How-To Library

Informatica

An approach to software development lifecycle operations for Cloud Application Integration assets

Abstract

This article outlines some of the frequent software development lifecycle (SDLC) operations that you might have to perform with Cloud Application Integration assets. It also describes the tools that you can use to perform SDLC operations. Based on your specific use case, you can use a tool that suits your requirements.

Supported Versions

- Informatica Cloud Application Integration July 2022

Table of Contents

Overview

When you work with Application Integration assets, you can use the following tools to perform SDLC operations:

IICS Asset Management CLI V2 Utility

Use the IICS Asset Management CLI V2 Utility with the IICS REST API to perform SDLC operations on IICS artifacts. The utility is a standalone, command line based utility, and offers options to easily list, export, extract, package, import, and publish assets.

Informatica Intelligent Cloud Services

Use the options available in Informatica Intelligent Cloud Services (IICS) to export, import, update, publish, and unpublish assets.

GitHub or Azure DevOps Git source control repository

Use a GitHub or Azure DevOps Git source control repository with IICS to manage and track changes made to IICS objects such as projects, folders, and assets.

Each tool has its own advantages and disadvantages. You might want to use one of these tools, or a combination of tools to achieve the desired results.

Comparative analysis of tools for SDLC operations

The following table provides a comparative analysis of the tools that you can use to perform SDLC operations:

Requirement	IICS Asset Management CLI V2 Utility	Informatica Intelligent Cloud Services	GitHub or Azure DevOps Git Source Control Repository
Copy of IICS assets between the Development and QA organization as part of bug fixing and testing. This activity can happen multiple times.	When you export an asset, sensitive fields such as passwords, access keys, and secret keys are not exported. You must either enter the values manually in the exported XML files and import the assets, or use the Update API to update the field values after importing the assets.	When you export an asset, sensitive fields such as passwords, access keys, and secret keys are not exported. You must either enter the values manually in the exported XML files and import the assets, or use the Update API to update the field values after importing the assets.	Values of sensitive fields are not lost after export. Therefore, this tool might be the most suitable for this scenario.
Copy of IICS assets to the Production organization as part of operationalizing	Since you will mostly perform an export and import only once when you push objects to a production environment, you can use this tool and use the Update API to update the sensitive field values after importing the assets.	Since you will mostly perform an export and import only once when you push objects to a production environment, you can use this tool and use the Update API to update the sensitive field values after importing the assets.	For security purposes, avoid using this tool for production updates.
Copy of assets from one organization to another	You can use the utility to perform a gamut of activities such as: - Listing assets for export - Exporting assets - Extracting assets into a local workspace directory and updating them - Packaging the updated assets	After you use the utility to perform the initial set of operations, you can then use IICS for: - Updating field values especially for sensitive fields with the Update API - Publishing assets in bulk	Only export and import operations can be performed with GitHub or Azure DevOps Git.
Dependency management	You must manually manage the dependencies.	Dependencies are maintained.	Dependencies are maintained.

Using the IICS Asset Management CLI V2 Utility

You can use the IICS Asset Management CLI V2 Utility to use the IICS REST API and perform SDLC operations on IICS artifacts. The utility is a standalone, command line based utility.

Access the following URL to download the IICS Asset Management CLI V2 utility:

https://github.com/InformaticaCloudApplicationIntegration/Tools/tree/master/IICS%20Asset%20Management%20CLI/v2

Download the utility based on the operating system that you use.

Perform the following steps to run the utility:

1. From the command prompt, navigate to the directory where you downloaded the utility.
2. Run the required command.

You can use the following commands with the IICS Asset Management CLI V2 utility:

- List
- Export
- Extract
- Package
- Import
- Import status
- Publish
- Publish status
- Version

Listing assets for export

Use the list command of the IICS Asset Management CLI V2 utility to list the assets that you want to export.

For more information about downloading and using the utility, see the following document:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Syntax

```
iics list [options]
```

Options

See the following community article for information about the list command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Exporting assets

Use the export command of the IICS Asset Management CLI V2 utility to export assets.

Syntax

```
iics export [options]
```

Options

See the following community article for information about the export command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Extracting assets into a local workspace directory

Use the extract command of the IICS Asset Management CLI V2 utility to extract the exported IICS assets ZIP file into a local workspace directory.

Syntax

iics extract [options]

Run this command to extract an IICS ZIP file into a local workspace directory. The command extracts the ZIP file contents into individual assets. On extraction, the utility creates one metadata file in the local workspace directory for each project, folder, or artifact. The metadata file resides alongside the project, folder, or artifact file and will have a period character (.) before the file name. For example, if you extract the file /Explore/MyProject/a.PROCESS.xml, a corresponding metadata file /Explore/MyProject/.a.PROCESS.xml is created. Note the period character before the file name.

Note: The metadata files might be hidden by default in some file systems because the file names start with a period character. The metadata files must be preserved along with the project, folder, or artifact file. For example, if you add the workspace contents to a version control system, you must also add the metadata files to the version control system. Similarly, when you check out the contents from a version control system, you must also check out the metadata files from the version control system.

Options

See the following community article for information about the extract command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Packaging assets for import

Use the package command of the IICS Asset Management CLI V2 utility to package artifacts from a local workspace folder into a ZIP file. You can then import the file into IICS.

Syntax

iics package [options]

Options

See the following community article for information about the package command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Importing assets

Use the import command of the IICS Asset Management CLI V2 utility to import artifacts into IICS.

Syntax

iics import [options]

Options

See the following community article for information about the import command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Checking the import status

Use the import status command of the IICS Asset Management CLI V2 utility to check the status of the import into IICS.

Syntax

```
iics import status [options]
```

Options

See the following community article for information about the import status command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Publishing assets

Use the publish command of the IICS Asset Management CLI V2 utility to publish connections, service connectors, guides, processes, and taskflows.

After the command runs successfully, the status of the asset changes to **Published** in IICS. You can view and copy the generated service URLs from IICS.

Syntax

```
iics publish [options]
```

Options

See the following community article for information about the publish command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Checking the publish status

Use the publish status command of the IICS Asset Management CLI V2 utility to get the status of a publish job.

Syntax

```
iics publish status [options]
```

Options

See the following community article for information about the publish status command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Viewing the application version

Use the version command of the IICS Asset Management CLI V2 utility to view the application version.

Syntax

```
iics version [options]
```

Options

See the following community article for information about the version command options:

https://knowledge.informatica.com/s/article/DOC-18245?language=en_US

Using Informatica Intelligent Cloud Services (IICS)

You can use Informatica Intelligent Cloud Services to export, import, update, and publish assets.

Exporting assets from IICS

When you export assets, Informatica Intelligent Cloud Services creates an export ZIP file that contains the assets that you selected for export.

You can select individual assets to export, or you can select an entire project or folder. When you export a project or folder, the export file includes all of the assets in the project or folder.

To export an asset, you need the following privileges and permissions:

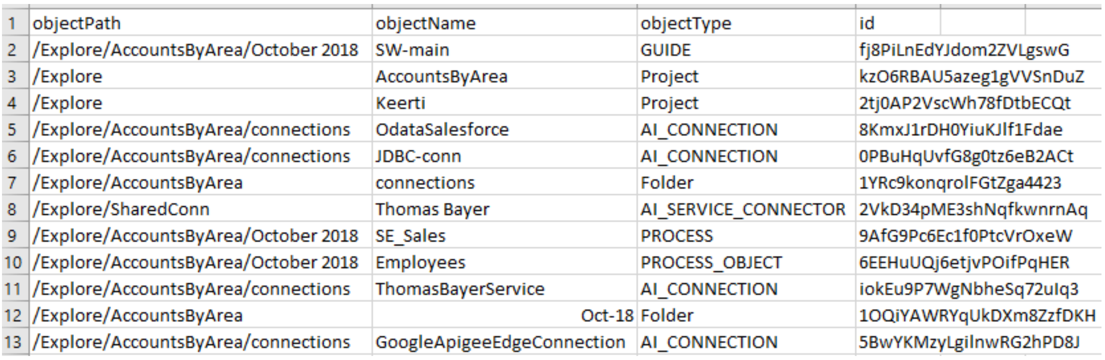
- Your user role must have privileges to export assets.
- You must have read permission on the asset.

Note: Informatica recommends that you include no more than 1000 objects in an export file.

Export files

The export file includes a spreadsheet that lists the objects within the file.

The following image is an example of an export spreadsheet:

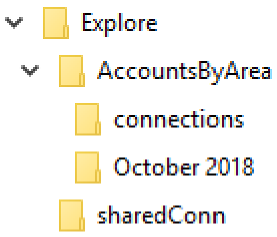


1	objectPath	objectName	objectType	id
2	/Explore/AccountsByArea/October 2018	SW-main	GUIDE	fj8PiLnEdYJdom2ZVLgswG
3	/Explore	AccountsByArea	Project	kzO6RBAU5azeg1gVVSnDuZ
4	/Explore	Keerti	Project	2tj0AP2VscWh78fDtbECQt
5	/Explore/AccountsByArea/connections	OdataSalesforce	AI_CONNECTION	8KmxJ1rDH0YiuKJlf1Fdae
6	/Explore/AccountsByArea/connections	JDBC-conn	AI_CONNECTION	0PBuHqUvfG8g0tz6eB2ACt
7	/Explore/AccountsByArea	connections	Folder	1YRc9konqrolFGtZga4423
8	/Explore/SharedConn	Thomas Bayer	AI_SERVICE_CONNECTOR	2VkD34pME3shNqfkwnrnAq
9	/Explore/AccountsByArea/October 2018	SE_Sales	PROCESS	9AfG9Pc6Ec1f0PtcVrOxeW
10	/Explore/AccountsByArea/October 2018	Employees	PROCESS_OBJECT	6EEHuUQj6etjvPOifPqHER
11	/Explore/AccountsByArea/connections	ThomasBayerService	AI_CONNECTION	iokEu9P7WgNbheSq72uIq3
12	/Explore/AccountsByArea	Oct-18	Folder	1OQiYAWRYqUkDXm8ZzfDKH
13	/Explore/AccountsByArea/connections	GoogleApigeeEdgeConnection	AI_CONNECTION	5BwYKMzyLgilnwRG2hPD8J

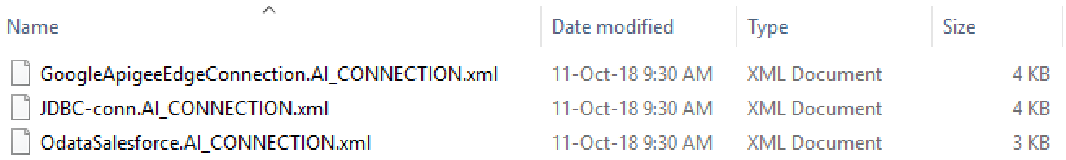
Export file structure

The export file retains the file structure of the source organization's **Explore** page for projects, folders, and assets. The Explore folder contains assets in the XML format. If you exported Data Integration assets, you see a SYS folder with connections and runtime environments.

The following image shows the contents of an export file:

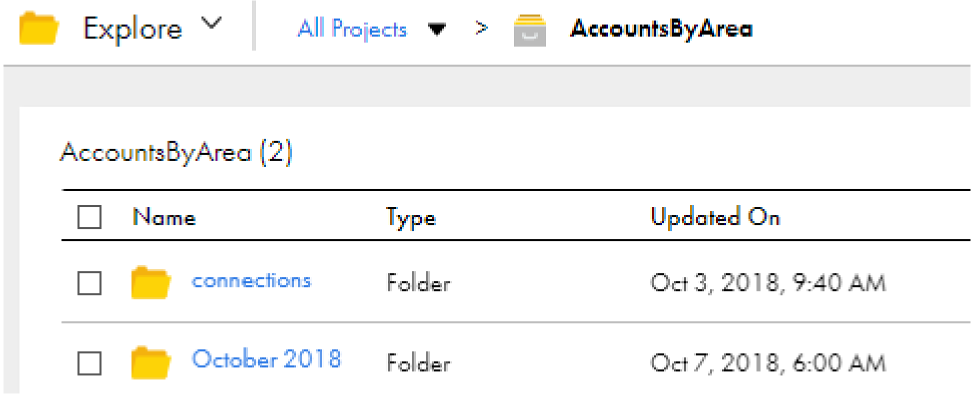


The following image shows the contents of the `Explore\AccountsByArea\connections` folder:



Name	Date modified	Type	Size
GoogleApigeeEdgeConnection.AI_CONNECTION.xml	11-Oct-18 9:30 AM	XML Document	4 KB
JDBC-conn.AI_CONNECTION.xml	11-Oct-18 9:30 AM	XML Document	4 KB
OdataSalesforce.AI_CONNECTION.xml	11-Oct-18 9:30 AM	XML Document	3 KB

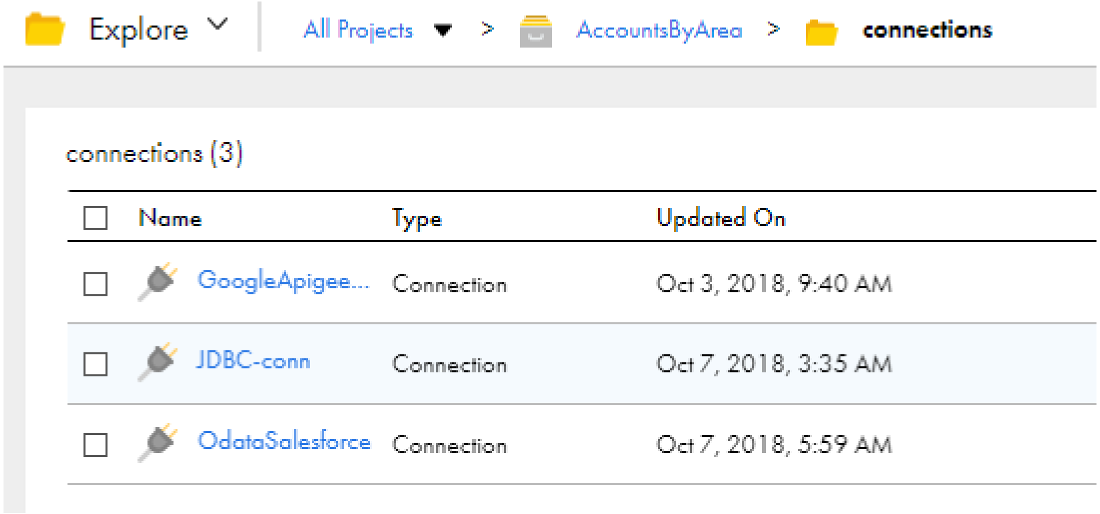
The following image shows the same folders in the AccountsByArea project of the source organization's **Explore** page:



AccountsByArea (2)

□	Name	Type	Updated On
□	connections	Folder	Oct 3, 2018, 9:40 AM
□	October 2018	Folder	Oct 7, 2018, 6:00 AM

The following image shows the assets contained within the connection folder of the source organization's **Explore** page:



connections (3)

□	Name	Type	Updated On
□	GoogleApigee...	Connection	Oct 3, 2018, 9:40 AM
□	JDBC-conn	Connection	Oct 7, 2018, 3:35 AM
□	OdataSalesforce	Connection	Oct 7, 2018, 5:59 AM

Object names

Each asset is contained in a ZIP file along with its associated metadata and JSON file. The ZIP file includes the asset name appended by the asset type.

For example, a mapping with the name of m_totals in the source organization has the name of m_totals.DTEMPLATE in the export file. When you export the m_totals mapping to the target organization, the asset has its original name of m_totals.

The following table lists the asset types and the associated extension appended to asset names:

Asset Type	Extension
Taskflow	TASKFLOW
Process	PROCESS
Guide	GUIDE
Connection	AI_CONNECTION
Service connector	AI_SERVICE_CONNECTOR
Process object	PROCESS_OBJECT

Exporting assets

You can select a single asset, multiple assets, or a project to export.

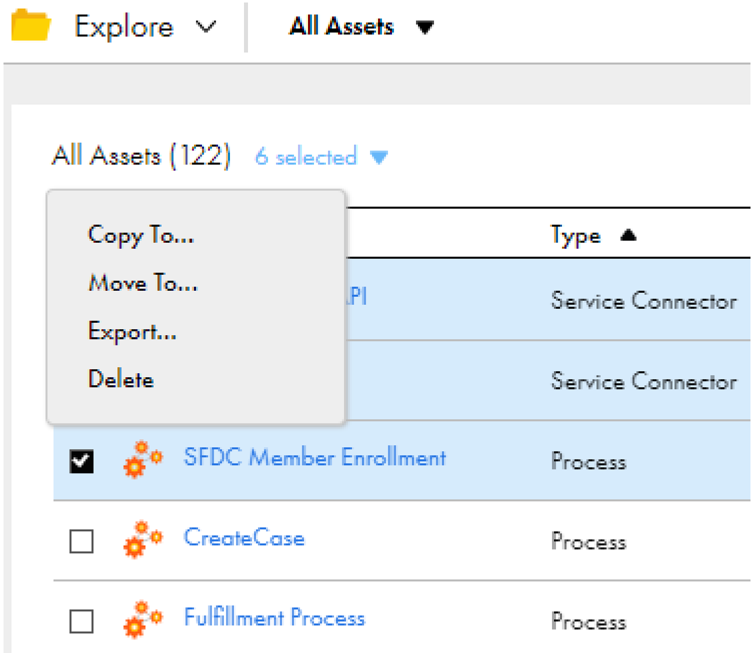
To include multiple assets, you can either select each asset within a folder or select a project or folder to export all of its assets. If you export a project, during import you can import the entire project or import only the assets that you select.

1. Log in to the source organization.
2. On the **Explore** page, navigate to the assets that you want to export.
3. Select the assets that you want to export.

   To export a single asset or project, select the asset or project, and then click **Actions** and select **Export**.

   To export multiple assets, select the check box to the left of each asset that you want to export. Or, select the check box for each project or folder that contains the assets that you want to export. From the selection menu, select **Export**.

The following image shows the selection menu with multiple assets selected:



4. On the **Export Assets** page, change the job name or leave the default.
5. Select whether to include dependent objects for the assets.
6. Click **Export**.
7. To see the progress of the job, select **My Import/Export Logs** from the navigation bar and then select **Export** from the menu at the top of the page. Click the name of the log to open the log details page.
8. To download the export file, click **Download** next to the export job on the **My Import/Export Logs** page.

Tip: You can quickly open the log details page for a completed export job in **Notifications**. When the export process is complete, a message appears in **Notifications**. Click the link in the message to open the log details page.

Importing assets into IICS

See the following document for information about importing assets into IICS:

https://docs.informatica.com/integration-cloud/cloud-application-integration/current-version/asset-management/asset-migration/asset-import.html

Using the update API to update field values

See the following community article for information about using the Update API to update field values:

https://knowledge.informatica.com/s/article/DOC-19383?language=en_US

Publishing assets in bulk

See the following link for information about publishing assets in bulk:

https://docs.informatica.com/integration-cloud/cloud-application-integration/current-version/invoke/publishing-application-integration-assets-in-bulk.html https://docs.informatica.com/integration-cloud/cloud-data-integration/current-version/taskflows/taskflows/publishing-taskflows-in-bulk.html

Note: You must publish Application Integration assets in this order to take care of the dependencies:

1. Service connectors
2. App connections
3. Processes and guides

Unpublishing and deleting dependent assets

See the following document for information about unpublishing and deleting dependent assets:

https://docs.informatica.com/integration-cloud/cloud-application-integration/current-version/asset-management/project-and-asset-management/asset-dependencies.html

Using GitHub or Azure DevOps Git source control repository

For information about using GitHub or Azure DevOps Git source control repository, see:

https://docs.informatica.com/integration-cloud/cloud-application-integration/current-version/asset-management/source-control.html

Use case scenarios

The following topics describe some common scenarios where you might perform SDLC operations and gives some tool recommendations.

Use case 1: first deployment

Consider the following scenario and best practices:

1. A Developer creates the first set of assets in the development environment. These include **Process proc1** that uses **Connection cn1**, which in turn uses **Service Connector svc1**. The Developer tags them as **v1**.
2. When the Developer needs to deliver the first drop to QA: Use the CLI utility and perform the following steps for source control:
    a. Uses **iics list -q "tag==v1" -o assetlist** to get the list of assets tagged with v1 and write them into the assetlist file.
    b. Use **iics export -f assetlist** to export all the assets.
    c. Use **iics extract** to extract all the assets to the Source Control Management (SCM) local workspace.
3. When QA needs to consume the first drop: Use the CLI utility and perform the following steps for deployment:
    a. Use **iics package -o assetlist** to create the import package.
    b. Use **iics import** to import the assets into the QA organization.
    c. Use **iics publish** to publish the assets in the organization. Ensure that you publish the dependencies first and then the dependents. This can be done by publishing the assets in the following order: Service Connectors, App Connections, Processes, and Guides.

Use case 2: simple patch deployment

Consider the following scenario and best practices:

1. QA finds a bug with the first deployment. The issue is with the Application Integration process.
2. The Developer updates the v1 Process in the development environment and assigns a new tag v2 for this process alone.
3. When the Developer needs to deliver the patched process to QA: Use the CLI utility and perform the following steps:
    a. Use **iics list -q "tag==v2" -o assetlist** to get the list of assets tagged with v2 and write them into the assetlist file.
    b. Use **iics export -f assetlist** to export all the assets.
    c. Use **iics extract** to extract all the assets to the SCM local workspace.
4. When QA needs to consume the patch: Use the CLI utility and perform the following steps for deployment:
    a. Use **iics package -o assetlist** to create the import package.
    b. Use **iics import** to import the assets into the QA organization.
    c. Use **iics publish** to publish the assets in the organization. Ensure that you publish the dependencies first and then the dependents. This can be done by publishing the assets in the following order: Service Connectors, App Connections, Processes, and Guides.

Use case 3: complex patch deployment with dependencies but no connection or service connector updates

Consider the following scenario and best practices:

1. QA finds a bug with the first deployment. The issue is with the Application Integration process.
2. The Developer updates the v1 Process in the development environment and assigns a new tag v2 for this process alone.
3. When the Developer needs to deliver the patched process to QA: Use the CLI utility and perform the following steps:
    a. Use **iics list -q "tag=v2" -o assetlist** to get the list of assets tagged with v2 and write them into the assetlist file.
    b. Use **iics export -f assetlist** to export all the assets.
    c. Use **iics extract** to extract all the assets to the SCM local workspace.
4. When QA needs to consume the patch: Use the CLI utility and perform the following steps for deployment:
    a. Use **iics package -o assetlist** to create the import package.
    b. Use **iics import** to import the assets into the QA organization.
    c. Use **iics publish** to publish the assets in the organization. Ensure that you publish the dependencies first and then the dependents. This can be done by publishing the assets in the following order: Service Connectors, App Connections, Processes, and Guides.

Use case 4: complex patch with changes to connection

In this case, use the CLI utility. Do not import the connection using IICS.

Authors

Sorabh Agarwal

Anu Chandrasekharan