
                                    "false"
                         ]
                    },
                    {
                         "name": "notificationInterval",
                         "values": [
                                    "10"
                         ]
                    },
                    {
                         "name": "notificationOverlap",
                         "values": [
                                     "false"
                         ]
                    }
                   ]
          }],
          "properties": [{
                    "name": "activatePkgOnInstall",
                    "values": [
                               "true"
                    ]
          }]
        }
      ]
    }
  }
 }
}
```

The response contains the following:

- **fromStageName**: The stage where the assets are available

- **toStageName**: The stage where the assets can be promoted

- **acdlComposite**: The composites, assets, and their properties. You can replace the properties for the assets wherever required, and invoke the Promote Assets API as mentioned in *Step a* to promote the assets to the next stage.

Best Practices and General Information

This section contains information on best practices and general notes on Cloud Deployment.

Best Practices

Extended Settings deployment

■ If you use the **Deploy To Cloud** option in Software AG Designer, you cannot deploy *Extended Settings* parameters of on-premises webMethods Integration Server. For deploying extended settings, use the **Landscape Navigator** to generate *Templates*, which can then be deployed to webMethods Integration Cloud.

General Information

■ For specifying JDBC Jars that can be shared by multiple packages, on-premises users may have placed those files under the custom/jars directory in webMethods Integration Server. For cloud deployment, place those files in the code/jars/static directory of some common package to make the required JDBC classes available when the package is deployed to webMethods Integration Cloud.

■ If keystores and truststores are created in the customer's environment, location of the keystores and truststores may be pointing to some file on a local file system. It is recommended to place these files in the *config* directory of some common package. This package can be deployed to webMethods Integration Cloud and keystores and truststores can use the packages/CustomPackage/config/myKeystore.jks path to refer to the keystore location.

■ In case of failure to send messages from Software AG Universal Messaging, webMethods Integration Server stores the messages in Client side Queue (CSQ), which is currently an embedded database. As webMethods Integration Server is in a Docker container, any such locally stored messages may be lost.

■ Time out errors like *504 Gateway Time-out* may appear, if the processing time is greater than 2 minutes.

■ Software AG Designer allows you to deploy assets without the required dependencies as well as delete assets which are referenced by other assets. It is recommended to manually check and manage the dependencies.