NSP
Network Services Platform
Release 24.8

Network Automation Guide

3HE-20012-AAAB-TQZZA
Issue 1
August 2024

Legal notice

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

Contents

About this document

Purpose

The Network Automation Guide introduces NSP network automation functions to operators and administrators by describing usage and features.

Scope

Network automation functions are available for OSS using programmable APIs. For general information about developer support, see the API page on the Network Developer Portal.

Document support

Customer documentation and product support URLs:

- Documentation Center
- Technical support

How to comment

Please send your feedback to documentation.feedback@nokia.com.

Part I: NSP Network Automation

Overview

Purpose

Describes the contents of the NSP Network Automation Guide.

Contents

1 Network automation in NSP

1.1 Network Automation functions

1.1.1 Functions included

Network Automation in NSP allows you to plan, perform, and audit device configuration at a network level rather than at an individual device level. Automating processes allows you to perform updates on multiple devices at one time.

Automation reduces or eliminates the need for manual intervention, which increases the productivity of operations teams across a wide range of tasks. These include provisioning new devices, configuring network services, monitoring and optimizing network and service performance, and troubleshooting problems.

The following, individually and in combination, are designed to assist operators in automating a variety of network management functions:

- Intents
- Workflows
- SDK for adaptors

1.2 What does this guide cover?

1.2.1 Guide scope

This guide provides an overview of many network automation topics in NSP.

The following table provides information about where to find additional network automation information.

Automation feature	Description	Information scope	Reference
Device Configuration	With Device Configuration, you can define reusable configuration templates covering physical configurations such as ports and services, and logical configurations such as QoS. These templates can be deployed to the network with fixed or flexible attributes.	Overview for operators	*NSP Device Management Guide*
		Detailed information for developers, including information about working with APIs.	Device Configuration tutorial on the Network Developer Portal

Automation feature	Description	Information scope	Reference
Operations	An operation is a series of executions, organized in phases, which are performed on a scope of NEs. You can use an operation to perform executions on large numbers of NEs concurrently; for example, upgrading all SR NEs in a network to the latest SR OS release.	Overview for operators	*NSP Device Management Guide*
		Detailed information for developers, including information about working with APIs.	Device Management tutorials on the Network Developer Portal
Service Management	Service Management allows for service provisioning and activation across networks accessible to the NSP. Through the application itself, or through the northbound interface (RESTCONF), Service Fulfillment enables users to make service requests that deploy services to the network using the NSP's mediation framework. A library exists with a predefined set of service models for both classic and model-mode SR OS networks. These service models can be installed and utilized by NSP to provide assurance that service configuration is as planned/requested, and also provides adaptability for custom service model requests.	Overview for operators	*NSP Service Management Guide*
		Detailed information for developers, including information about working with APIs.	Service Management tutorial on the Network Developer Portal
Resource Management	Resource Management manages resource pools and monitors pool usage. It maintains a list of available resource pools in a network, and allows the user to create and modify pools. Resource Management provides a quick way to assign IP addresses, numbers, or text strings to ports when you are setting up automated processes.	Overview for operators	*NSP System Administrator Guide*
		Detailed RESTCONF NBI information for developers	Resource Administrator tutorial on the Network Developer Portal

Part II: Artifacts

Overview

Purpose

Describes the use of NSP to manage artifacts.

Contents

2 Artifact management

Artifact support

2.1 What is artifact support in NSP?

2.1.1 Artifacts overview

NSP may require artifacts, as defined in 2.6 “What is an artifact?” (p. 23), to be installed in order to perform specific functions. For example, the NE upgrade operation requires a series of workflows. The functional area within NSP that the artifact is designed to support is called the target application, or simply, target.

NSP facilitates the tracking and management of NSP artifacts and artifact bundles by providing one location and one method for managing artifacts and artifact bundles in your system.

Adaptor artifacts

NSP supports a variety of multi-vendor devices via pluggable artifacts and adaptors, which are grouped by vendor and type/use case and bundled together for delivery.

Some artifacts are generic, supporting multiple NE releases. Other artifacts are specifically targeted to a single NE release.

Certain artifacts can be installed from the **Artifacts**, **Artifact Bundles** view, as shown in Table 2-1, “Artifacts supported for installation using the NSP Artifacts views” (p. 16), while other artifacts must be installed using legacy tools.

Obtaining artifacts

Depending on your feature packages, a set of artifacts and bundles is included with your NSP installation. Additional or updated artifact bundles can be downloaded from the Nokia NSP software delivery site:

- Artifacts that can be installed using the Artifacts import functions in the NSP UI are located in the NSP/<release>/Artifacts hierarchy. See 2.3 “How do I install an artifact bundle?” (p. 20).
- Production adaptors and their associated artifacts can be obtained from the NSP/<release>/Adaptors folder. For installation of adaptors and other artifacts that cannot be installed using the Artifacts views, see “How do I install adaptor artifacts that are not supported in the Artifacts view?” in the *NSP System Administrator Guide*

  Artifact guides are provided with the adaptors for each NE family and NSP release. For example, the Nokia SR OS Artifact Guide for Release 23.11 lists and describes the adaptor suites delivered to support management of Nokia SR OS devices by NSP Release 23.11 over model-driven interfaces. The artifact guides also contain information about the NSP functionality supported by the adaptors, NE compatibility with those NSP functions, NE commissioning information and a view of active issues.

Artifacts can be delivered or updated outside the NSP release cycle.

Compatible artifacts

The following table shows the artifact types that can be imported via NSP's Artifacts views.

Table 2-1 Artifacts supported for installation using the NSP Artifacts views

Artifact type	Target application
Device mappings for MDM Server telemetry	MDM Telemetry Mappings
Resync device mappings for MDM managed NEs	MDM Resync Mappings
Resync device mappings for classically managed NEs	NFM-P Resync Mappings
YANG to YANG mappings for IETF	YANG-to-YANG Mappings
JSON ACT alarm rules mapping	NSP ACT Framework
Operation types	Device Management - Operations
Cloud native telemetry resources	Telemetry
Workflows, actions, and Jinja2 templates	workflow-manager These artifacts are found in the Workflows views after installation.
Intent types	intent-manager All intent types are found in the Network Intents views after installation. To use intent types in other views, such as Service Fulfillment, you need to import them from the view they will be used in.
Service Management metadata files for data sync	service-fulfillment
Model-driven adaptor artifacts[1]	MDM

Notes:

1. Some adaptor artifacts may not be supported for installation using the Artifacts views. See the artifact guide for the adaptor bundle to verify compatibility. If the adaptor bundle is not installable from the Artifacts view, see "How do I install adaptor artifacts that are not supported in the Artifacts view?" in the *NSP System Administrator Guide* for information about installing it.

2.1.2 Developer mode

If developer mode is disabled in the NSP, user actions are limited. Importing artifacts using the Artifacts views is supported for signed artifacts only.

See the *NSP Installation and Upgrade Guide* for more information.

2.1.3 Access control

Depending on your assigned role, some operations in the Artifacts views may not be available to you. Contact your system administrator if you can't access certain network resources or information.

Note: Disabling developer mode overrides access control settings. If developer mode is disabled, restricted functions are restricted for all users.

Artifact bundles

2.2 What is an artifact bundle?

2.2.1 Artifact bundle overview

An artifact bundle is a collection of artifacts that may make up a specific use case, for example, an NE backup, which may involve few artifacts, or a complex service rollout use case that involves many artifacts.

Artifacts are packaged for download as bundles, in zip format. When you import an artifact bundle from local storage to Artifacts and install it, the artifacts are installed by the dedicated deployers for the artifact type; for example, the Network Intents deployer installs intent type artifacts. After installation, the artifacts appear in the **Artifacts**, **All Artifacts** view.

Artifacts that were not installed from the Artifact Bundles view, such as adaptors, are not displayed in the All Artifacts view. You can list the installed adaptors using the **`./adaptor-suite.bash --list`** command; see “How do I install adaptor artifacts that are not supported in the Artifacts view?” in the *NSP System Administrator Guide*.

2.2.2 Artifact bundle parameters

The following table describes the parameters that appear on the **Artifacts**, **Artifact Bundles** view.

Table 2-2 Artifact bundle parameters

Parameter	Predefined values	Notes
Bundle Name	—	If a bundle has a verified signature, the certificate is displayed, followed by the bundle’s name.
Author	—	If a bundle has a verified signature, the author is displayed.
Version	—	Version number of the bundle. Artifacts in the bundle may have different version numbers.
Number of Artifacts	—	The number of artifacts in the bundle. This parameter is displayed when the bundle is installed.

Table 2-2 Artifact bundle parameters (continued)

Parameter	Predefined values	Notes
Status	Importing	The bundle is being imported.
	Imported	The bundle has been imported to NSP.
	Installing	At least one artifact in the artifact bundle is in an interim state: installation has not succeeded or failed yet.
	Installed	Installation has completed successfully for all artifacts in the bundle.
	Partially Installed	Installation of one or more artifacts in the bundle has succeeded, and installation of one or more artifacts has failed.
	Installation Failed	Installation has failed for all artifacts in the bundle.
	Verification Failed	The bundle signature is not valid.
	Creation failed	One or more artifacts of the bundle have incompatible names. or there are syntax issues with the metadata.json file
	Uninstalling	The artifacts in the bundle are being uninstalled.
	Uninstalled	Uninstallation has completed successfully for all artifacts in the bundle.
	Partially Uninstalled	Uninstallation of one or more artifacts in the bundle has succeeded, and uninstallation of one or more artifacts has failed.
	Uninstallation Failed	Uninstallation has failed for all artifacts in the bundle.
	Waiting	The bundle is imported but the target has not yet responded to the installation request.
	Skipped	No target has been detected to install the artifacts in the bundle.
	Obsolete	All the artifacts in the bundle have become obsolete.
Import Time	—	—

2.2.3 Deleting an artifact bundle

You can delete a bundle from the Table row actions menu; see “How do I navigate a list?” in the *NSP User Guide*. To be deleted, the value of the Status parameter must be Imported, Verification Failed, or Creation Failed.

2.3 How do I install an artifact bundle?

2.3.1 Purpose

Use this procedure to install a bundle of artifact using the NSP Artifact Bundles view. The artifact bundle must be compatible with the Artifacts views; see Table 2-1, “Artifacts supported for installation using the NSP Artifacts views” (p. 16). For installation of adaptors for model-driven NE management, see “How do I install adaptor artifacts that are not supported in the Artifacts view?” in the *NSP System Administrator Guide*.

2.3.2 Before you begin

Before you begin, the artifact bundle must be downloaded from the Nokia NSP software download site and stored to your local system in .zip file format; see “Obtaining artifacts” (p. 15).

Importing an artifact bundle transfers it from your local system to the NSP. When the bundle is installed, the contents of the zip file are extracted and made available to their targets.

You can install a bundle at the time you import it, or later.

The installation process starts immediately when you click **INSTALL**. As the installation process completes for each artifact in the artifact bundle, the artifacts are added to the **Artifacts**, **All Artifacts** view. When the automatic reconcile operation occurs next, NSP checks the status of all artifacts, updates the artifact bundle status if possible, and retries any failed installations.

Installing new versions of artifacts

If you install a bundle containing a new version of an artifact that is already installed in Artifacts, the new version is handled according to the version control policy of the target. If the old version of the artifact remains in Artifacts, its status is changed to Obsolete and the status of the bundle is changed to Partially Installed.

If you are installing a bundle whose contents may supersede artifacts you are using, such as an NSP service pack, ensure that you have performed any required backups; see 2.7.2 “Version management best practices” (p. 25).

2.3.3 Steps

1

Open **Artifacts**, **Artifact Bundles**.

2

Click **IMPORT & INSTALL**.

3

In the form that opens, drag and drop up to 10 zip files, or click **Browse** and navigate to the files on your system.

4

To install the artifact bundle immediately, click **IMPORT & INSTALL**. To import without installing, click **IMPORT**.

The chosen operation is triggered immediately. The artifact bundle status is updated to Imported or Installed when NSP has confirmed the status of all artifacts in the artifact bundle.

5

To install a bundle in Imported status, choose **Install bundle** from the ⋮ (Table row actions) menu.

END OF STEPS

2.4 How do I view artifact bundle contents?

2.4.1 To open a list of artifacts in an artifact bundle

The artifact bundle must in Installed status to view the list of artifacts.

In the **Artifacts**, **Artifact Bundles** page, click on an installed bundle and choose **View artifacts** from the ⋮ (Table row actions) menu. A filtered page opens showing the list of artifacts in the bundle.

Click **Artifacts** to return to the **Artifacts**, **Artifact Bundles** view.

2.5 How do I uninstall an artifact bundle?

2.5.1 Before you begin

The uninstallation process starts immediately when you click **Uninstall bundle**. As the process completes for each artifact in the artifact bundle, the status of the artifacts are changed on the **Artifacts**, **All Artifacts** view. When the automatic reconcile operation occurs next, NSP checks the status of all artifacts, updates the artifact bundle status if possible, and retries any failed uninstallations.

If an artifact appears in more than one bundle, uninstallation of any of the bundles will not affect the other bundles. The artifact remains in the list; the Artifact Summary shows the remaining bundles the artifact is in.

If you uninstall a new version of an artifact that was previously installed using Artifacts, Artifacts automatically reinstalls the old version, according to the version control policy of the target.

2.5.2 Steps

1

Open **Artifacts**, **Artifact Bundles**.

2

Click on a bundle and choose **Uninstall bundle** from the ⋮ (Table row actions) menu.

3

In the confirmation form, click **UNINSTALL**.

The statuses of the bundle and its artifacts are updated to Uninstalled when the process succeeds, then revert to Imported.

END OF STEPS

Artifacts

2.6 What is an artifact?

2.6.1 Artifacts support NSP functions

An artifact, in the context of an NSP function, is one of a range of objects that evolves with the NSP product. The following are examples of NSP artifacts:

- workflows, actions, and Jinja templates
- configuration, service, or ZTP intent types
- operation types
- attribute dependent alarm rules for Model Driven (SROS/SRL) devices
- adaptors for model-driven NE management

2.6.2 Artifact dependency

An artifact may require the presence of other artifacts, such as adaptors, to work correctly. For example, an intent type may require NE adaptors to be available before it can be used to create a service. Network Intents can install the artifact, but the artifact is dependent on the NE adaptor to work. The dependency status parameter indicates whether dependencies exist and whether they are resolved, that is, whether the dependent artifact is present.

2.6.3 Artifact parameters

The following table describes the parameters that appear on the **Artifacts**, **All Artifacts** view.

Table 2-3 Artifact parameters

Parameter	Predefined values	Notes
Artifact Name	—	If an artifact has a verified signature, the certificate is displayed, followed by the artifact's name.
Author	—	If an artifact has a verified signature, the author is displayed.
Version	—	Version number of the artifact.
Dependency Status	Pending	NSP is checking for dependencies.
	No Dependencies Exist	—
	Resolved	All dependent artifacts are installed.
	Unresolved	At least one dependent artifact is not installed.

Table 2-3 Artifact parameters (continued)

Parameter	Predefined values	Notes
Retry	true	The artifact will be included in the next automatic reconcile operation. See the Artifact Information panel for the interval before the next reconcile.
	false	The artifact will not be included in the next automatic reconcile. There may be a problem with the artifact, or with the target.
Target	—	The functional code, or application, that installs the artifact.
Status	Installing	The artifact is being installed in NSP.
	Installed	Installation has completed successfully.
	Installation Failed	Installation has failed.
	Uninstalling	The artifact is being uninstalled.
	Uninstalled	Uninstallation has completed successfully.
	Uninstallation Failed	Uninstallation has failed.
	Verification Failed	The artifact digest is not valid
	Creation Failed	The artifact has an incompatible name, or there are syntax issues with the metadata.json file.
	Waiting	The target has not yet responded to the installation request.
	Obsolete	The artifact has been replaced by a newer version.
	Skipped	No target has been detected to install the artifact.
Import Time	—	—

2.7 Version management

2.7.1 Artifact version management

NSP supports version control for artifacts. Artifact versions are managed by the target deployer responsible for the artifact type.

For example, the unique identifier for a workflow is the workflow name. If multiple versions of the same workflow are available, such as 1.1.0 and 2.2.0, the deployer only installs the latest version, version 2.2.0.

In Network Intents, the unique identifier is intent type and major version, therefore if the artifact bundle contains version 1.1.0 and version 2.2.0 of the same intent type, the deployer installs both.

The version control policy of the target deployer also determines how new versions are handled when an existing version of an artifact is present. In many cases, the status of the lower version artifact is updated to Obsolete. The status of the higher version artifact is Installed.

2.7.2 Version management best practices

When you install an artifact using Artifacts, the artifacts in the bundle are installed in the targets according to the version control policy of the target. This means that new artifacts may supersede existing artifacts in use, regardless of how the existing artifact was installed.

If you are installing a bundle that may contain new versions of artifacts you are using, such as an NSP service pack, the following steps are recommended:

- Check the list of artifacts in the bundle.

  You can do this by opening the zip file in a zip file utility, or checking documentation that accompanies the bundle.
- For each affected function, verify whether backups are needed.

  For example, if the bundle includes new service intent types, export intents for the intent types you are currently using in Service Management.

2.8 What is automatic reconcile of artifacts?

2.8.1 Automatic reconcile

The automatic reconcile operation retries failed installation and uninstallation operations. The reconcile operation sends eligible artifacts to the deployers for their targets to run the operations again. To be eligible for a reconcile, an artifact's status must be Installation Failed or Uninstallation Failed, and the retry parameter must be true.

By default, Artifacts performs a reconcile operation every three minutes.

2.9 How do I retry a failed artifact operation?

2.9.1 Retry parameter

Artifacts retries failed installation and uninstallation operations automatically; see 2.8 "What is automatic reconcile of artifacts?" (p. 25).

- To verify that a failed artifact is eligible for the next reconcile operation, verify that the retry parameter is true.
- To change the retry parameter, choose an artifact and click **Enable retry** or **Disable retry**.
- To check when the next automatic reconcile occurs, choose a failed artifact and view the Artifact Summary panel.

2.10 How do I edit or delete an artifact?

2.10.1 Modifying artifacts

Artifacts cannot be edited in the **Artifacts**, **All Artifacts** view. Modification, if possible, can be done in the target or using a developer tool.

Signed artifacts cannot be modified in NSP. Contact Nokia to request changes to a Nokia-signed artifact.

2.10.2 Deleting artifacts

An artifact is uninstalled when a bundle that contains it is uninstalled; see 2.5 “How do I uninstall an artifact bundle?” (p. 21).

Artifacts cannot be deleted individually.

See “How do I uninstall MDM adaptor suites?” in the *NSP System Administrator Guide* for information about removing artifacts that were installed using the **`adaptor-suite.bash`** script.

2.11 How do I roll back an artifact version?

2.11.1 Artifact version overwrites

When you install an artifact bundle, artifacts in the bundle are installed according to the version management policy of the target; see 2.7 “Version management” (p. 24). In most cases, a new version of an existing artifact is installed, the older version remains in Artifacts, in Obsolete status.

To revert to the previous version, uninstall the new bundle. The older version of the artifact is reinstalled according to the version control policy of the target.

Part III: SDK Application

Overview

Purpose

Describes the SDK application for operators.

Contents

3 SDK

3.1 What is the SDK?

3.1.1 SDK overview

The software development kit (SDK) is a collection of software development tools in one installable package. They facilitate the creation of applications by having a compiler, debugger and sometimes a software framework. They are normally specific to a hardware platform and operating system combination.

The SDK can take the form of application programming interfaces (APIs) in the form of on-device libraries of reusable functions used to interface to a particular programming language, or it may be as complex as hardware-specific tools that can communicate with a particular embedded system.

The SDK facilitates adaptor development in the MDM environment.

3.1.2 Prerequisites

The following prerequisites must be met before installing and using the SDK.

- The RHEL 7 or RHEL 8 operating system must be installed on the host. The *NSP Installation and Upgrade Guide* provides detailed instructions for the RHEL OS installation.
- Docker-CE version 18.x with compatible Docker Compose (for RHEL 7) or version 24.x with compatible Docker Compose (for RHEL 8) are required for the SDK.
- To start the installation, log in as the root user on the SDK host.
- The deployer host must have following utilities: **`Openssl`**, **`Curl`**, and **`Wget`**.

Server size requirements

The deployer host must have at least 32 GB of memory and a minimum of 500 GB disk space available to run the SDK.

3.1.3 Port information

The following table describes the list of ports available on the SDK.

Table 3-1 Port information

Default Port	Type	Encryption Info	Description
80	TCP	There is no encryption.	This port redirects to port 443.
3002	TCP	Encryption provided by TLS. Strong ciphers are supported.	This port provides an HTTPS interface to the web application.
443	TCP	Encryption provided by TLS. Strong ciphers are supported.	This port provides an HTTPS interface to the web application.

Table 3-1 Port information (continued)

Default Port	Type	Encryption Info	Description
22	TCP	Dynamic encryption is used. The cipher suite and strength as per RFC 4253.	This port provides remote access through the SSH/SCP/SFTP protocols.
8101	TCP	Dynamic encryption is used. The cipher suite and strength as per RFC 4253.	This port provides remote access through the SSH/SCP/SFTP protocols.
8181	TCP	There is no encryption.	This is a Web console port.
5005	TCP	There is no encryption.	This port is for Java debugging.
162	UDP	When SNMPv3 is configured, static encryption is used. The cipher suite and strength are dependent on the network element.	This server port receives SNMP traps from network elements by default.

3.2 How do I install the SDK?

3.2.1 Before you begin

If the SDK is being installed on top of an existing SDK setup (upgrade scenario), use the command **`./nsp-mdm-sdk.sh stop`** to stop the SDK application and continue with the installation.

3.2.2 Steps

1

Download the MDM server and SDK application installer files from the NSP downloads page on the Nokia Support portal.

2

Open a console window and navigate to the folder where the installer files are located. Depending on your OS, you may need to change the permissions for each of the installer files.

For example, **`chmod +x mdmsdk-*.sh`**

3

Place the valid NSP license file in a folder containing the installer files before starting the installation.

Without a valid license file, the following prompt is displayed during the installation.

```
Please place the valid NSPLicense.zip in the current directory for SDK
installation and proceed.
```

4

Run the downloaded MDM server installer file.

For example, **`./mdmsdk-mdm-server-<version>.sh`**

5

Run the downloaded SDK application installer file.

For example, **./mdmsdk-installer-<*version*>.sh**

If the admin user is not already present, the following prompt is displayed.

```
Input your desired user or Press Enter for the Default user [admin] to
be created:
```

When you enter a custom user name, the following prompt is displayed.

```
Input your desired user or Press Enter for the Default user [admin] to
be created: <user_name>
Changing password for user <user_name>.
New password:
```

6

Enter the password.

The following prompt is displayed.

```
Retype new password:
```

7

Re-enter the password. The following prompt is displayed and SDK installation begins.

```
passwd: all authentication tokens updated successfully.
```

Note: Python and Git are installed in the process. If the Python or Git download fails, see 3.9 “Web Proxy” (p. 37).

A prompt is displayed to specify the <*sdk-installation-directory*> where the SDK should be installed.

```
Input your desired installation directory
[Press Enter for default: /root/nsp-mdm-sdk]:
```

Note: When requested for an installation directory during upgrade scenario, enter the same value as in the previous installation directory.

8

Perform one of the following

- Specify a new location for the installation directory.
- Specify an existing location for the installation directory. If the <*sdk-installation-directory*> already exists on the system, ensure that it has the necessary permissions to allow the SDK installer to write to it, otherwise, the installation will fail.

9

When the installation is finished, close the console window.

Note: The ndk user and group are created as part of the installation process. User authentication and ownership of the SDK filesystem will be assigned to this user.

END OF STEPS

3.3 How do I backup the SDK?

3.3.1 Steps

1

Navigate to the <*sdk-installation-directory*> directory and execute the command:

`./nsp-mdm-sdk.sh backup`

The following prompt is displayed.

```
Backing up to the 'sdk-backup' folder.
Backup created successfully in folder: sdk-backup with filename:
backup_<timestamp>.tar.gz
```

Note: Back up file format: *backup_YYYY-MM-DD_HH*h-*mm*m-*ss*s.*tar.gz*

The SDK backup creates a backup of all repositories and other settings.

END OF STEPS

3.4 How do I restore the SDK?

3.4.1 Before you begin

Before you begin the restore process, place the valid backup file in the ‘sdk-backup’ folder in the <*sdk-installation-directory*> directory.

Note: If there is an issue with upgrading the current system OS (for example, from RHEL 7 to RHEL 8), it is recommended to create a fresh installation in a new system and then copy the backup file to the "sdk-backup" folder in the <*sdk-installation-directory*> directory of the new system and begin the restore process.

3.4.2 Steps

1

Navigate to the <*sdk-installation-directory*> directory and execute the command:

`./nsp-mdm-sdk.sh restore`

The following prompt is displayed, listing the available backup archives.

```
Enter the number of the backup archive to restore (or 'q' to quit):
```

Note: If no backup files exist, the following prompt is displayed. `No backup archives found in the backup archive folder: sdk-backup.`

2

Enter the number that corresponds to the backup archive you want to restore and press Enter.

The following prompt is displayed.

```
Restoring backup: backup_<timestamp>.tar.gz ...
Backup 'backup_<timestamp>.tar.gz' restored successfully.
```

Note: If a valid number is not entered, the following prompt is displayed. `Please try again.`

Restore replaces the other settings and repositories with the same names as in the backup.

END OF STEPS

3.5 Working with the SDK

3.5.1 Command line arguments

The **nsp-mdm-sdk.sh** file in the <*sdk-installation-directory*> directory is used to start, stop, or view the logs of the running SDK application.

Supported command line arguments are:

```
./nsp-mdm-sdk.sh start
./nsp-mdm-sdk.sh stop
./nsp-mdm-sdk.sh logs
./nsp-mdm-sdk.sh mdm_console
./nsp-mdm-sdk.sh backup
./nsp-mdm-sdk.sh restore
```

Once the SDK application has started, the running MDM server version can also be changed. To do this, run the following command and select a different MDM server version.

```
./nsp-mdm-sdk.sh start
```

3.5.2 Documentation

The following documents are available from the Help menu in the SDK application:

- user guide – documentation of the SDK application, modeling guides for the NSP FM AMI and the NSP Equipment AMI, and detailed instructions for using the SDK Adaptor Designer Application to create and test adaptor suites
- troubleshooting guide – detailed description of the development environment (Gradle build and the MDM SDK Gradle plugins), troubleshooting information, and the online Javadoc API

3.6 How do I start the SDK application?

3.6.1 Before you begin

To successfully launch the SDK application, the necessary key and certificate for establishing an HTTPS connection must be available under `<sdk-installation-directory>/docker/nginx/ssl/` directory. See 3.7 “How do I generate a key and a certificate?” (p. 35).

Existing device modules must be reinstalled in the event of a restart.

3.6.2 Steps

1

Navigate to the <*sdk-installation-directory*> directory and execute the command:

`./nsp-mdm-sdk.sh start`

The following prompt is displayed requesting credentials for Git.

```
Input your desired user for GIT configuration or Press Enter for the
Default user [admin] to be configured:
```

```
Input your desired email for GIT configuration or Press Enter for the
Default email [admin@company.com] to be configured:
```

2

Gradle will be downloaded and installed.

Note: If there is issue in downloading gradle, you must manually place `gradle-4.8.1` and `gradle-5.6.4` under <*user-home-directory*> or try setting up a proxy. For more information on setting up the proxy, see 3.9 “Web Proxy” (p. 37), then return to Step 1 of this procedure.

3

The following prompt is displayed, listing the available MDM servers.

```
Please select one of the following MDM Server versions:
```

Enter the number that corresponds to the MDM server version you want to use for the SDK and press Enter.

4

The following prompt is displayed requesting to select build environment.

```
Please select one of the following build environments: (the generated
project build files will work in the selected build environment)
```

Enter the number that corresponds to the build environment you want to use for the SDK and press Enter.

END OF STEPS

3.7 How do I generate a key and a certificate?

3.7.1 Steps

1

Log in as the root user on the NSP, NFM-P VM, or alternative station, as required.

2

Open a console window.

3

Generate a private key. Enter the following command:

 # openssl genrsa -out key.pem *size_of_private_key*

where

size_of_private_key is the size of the private key file. You have a choice of five sizes: 512, 758, 1024, 1536, or 2048 (these numbers represent bits). Larger sizes provide greater security, but at the expense of CPU performance. The recommended size is 1024.

4

Generate a certificate signing request (CSR).

1. Enter the following:

   # openssl req -new -key key.pem -out csr.pem

   The following prompt is displayed:

   ```
   You are about to be asked to enter information that will be
   incorporated into your certificate request.
   What you are about to enter is what is called a Distinguished Name
   or a DN.
   There are quite a few fields but you can leave some blank.
   For some fields there will be a default value, If you enter '.',
   the field will be left blank.
   Country Name (2 letter code) [XX]:
   ```

2. Enter the country name. The following prompt is displayed:

   ```
   State or Province Name (full name) []:
   ```

3. Enter the state or province name. The following prompt is displayed:

   ```
   Locality Name (eg, city) [Default City]:
   ```

4. Enter the locality name. The following prompt is displayed:

   ```
   Organization Name (eg, company) [Default Company Ltd]:
   ```

5. Enter the organization name. The following prompt is displayed:

   ```
   Organizational Unit Name (eg, section) []:
   ```

6. Enter the organizational unit name. The following prompt is displayed:

   ```
   Common Name (eg, your name or your server's hostname) []:
   ```

7. Enter the name identifier. The following prompt is displayed:

   ```
   Email Address []:
   ```

8. Enter the E-mail. The following prompt is displayed:

   ```
   Please enter the following 'extra' attributes to be sent with your
   certificate request
   ```

   ```
   A challenge password []:
   ```

9. Enter the password. The folowing prompt is displayed:

   ```
   An optional company name []:
   ```

10. Enter the company name. The utility generates a CSR file.

5

Generate a certificate. Enter the following command:

openssl x509 -req -days *no_of_days* -in csr.pem -signkey key.pem -out cert.pem

where

no_of_days is the number of days for which the certificate is to be valid.

6

Delete certificate signing request.

1. Enter the following:

   # rm csr.pem

   The following prompt is displayed:

   ```
   rm: remove regular file ‘csr.pem’?
   ```

2. Enter the response: **Y**.

7

Close the console window.

END OF STEPS

3.8 Code-Server

3.8.1 What is Code-Server?

Code-Server is a Web-based IDE that simplifies project development. The SDK can run independently of the code-server, however the code-server requires the SDK to run.

3.8.2 How do I install the Code-Server?

Before beginning the Code-Server installation, ensure that the SDK is already installed.

1. Download the Code-Server installer from the NSP downloads page on the Nokia Support portal.
2. Open a console window and navigate to the folder where the Code-Server installer file is located. Depending on your OS, you may need to change the permissions for the installer file.

   For example, **chmod +x mdmsdk-code-server-<*version*>.sh**
3. Run the downloaded Code-Server installer file.

   For example, **./mdmsdk-code-server-<*version*>.sh**

3.8.3 How do I start the Code-Sever?

Navigate to the <*sdk-installation-directory*> directory and execute the command:

```
./code-server.sh start
```

The following prompt is displayed, if proxy is not set:

```
Web Proxy is not configured.

You can refer the SDK Installation guide for setting up Web proxy. Press
enter to proceed, (press q to exit):
```

After pressing Enter, set up the proxy. This is a one-time activity; once set, you won't be prompted to set it up again on future Code-Server startups. For more information on setting up the proxy, see 3.9 “Web Proxy” (p. 37).

After setting up the proxy, Code-Server starts up.

Execute the following command to see the Code-Server status:

```
./code-server.sh logs
```

3.8.4 Working with the Code-Server

Refer to the SDK application's User guide and Troubleshooting guide for more information.

3.8.5 How do I stop the Code-Sever?

Navigate to the <*sdk-installation-directory*> directory and execute the command:

```
./code-server.sh stop
```

3.9 Web Proxy

3.9.1 Why is Proxy needed?

If the server is in a private network and no proxy is configured, some features, such as installing Python and Git during SDK installation and installing new extensions through the Code-Server GUI, may not work.

3.9.2 How to set or unset Proxy?

Set Proxy:

export http_proxy=<*proxy*>

export https_proxy=<*proxy*>

Unset Proxy:

unset http_proxy

unset https_proxy

Part IV: Network Intents

Overview

Purpose

Describes the management of intent types and intents for operators.

Contents

4 Intent types

Overview

4.1 What are intent types?

4.1.1 Intent types

NSP allows you to create and execute intent-based automation flows to allow you to implement network-level planning and design. Intents translate high-level goals to the necessary network configuration, and the NSP generates and validates the configuration and continually verifies the state of the network.

With the use of intents, you can implement planning and design at a network level rather than at an individual device level. NSP translates the high-level goal from an intent to necessary network configuration. NSP generates and validates the configuration and continually verifies the state of the network and systems under its administrative control.

For example, you can have a goal to set up a service for a subscriber. In NSP, you define this goal using an intent type. The intent type defines the parameters for the service and provides mapping for device models. The intent type defines how an intent will work. When you create an intent using an intent type, you create an instance of the intent type with any required inputs provided. Network Intents maintains a catalogue of intent types created or imported by the user.

You can store service topology and configuration information in the database for later deployment and schedule audits of the network against the saved configuration requirements. Intents are customizable and can be deployed during runtime.

RESTCONF APIs are available for Network Intents; see the Intent Manager API page on the Network Developer Portal.

4.1.2 Intent types and intents

An intent type is a detailed specification for a desired network configuration. The intent type includes the YANG model and scripts and templates for the configuration to be performed. The intent type describes both what the creation of the intent looks like in the NSP, and what executing it does.

An intent is an instance of an intent type. The intent provides inputs to the intent type and executes the configuration.

Intent types can be modified or reused as often as needed, and can be deployed at any time for maximum flexibility.

4.1.3 Developer information

The Network Automation Guide provides an overview of network intents for operators.

For detailed information about the following topics for developers, see the Network Intents tutorial on the Network Developer Portal:

- intent type design

- resource file configuration
- schema form configuration

4.1.4 Developer mode

If developer mode is disabled in the NSP, users cannot create, import, modify, or delete intent types.

See the *NSP Installation and Upgrade Guide* for more information.

4.1.5 Access control

User groups are assigned access to menus, network resources, and Analytics resources by assigning roles in the NSP. Action permissions are assigned to roles.

Note: Disabling developer mode overrides access control settings. If developer mode is disabled, restricted functions are restricted for all users.

The following table describes scopes that are specific to intents.

Scope	Available operations
Manage Intent Types	Manage life cycle and delete intent types
Manage Intents	Create, edit, and delete intents, approve misalignments
Manage mediators	Perform all operations on the Mediators view
Import	Import all available file types
Operate-Intents	Synchronize, audit, mark as misaligned, run RPC actions, activate, migrate
Manage-Policies	All operations on the Policies and Policy Actions pages
Write Intent Types	Create and edit intent types

The scopes are combined to create the following roles:

- Developer:
  - Write Intent Types
  - Import
- Troubleshooter:
  - Manage Policies
  - Manage Mediators
  - Operate Intents
- Operator:
  - Manage Intents
  - Operate Intents
- Network Engineer
  - All scopes

Depending on your access settings, some of these pages or operations may not be available to you. See the *NSP System Administrator Guide* for more information.

4.2 What are the components of an intent type?

4.2.1 Intent type components

An intent type provides the logic for the creation of an intent, and for the execution of that intent. When a user creates an intent from the intent type, a Create Intent form launches requesting the required inputs. The YANG model, Target, and View components define the form. The Script and Resources components provide the logic required to realize the intent.

For detailed information about intent type design, see the intent types tutorial on the Network Developer Portal.

An intent type consists of the following components. The components appear as tabs in the Create and Edit forms.

- GENERAL
- TARGET
- YANG
- SCRIPT
- RESOURCES
- FORM
- VIEWS

4.2.2 GENERAL

The GENERAL tab includes the following panels.

General

The General panel shows the basic information about the intent type, for example, the name, version, labels and lifecycle state.

The **Mapping Engine** indicates the JavaScript engine used to create the intent type.

The **Labels** field lists the labels applied to the intent type. Labels are used to filter and group intent types. An intent type can have one or multiple labels.

If an intent type is compliant with a set of format requirements, such as Service Management, a label must be used to indicate what the intent type is configured for. For example, for an intent type to be imported by Service Management, it must have a ServiceFulfillment label.

The **Live state retrieval** check box dictates whether intent state attributes will be updated in the intent details on demand or when the intent is synchronized. Enable the check box to update state attributes on demand.

The **Composite** check box indicates a composite intent type.

The **Build** field displays the build number of the intent type from the intent type code. For Nokia-created intent types, the format is *release_x.y.z* where the numbers *x.y.z* are used for version control. The *x* number is the Version parameter in NSP. For example, if you install an intent type with the build number `22.9_2.1.1`, the intent type version will be 2, regardless of whether you have a version 1 in your system.

Policy

The Policy panel indicates the priority of the intent type and the targeted device. Policy configuration in the intent type helps NSP administer mass operations correctly.

The priority is used by NSP to determine the order of execution. An intent belonging to an intent type with a smaller priority number is executed by NSP before an intent belonging to an intent type with a bigger priority number. Setting the priority can ensure, for example, that port configurations are performed before the configuration of services that will require the ports.

The targeted device is used to allocate jobs so that only one job is executed at one time per device. This prevents delays and job timeouts due to a job attempting to run on a resource that is blocked by another job.

The targeted device is defined in JSON format using intent-type, targets, yang, inherit, and function names.

It can be defined in the following ways:

- target component of the current intent:

```
"targetted-device": [ {
  "target-component": "device-name"
 }]
```

- target component of a dependent intent:

```
"targetted-device": [ {
  "intent-type": "l2-infra",
```

(empty or omitted indicates this intent type)

```
  "target-component": "device-name"
 }]
```

- leaf of the YANG of the current intent:

```
"targetted-device": [ {
"intent-type": "l2-infra",
```

(empty or omitted indicates this intent type)

```
  "yang": "fiber:fiber/pon-port/device-name"
 }]
```

- same devices as a dependent intent:

```
"targetted-device": [ {
  "inherit": "true",
   "intent-type": "device-fx"
}]
```

- in special cases, we can use special logic in JavaScript:

```
 "targetted-device": [ {
  "function": "getTargettedDevices"
 }]
```

Migration

The Migration panel is displayed for intent types with a version number greater than 1. The migration table shows the configured migration and rollback paths between versions of the intent type.

4.2.3 TARGET

The target attribute of the intent type is used to uniquely identify any intent created from the intent type. The definition must be in JSON format.

The target attribute is made up of one or more target components.

All attributes are optional: the only requirement is that the intent is uniquely defined.

When an intent is created, the target components are displayed as input fields in the intent creation dialog. NSP uses the user inputs to create the target value.

The following table describes the target attributes.

Attribute	Description	Example
name	Internal name of the component This value is not displayed in the intent creation dialog.	device-id
value-type	Two types are supported: • STRING • NUMBER	STRING
i18n-text	The Create Intent form display name for the component	Device Name
order	The order in which this is displayed in the Create Intent form; 1 being the highest. You can give two components the same order; they appear side by side in the form.	1
pattern	Supported pattern check as supported by YANG pattern statement.	"AV-[a-zA-z0-9]+"
function-name	The name of the suggest function that provides a list of suggested inputs, if you want to use one. The suggest function is defined in the Script panel of the intent type.	"suggestDeviceNames" The function returns device names to populate the target attribute.
length	Supported length as supported by YANG length statement.	1..10

The following sample shows the structure of a target component definition.

Sample target

The following sample creates a target component:

```
{
  "target-component": [
    {
      "i18n-text": "Service Name",
```

```
        "name": "service-name",
        "pattern": "[A-Za-z0-9_]{1,}",
        "value-type": "STRING",
        "order": 1
      }
    ]
}
```

You can include more than one target component, if needed. The target value separates the two components with a hash; for example, `service#port`.

The sample target component is displayed in the form as shown in Figure 4-1, “Form with Sample target and Sample YANG” (p. 51). The user input becomes the target attribute.

4.2.4 YANG

The YANG panel provides an abstraction for the instance of the intent type.

The following types of arguments are supported:

- Configuration arguments: attributes where the user needs to provide a value
- State arguments: attributes where the intent type reports relevant state information

The YANG model must extend the Nokia IBN YANG definition. The arguments must use the `ibn:configuration` container.

4.2.5 SCRIPT

The SCRIPT tab contains the realization logic of the intent type. This tab contains the programming required for the intent configuration to be performed.

The script contains an intentObject and needs to implement some predefined functions. When a user triggers an action, the predefined functions are called by the framework to fulfill the task. These methods have implementation specific to an Intent type.

The following functions are optional.

- Validate runs at the time an intent is synchronized, either at or after intent creation. If the intent input fails the validation condition, the intent cannot be synchronized.
- Reconcile is the opposite of a synchronize function. It will pull the configuration from the associated object and update the intent configurations to match it.

Script engines

NSP uses the Nashorn JavaScript engine by default. The GraalJS JavaScript engine is also available.

GraalJS supports some ECMAScript features and capabilities that are not available with Nashorn. For more details, see the Network Intents information on the NSP Developer Portal.

Sample script

The following sample shows a basic script structure.

```
var RuntimeException = Java.type('java.lang.RuntimeException');



var prefixToNsMap = {

     "ibn" : "http://www.nokia.com/management-solutions/ibn",

     "nc" : "urn:ietf:params:xml:ns:netconf:base:1.0",

     "device-manager" : "http://www.nokia.com/management-solutions/anv",

};



var nsToModule = {

     "http://www.nokia.com/management-solutions/anv-device-holders" :
"anv-device-holders"

};



function synchronize(input) {

  var target = input.getTarget();

    var config = input.getIntentConfiguration();

    var topology = input.getCurrentTopology();

    //Retrieve the network state, it exists in the input argument.

    var networkState = input.getNetworkState().name();

    var customNetworkState = input.getCustomRequiredNetworkState();

    logger.info("######### NETWORK STATE "+ networkState);



    //If the network state is 'custom' then we must look in the
customRequiredNetworkState property to get the actual state
```

```
    if(networkState === 'custom'){
      logger.info("######### CUSTOM NETWORK STATE "+ customNetworkState);

      result.setSuccess(true);
      return result;
}

function audit(input) {
   var report = auditFactory.createAuditReport(null, null);
   // Code to audit goes here

   return report
}

function validate(syncInput) {
  var contextualErrorJsonObj = {};
  var intentConfig = syncInput.getJsonIntentConfiguration();

     // Code to validation here. Add errors to contextualErrorJsonObj.
  // contextualErrorJsonObj["attribute"] = "Attribute must be set";
     if (Object.keys(contextualErrorJsonObj).length !== 0) {
        utilityService.throwContextErrorException(contextualErrorJsonObj);
  }
}
/*
```

```
RECONCILE

The return object that is expected from a reconcile is ReconcileOutput

ReconcileOutput POJO defined like below

  updatedIntentConfiguration - filled when the intent configuration can
be adapted

  updatedStateXML - filled when the Intent State can be adapted

  alignmentState - filled if intent needs to be aligned state after
reconcile

  updateDependents - whether dependent intents to be updated

  syncDependents - whether dependent intents to be synched

  topology - filled if there is change in Topology /  Intent Dependency ,
XtraInfo

*/

function reconcile (input){

  var reconcileOutput;

  var topologyXtraInfo = null;

  var topologyXtraInfo = null;

  var target = input.getTarget();

  var topology = input.getCurrentTopology();

  var targettedDevice = input.getTargettedDevices();

  var networkState = input.getNetworkState();

  var customRNS = input.getCustomRequiredNetworkState();

  var intentConfigXml = input.getIntentConfiguration();



  //Here we are changing the configuration

  var testStringNode = intentConfigXml.getElementsByTagName
("test-string").item(0);
```

```
  if(testStringNode.getTextContent() == 'reconcile'){

    testStringNode.setTextContent("reconcile");

  }



  var updatedConfig = domUtils.documentToString(intentConfigXml);

   if (networkState == "delete") {

logger.info("Reconcile intent with input networkState {}", networkState);

        throw new RuntimeException("Reconcile with required-network-state
as 'delete' is not possible");

    }

    if (networkState == "suspend") {

logger.info("Reconcile intent with input networkState {}", networkState);

        return new ReconcileOutput(null, null, "false", false, false,
null);

    }

    //Return for reconcile

    return new ReconcileOutput(updatedConfig, null, "true", false, false,
null);

}
```

4.2.6 RESOURCES

The RESOURCES tab shows the list of resource files required to realize the intent. The resources are a library of files that the intent type can access.

Different intent types that target similar system configurations often use similar logic. The use of resource files allows the common logic pieces to be stored as framework and mapping files, and loaded when the intent runs.

The resource files can be in various formats, for example, YANG, JavaScript, or JSON.

The GraalJS script engine supports folders in the Resources library.

To configure resource libraries, click ⋮ **More**, **Resource Management** at the top of any view in the Network Intents path.

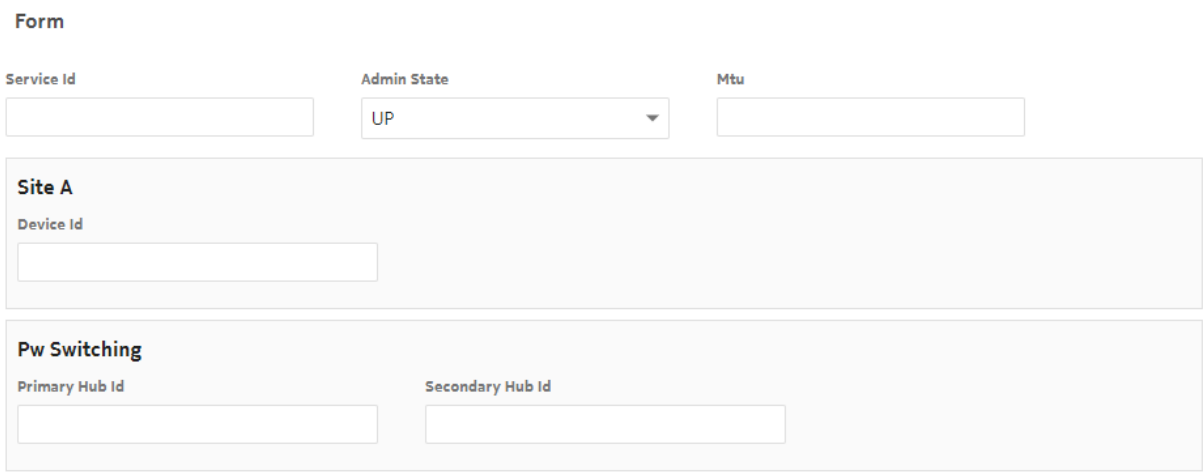
For detailed information about resource file creation, see the Network Intents tutorial on the Network Developer Portal.

4.2.7 FORM

The FORM tab shows how the Create Intent dialog is displayed when the user creates an intent. NSP implements a schema file to generate and preview the form. Choose a schema form in the Schemas drop-down to preview the form implemented by the schema form selected.

Intent types that are instantiated within Network Intents use the default schema form. Other schema forms, created from viewConfig files, may be available for intent types used by other applications.

Figure 4-1 Form with Sample target and Sample YANG



4.2.8 VIEWS

The VIEWS tab shows the list of view files available to the intent type. View files are used to augment the YANG to create the input form the user will see when creating an intent, or when using intent-based configuration in another application.

Nokia-signed intent types, which are installed using Artifacts, cannot be edited except to add, delete or modify view files.

The list of files includes:

- viewConfig files: a viewConfig file defines the changes to make to the attributes in the intent type YANG to define the fields that will appear in the input form and their properties. For example, if the YANG declares that the intent creation form will contain two string fields, a viewConfig file can provide names for the fields.

The NSP creates a default viewConfig file based on the YANG. A developer can make changes to the default or configure additional viewConfig files.

- schema forms: the schema form is automatically generated from the viewConfig file. NSP implements the schema form to create the input form.
- view.settings files: the view.settings file is automatically generated. It provides a link to the module and container defined in the intent type YANG to identify the YANG entry point.

For detailed information about View file configuration, see the ViewConfig section of the Network Intents tutorial on the Network Developer Portal.

Parent and child viewConfig files

A viewConfig file can be attached to another as a child. The child file inherits all the attributes and rules configured in the parent file.

The child file can add attributes or overwrite attribute values and rules.

For example:

- The parent file augments one attribute: `attr1`. The title of `attr1` is set to `Parent`.
- The child file augments two attributes: `attr1` and `attr2`. The title of `attr1` is set to `child` and `attr2` is set to `child2`.

On the FORM panel, if you choose the parent schema form, one attribute is changed. The title is `Parent`. If you choose the child schema form, two attributes are changed. The titles are `child` and `child2`.

4.3 What is a composite intent type?

4.3.1 Composite intent type

A composite intent type is an intent type that encompasses multiple child intent types.

An intent type that is a child of a composite intent type can be managed on its own, or included in multiple composite intent types.

Combining several intent types into a composite intent type allows you to perform operations on the child intent types collectively. For example, you could use a composite intent type for managing several services as an end-to-end service: creating an intent from the composite intent type can configure all three services at once, and you can audit or sync the intent to monitor and maintain configuration on the end-to-end service as a whole.

A composite intent type has the same required components as other intent types, with the addition of an XML mapping file. The mapping file provides information about how to apply the parameters from the composite intent to the child intents. For example, the composite intent type may have a parameter called Service 2 Name. The mapping file tells NSP to apply the value of Service 2 name to the Service Name parameter of a specified child intent.

When the user creates an intent for the composite intent type, the user configures the parameters in the composite. NSP creates intents for the child intent types, passing the user inputs to the child intents as specified in the mapping file. You can configure dependency to the intents, for example, child 2 requires child 1. With this configuration, NSP creates the intent for child intent type 1 first, then creates the intent for child intent type 2.

If no order is configured, all the intents are created at the same time.

The following restrictions apply:

- The intent types that you want to add as child intent types must be created before the composite intent type.
- Composite parameters cannot be changed in edit mode. You can edit a composite intent to change the name, for example, but not to change the list of child intents or change child intent dependency.

  If you need to change composite parameters, delete the composite intent type.

4.4 What is an intent type version?

4.4.1 Intent type version

Intent type versions are a tool to allow working with different intent type YANG models.

Changing certain attributes in an intent type can affect OSS integrations. If you need to change OSS-affecting attributes, you can create a new version of the intent type with the change, leaving earlier versions available to the OSS. Intents can be migrated between versions of the intent type when the OSS integration has been updated to work with the new version. Intents can be migrated from one version to another, avoiding the need to create them again.

If the new definition of the intent is incompatible with the previous version, for example if a new attribute has been added, you can configure migration handlers and rollback handlers as part of the script of the newer version.

Migration and rollback handler functions can also be used to indicate how approved misalignments are handled when intents are migrated.

The Migration table in the GENERAL tab of the new version shows the configured migration paths and the functions that support migration and rollback.

- Supported version: the Supported Version column in the Migration table refers to the “from” version in a migration path. When a version is created, for example, version 3, the migration table is populated with rows for versions 1 and 2. However, migration and rollback handler functions are not automatically assigned.
- Migration handler: the migration handler function ensures that your data instance remains in line with the updated YANG model, for example by adding a new attribute that was included in the newer version.
- Rollback handler: the rollback handler function manages migration from a newer version to an older version, for example, removing an attribute that was added by the new version.

See the Network Intents tutorial on the Network Developer Portal for details about scripting migration.

4.5 What is intent dependency?

4.5.1 Intent dependency

Many use cases involve multiple intent types where one intent can be dependent on one or many other intents, for example when both intents target the same network object.

Network Intents supports two types of dependency:

- Existence dependency: An intent's existence is dependent on the existence of one or many other intents. When existence dependency is present, the direct dependents and the direct dependencies appear in the intent details page. If an intent has direct dependencies, it can't be deleted unless the depended intents are deleted first.
- Argument dependency: An intent's configuration logic could be dependent on the values of another intent's arguments.

Intent dependency is configured as part of the Script of an intent type. See the Network Intents tutorial on the Network Developer Portal for details.

4.6 What are intent type states?

4.6.1 Intent type states

The state of an intent type shows whether it is available for intent creation:

- Draft: intents cannot be created.

  When an intent type is cloned or a new version is created, the clone or new version is created in Draft state.
- Released: intents can be created.
- Phased-out: intents cannot be created. Existing intents can be synced and audited.

A user with the required permissions can edit an intent type in any state, including changing the state.

If intents have been created from an intent type, it is a best practice to create a new version of the intent type and edit the new version instead of editing the version with intents present.

Intent type procedures

4.7 How do I create an intent type?

4.7.1 Steps

1

Open **Network Intents**, **Intent Types**.

2

Click + **CREATE**.

3

In the **Create Intent Type** form that opens, enter a name for the intent type.

Important! Intent Type names must meet the following requirements.

- The name must begin with a letter.
- The name of an intent type can only contain lowercase letters, dashes or underscores. If other characters are present, the intent type cannot be created.

4

Configure the parameters in the **GENERAL** pane as required:

- Click in the Mapping Engine field to choose the JavaScript engine.
- Click **Composite** to create a composite intent type.
- Click **Live state retrieval** to enable updating intent state attributes in the intent details on demand.
- Enter labels in the **Labels** field. The intent type must have at least one label.

5

Configure the parameters in the **Policy** pane.

If you are not creating a composite intent type, proceed to Step 7.

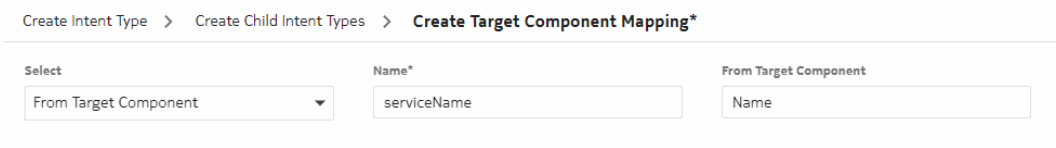
6

If applicable, configure the parameters in the **Composite Intent Types** pane:

1. Click + **ADD** to add child intent types.
2. In the **Name** field, enter an alias for the child intent type, to be used by the mapping file.
3. In the **Intent Type** and **Version** fields, enter the name and version of the intent type you are adding.
4. The **Requires Child Intent** parameter creates a dependency between child intent types. If the intent type you are adding is dependent on another child intent, enter the name of the other intent and click + .

5. Configure target component mapping for each target component in the child intent type:
   a. Click **+ ADD**
   b. Choose **From Target Component** or **From Data Component**.
   c. In the **Name** field, enter the To mapping, that is, the component in the child intent to configure.
   d. In the **From Target Component** or **From Constant Value** field, enter the From mapping, that is, the composite intent type component or data value to apply to the child intent.

      The following sample shows mapping of a component in the composite intent called Name to a child intent component called serviceName.



   e. Click **ADD** to finish adding the target component mapping.

      Repeat this step as needed to configure mapping for all target components in the child intent type.
6. Click **ADD** to finish adding the child intent type.

   Repeat this step as needed to add another child intent type.

7

In the **TARGET** tab, configure the identification information for intents created from the intent type.

Modify the default text, or enter, copy or paste as needed.

8

In the **YANG** tab, configure the form details for creation of an intent.

1. Choose the YANG module in the panel on the left.
2. Modify the default text, or enter, copy, or paste as needed.

9

Add additional YANG modules as required.

1. Click **Yang +**.
2. In the **Add Yang** form that opens, enter the name of the module and click **ADD**.
3. Enter or paste the YANG text in the panel.
4. Repeat as needed to add more modules.

10

In the **SCRIPT** tab, configure the script for executing the intent.

Modify the default text, or enter, copy, or paste as needed.

11

In the **RESOURCES** tab, add or modify resources as required.

1. Click **Resource** + .
2. In the Add Resource form that opens, enter the name of the resource and click **ADD**.
3. Enter or paste the resource file in the panel.
4. If the intent type has child intents, a generated mapping file called config-transformer is present. Verify the mapping information in the file and make changes if required.

Resource libraries can be configured using NSP Resource Management. To open Resource Management, click ⋮ **More**, **Resource Management** at the top of any Network Intents view.

12

Click **CREATE**.

END OF STEPS

4.8 How do I import an intent type from my computer?

4.8.1 Purpose

This procedure describes importing a zip file of intent types from your computer to Network Intents. You can also import intent types using Artifacts; see 2.3 “How do I install an artifact bundle?” (p. 20). If your zip file includes artifacts other than intent types, you must use Artifacts to perform the import.

In NSP 21.11, the engine for network intents was updated to Java 11, which fixed a bug that was present in Java 8. Importing intent types that were created before the update may cause errors. To fix the issue, method declaration must be updated to use a supported style pattern. Contact Nokia for more information.

4.8.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Click **IMPORT**.

3

In the form that opens, navigate to the file you want and click **Open**.

4

If the intent type and version are already present, a dialog opens:

a. Click **OVERWRITE** to overwrite the existing intent type.

b. Click the check box to import the intent type with a new version number, and click **IMPORT**.

c. Click **CANCEL** to cancel the import.

END OF STEPS

4.9 How do I clone an intent type?

4.9.1 Purpose

Use this procedure to create a copy of an intent type. You can edit the clone as needed and keep the original intent type intact.

Note: If you clone a signed intent type, the clone is not signed.

4.9.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Choose an intent type and choose ⋮ (Table row actions), **Clone Intent Type**.

3

In the **Clone Intent Type** form, enter a name for the clone.

4

Click **CLONE**. The cloned intent type is added to the intent types list in draft state.

5

See 4.10 “How do I edit an intent type?” (p. 57) to edit the clone.

END OF STEPS

4.10 How do I edit an intent type?

4.10.1 Steps

1

Open the **Edit Intent Type** form:

a. From **Network Intents**, **Intent Types**, select an intent type and click ⋮ (Table row actions), **Edit**.

b. From the intents list for an intent type, click **EDIT INTENT TYPE**.

2

Make changes as needed.

3

Click **UPDATE**.

END OF STEPS

4.11 How do I add or change a View file?

4.11.1 Purpose

Use this procedure to add or update a viewConfig file. The `default.viewConfig` file is always used by intents created from the Intent Types view. Other viewConfig files can be created after intent type creation.

The ArtifactAdmin label is used to indicate that an intent type has been installed using Artifacts. For these intent types, if you want to make changes to views, Nokia recommends creating a new view instead of modifying views. This will allow you to export the views you create and save them to be added into any future versions of the intent type created.

4.11.2 Steps

1

Open the **Edit Intent Type** form:

a. From **Network Intents**, **Intent Types**, select an intent type and click ⋮ (Table row actions), **Edit**.

b. From the intents list for an intent type, click **EDIT INTENT TYPE**.

2

In the **VIEWS** tab, add or modify view files as required.

1. Click **Views** +.
2. In the Add View form that opens, enter the name of the view and click **ADD**.

   A *name*.viewConfig file is created in the list at the left of the page.
3. Click on the viewConfig file in the list and enter or paste the view configuration information in the panel.
4. Click **SAVE VIEWS**. A *name*.schemaForm file is automatically created.
5. If needed, click on the **FORM** tab and select *name*.schemaForm from the drop-down to test the input form.

If the intent type includes an RPC action, add a View file called rpc_*schema*.viewConfig to provide user interface information for the RPC action.

See the Network Intents tutorial on the Network Developer Portal for more information if needed.

3

To attach a viewConfig file as a child of another:

1. Select a viewConfig file and click ⋮ (Table row actions), **Attach to parent**.
2. In the form that opens, click in the **Parent ViewConfig** field and select a viewConfig file as the parent.
3. Click **ADD**.

The child viewConfig and its schema form appear as a child in the list, below the parent viewConfig file.

Click to display the Change log. The change log shows the attributes that have been augmented and indicates which viewConfig implements the change.

4

To detach a child viewConfig file from its parent, select the child viewConfig file and click ⋮ (Table row actions), **Detach from parent**.

5

Click **UPDATE**.

END OF STEPS

4.12 How do I delete an intent type?

4.12.1 Steps

1

Open **Network Intents**, **Intent Types**.

2

Select an intent type.

3

Click ⋮ (Table row actions), **Delete**.

4

In the confirmation dialog box, click **DELETE**.

END OF STEPS

4.13 How do I change the state of an intent type?

4.13.1 Purpose

Use this procedure to update the deployment status of an intent; see 5.2 “What is a network state?” (p. 65).

4.13.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Click on an intent type and click ⋮ (Table row actions), **Edit**.

3

In the **GENERAL** panel, choose the required state from the **Lifecycle State** drop-down list.

4

Click **UPDATE** to change the state.

END OF STEPS

4.14 How do I create a new version of an intent type?

4.14.1 Purpose

You can create a new version starting from any version of an intent type. For example, if there are three versions, you can create a new version of version 2. The new version is numbered 4, but it does not include any updates made in version 3.

The new version number is auto-assigned and cannot be changed.

4.14.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Select an intent type.

3

Click ⋮ (Table row actions), **New Version**.

The new version is created in draft state.

4

Configure migration and rollback functions as needed.

1. Select the new version from the intent types list and click ⋮ (Table row actions), **Edit**.
2. Add the migration and rollback functions to the SCRIPT.

5

Configure migration paths.

1. In the GENERAL panel, click + **ADD** in the Migration pane.
2. In the form that opens, configure the parameters and click **ADD**.
3. Repeat steps1 and2 to configure additional migration paths as needed.

END OF STEPS

4.15 How do I export an intent type or its view files?

4.15.1 Purpose

Exporting an intent type exports all components other than the view files. View files can be exported separately. The exported files can be repackaged together outside NSP; see the Network Intents tutorial on the Network Developer Portal.

Exported view files can also be reused and customized in another intent type with the same override properties.

4.15.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Select an intent type.

3

Perform the exports as needed:

a. Click ⋮ (Table row actions), **Export Intent Type**.

b. Click ⋮ (Table row actions), **Export Views**.

The files are exported in zip format.

END OF STEPS

4.16 How do I configure user access to an intent type?

4.16.1 Purpose

Use this procedure to grant or restrict access to an intent type for a user group. Before you can configure user access, user groups must be created; see the *NSP System Administrator Guide*.

This procedure requires an administrator role.

4.16.2 Steps

1

Log in to the NSP as an administrator.

2

Open **Network Intents**, **Intent Types**.

3

Select one or more intent types.

Press and hold the Shift key to select multiple intent types.

4

Open the User Access form:

- If you have selected one intent type, click ⋮ (Table row actions), **User Access**
- If you have selected multiple intent types, click **User Access** at the top right of the view.

5

In the **User Access** form, configure access to the selected intent types:

1. Choose an action from the drop-down list at the top right of the form.
2. If you chose **Choose access**, configure the drop-down lists for each group in the panel at the right.

6

Click **SAVE**. The user access is updated.

END OF STEPS

5 Intents

Overview

5.1 What is an intent?

5.1.1 Network intents

An intent is an instance of an intent type. The intent provides inputs to the intent type and executes the configuration.

For example, you can create an intent type for creation of a VPRN service. The intent type provides the parameters to set and the script to perform the configuration. When you are ready to create the VPRN, you create an intent to specify the parameter values and implement the service creation.

5.2 What is a network state?

5.2.1 Network state

The network state of an intent shows its deployment status to the network.

NSP offers several options for the network state of an intent. The intent type script declares the recommended implementation of the state. Synchronizing the intent performs the implementation.

When an intent is audited, the Audit operation compares the configuration on the target to the network state.

See the Network Intents tutorial on the Network Developer Portal for information about how network states are programmed, and how they are managed by the API.

The following table provides an example of the way the network states are recommended to be used, for an intent type that creates services.

Network State	Description based on a service creation example
Saved	The intent is created and the service entity name will be reserved. The user can edit the intent to resume creation when needed.
Planned	The service resources are reserved.
Suspended	The intent must be activated before it can be deployed.
Activate	Remove suspension and send the configuration to the network.
Deployed	The desired configuration is sent down to the network.
Not Present	The service is removed from the network, but its resources remain reserved.

5.3 What is an intent audit?

5.3.1 Identify misalignments

Perform an intent audit to identify any mismatch between the intended configuration and network state and the actual configuration in the network.

If a mismatch exists, the **Aligned** field is updated to **Misaligned**.

Note: You can also choose ⋮ (Table row actions), **Mark as misaligned** to manually set the alignment status to Misaligned.

If a misalignment is found, the audit report form provides details. Misalignments are presented in the following categories:

- misaligned attributes: the value of an attribute in the network (actual value) is different from the value in the intent (expected value)
- misaligned objects: an object that is configured in the intent is missing in the network
- undesired objects: an object that is not configured in the intent is present in the network

You have several options to resolve misalignments:

- Synchronize: send the intent configuration to the misaligned target, updating the configuration to match the intent.

  If you have approved misalignments, the approved misalignments will not be changed when you synchronize.
- Reconcile: retrieve the target configuration and update the intent to match the configuration

5.4 What is the difference between synchronize and reconcile?

5.4.1 Source of truth

For the synchronize operation, the intent is the source of truth. Misalignments are corrected by updating the network to match the NSP.

If approved misalignments are present, the synchronize operation skips them. For example, if you have two misalignments and have approved one, the synchronize operation will only update the misalignment you have not approved.

For the reconcile operation, the network is the source of truth. The intent is updated in the NSP to match the network.

5.5 What are approved misalignments?

5.5.1 Partial synchronize

Approved misalignments are differences between the current configuration in the network and the intended configuration in the NSP, which the user has allowed. Approved misalignments are not flagged by a future audit or overwritten when the intent is synchronized.

You can approve changes and remove approvals from the audit report for an intent; see 5.10 “How do I approve misalignments?” (p. 70).

You can view the list of approved misalignments for all intents and remove approvals from the Network Intents, Approved Misalignments view; see 5.11 “How do I modify approved misalignments?” (p. 71).

Procedures

5.6 How do I create an intent?

5.6.1 Steps

1

Open **Network Intents**, **Intent Types**.

Open the Create Intent form:

a. Select an intent type and click ⋮ (Table row actions), **Create Intent**.

b. Double click on an intent type to open **Network Intents**, **Intent Type** *name_version*, **Intents**, and click + **CREATE**.

2

Choose the network state.

3

Configure the parameters.

4

Click the **Synchronize** check box if needed to synchronize the configuration with the network as soon as the intent is created.

5

Click **CREATE**.

END OF STEPS

5.7 How do I view intents?

5.7.1 Viewing intents for an intent type

From **Network Intents**, **Intent Types**, double-click on an intent type to view the list of intents in the network for the intent type. Double-click on an intent in the list to open an information page showing intent details and dependencies.

The intent details page shows the intent state (active or suspended), the intent type, the intent version, timestamps for the last audit and synchronization, alignment state, and reason for misalignment, if applicable.

The **DETAILS** tab in the center panel shows the parameter values.

The **INTENT DEPENDENCY** tab shows dependency relationships:

- Direct Dependents: intents that depend on this intent
- Direct Dependencies: intents that this intent depends on

5.8 How do I modify an intent?

Note: Target components can't be edited. To deploy an intent type to a different target, create another intent.

5.8.1 Steps

1

Open **Network Intents**, **Intent Types**.

2

Double-click on an intent type to open **Network Intents**, **Intent Type** *name_version*, **Intents**.

3

Click **EDIT INTENT**.

4

In the form that opens, update the parameters.

The **Synchronize** check box is checked by default. Uncheck this check box if you do not want the intent to synchronize after you edit it.

5

Click **UPDATE**.

END OF STEPS

5.9 How do I audit an intent?

5.9.1 Steps

1

Open **Network Intents**, **Intent Types**.

2

Double-click on an intent type to open **Network Intents**, **Intent Type** *name_version*, **Intents**.

3

Select an intent and choose ⋮ (Table row actions) **Audit Intent**.

4

When the audit completes, the alignment status is updated in the Aligned column. If the intent is misaligned, the Audit Report form opens, showing the misalignments.

END OF STEPS

5.10 How do I approve misalignments?

5.10.1 Steps

1

Open **Network Intents**, **Intent Types**.

2

Double-click on an intent type to open **Network Intents**, **Intent Type** *name_version*, **Intents**.

3

Select an intent in misaligned status and choose ⋮ (Table row actions) **Last Audit Report**.

The Audit Report form opens.

4

The Audit Report form displays the misalignments by category. An indicator appears beside the category name with the number of misalignments present in the category, for example, **MISALIGNED ATTRIBUTES [2]**.

Click the category name to view a table of misalignments in the category.

5

To approve a change, click the check boxes in the row for the change you want to approve and click **APPROVE SELECTED**.

The change is added to the **APPROVED MISALIGNMENTS** panel and the number indicator is updated.

6

Open other category panels and approve additional changes as needed.

7

View the list in the **APPROVED MISALIGNMENTS** panel.

If you don’t want to approve any change on the list, uncheck the check box.

8

Click **APPLY**. The approved misalignments are added to the **Network Intents**, **Approved Misalignments** view.

The intent is audited automatically.

END OF STEPS

5.11 How do I modify approved misalignments?

5.11.1 Purpose

You can delete approved misalignments for any intent from the **Network Intents**, **Approved Misalignments** view. Deleting an approved misalignment removes the exception for the misalignment. That is, the change is declared misaligned when the intent is audited, and overwritten when the intent is synchronized.

To add approved misalignments, perform 5.10 “How do I approve misalignments?” (p. 70).

5.11.2 Steps

1

Open **Network Intents**, **Approved Misalignments**.

The view shows a list of intents with approved misalignments, and the number of approved misalignments for each.

2

Double-click on an intent to view the list of approved misalignments.

3

In the form that opens, click Delete to remove approval for a change.

4

Click **OK**. The approved misalignments are added to the **Network Intents**, **Approved Misalignments** view.

When you audit the intent again, the change will be marked as a misalignment.

END OF STEPS

5.12 How do I migrate an intent?

5.12.1 Purpose

Use this procedure to migrate a single intent. To perform a mass migration, create a policy.

If you migrate an intent with approved misalignments, the approved misalignments will be handled based on the migration settings in the parent intent type.

5.12.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Double-click on the intent type version that includes the intent you want to migrate.

3

From the list of intents that appears, double-click on the intent you want to migrate.

4

Click **MIGRATE INTENT**.

5

In the form that opens, choose the version to migrate the intent to.

6

Click **OK**.

END OF STEPS

5.13 How do I execute an action from an intent?

5.13.1 Purpose

If your intent includes RPC actions, you can run the actions without auditing or synchronizing the intent.

5.13.2 Steps

1

Open **Network Intents**, **Intent Types**.

2

Double-click on the intent type includes the RPC action you want to execute.

3

From the list of intents that appears, double-click on an intent.

The intent information page opens with the list of available actions in a panel on the right of the screen.

4

Click on the action to execute.

a. If the action has no input parameters, it executes immediately.

b. If the action has input parameters, a form opens. Configure the parameters and click **OK**.

END OF STEPS

6 Intent Policies

6.1 What is a policy?

6.1.1 Mass operations for intents

Policies trigger and execute intents. You can create a policy to perform a mass operation on intents that fit specified criteria; for example, synchronize all intents that have not been synchronized in the last 10 days.

You can trigger policies manually, or configure them to execute automatically on a schedule. Scheduled policies help you to automatically manage your intents network-wide.

6.1.2 Editing and deleting policies

You can edit or delete a policy from the Table row actions menu; see “How do I navigate a list?” in the *NSP User Guide*.

6.2 How do I create an intent policy?

6.2.1 Steps

1

Open **Network Intents**, **Policies**.

2

Click + **CREATE**.

3

In the **Create Policy** form that opens, update the required fields in the General panel.

4

In the Action panel, configure the operation for the policy to perform.

- **Audit**:
  Audit the intents.
- **Mark as Misaligned**:
  Mark target intents as misaligned.
- **Migrate**:
  Migrate target intents to a new version of their intent types.
- **Modify and Synchronize**:
  Modify and synchronize intents of a specified intent type and intent type version.
- **Synchronize**:

Synchronize the intents.

5

In the Action panel, configure the scope of the policy:

1. Click **+ ADD** under the **Scope (Match all)** parameter.
2. Combine filters from the **Filter On** drop-down list with operators and values to create filters, for example `'sync-timestamp' older-than '3' days`.

If the Action is Modify and Synchronize or Migrate, the scope must include an Intent Type and Intent Type Version.

6

If the Action you configured is Modify and Synchronize, configure the Operation Arguments panel.

7

In the Execution panel, modify the default values as needed.

8

In the Trigger panel, click **+ ADD** under the **Time** parameter to create a schedule for the policy.

9

Click **CREATE POLICY**. The policy is added to the list.

END OF STEPS

6.3 How do I trigger a policy?

6.3.1 Steps

1

Open **Network Intents**, **Policies**.

2

Select a policy.

3

Click ⋮ (Table row actions), **Trigger** ⊙.

4

In the Confirm dialog, click **OK**.

The policy is triggered.

End of steps

6.4 What is a policy action?

6.4.1 Policy action

A policy action is an instance of a triggered policy.

Choose **Policy Actions** from the drop-down list at the top left of the screen to open the **Network Intents**, **Policy Actions** view.

The Policy Actions view shows the list of policy actions in Network Intents. The list shows the total job count, error and failed jobs along with status for all of the policies.

Double-click on an action to open a policy action details page, showing the job count, error count, and the list of jobs and their status.

7 Mediators

7.1 What is a mediator?

7.1.1 Mediator

Mediators serve as communication proxies between Network Intents and other systems, such as NSP components that manage network devices, or external controllers. The use of a mediator removes the need for Network Intents to provide authentication credentials to reach the desired endpoint.

The information on the Mediators views may be useful to administrators for troubleshooting communication issues to the underlying components.

7.2 What mediators are available?

7.2.1 Mediator types

NSP is shipped with mediators for model-driven mediation (the MDC mediator) and for classic management (the NFM-P mediator). Depending on how a networking device is managed, the appropriate mediator is used.

The NFM-P mediator can be used to provide helper functions to the SAM-O XML API and to other applications that use intents, such as Service Management and Inventory Configuration Management. To allow this, the XML API username and password attributes in the helm chart values file (values.yaml) must match the NSP credentials of an active user with the OSS Management role.

Multiple NFM-P mediators may be used for multi-NFM-P deployment scenarios. These mediators, in addition to simplifying communication, provide a high-level inventory of the devices they manage. This allows NSP to provide information about which mediator is in use for communication with a specific device.

In addition to the NFM-P and MDC mediators, there is a common NSP mediator that can communicate with any NSP component, which allows for advanced use-cases like path control or telemetry collection controlled by an intent type.

Generic mediators can be configured to integrate with external controllers; see the *NSP System Administrator Guide* for configuration information. A generic mediator can be used with any REST or RESTCONF endpoint.

7.3 How do I edit a mediator?

7.3.1 Steps

1

Open **Network Intents**, **Mediators**.

2

In the Attributes panel of the INFO tab, configure the parameters as needed.

Parameters with an asterisk(*) are required.

3

Click **UPDATE**.

END OF STEPS

7.4 How do I delete a mediator?

7.4.1 Steps

1

Open **Network Intents**, **Mediators**.

2

To remove the mediator from the NSP GUI, click **DELETE**.

3

To remove the mediator from the NSP system, perform a helm uninstallation; see “How do I configure a generic mediator?” in the *NSP System Administrator Guide*.

Note: If a mediator is deleted in error, restart the mediator pod to re-register it with the NSP system.

END OF STEPS

Part V: Workflows

Overview

Purpose

Describes the use of workflows for operators.

Contents

8 Workflows

8.1 Overview

8.1.1 What are Workflows?

Using workflows in NSP, you can create automated procedures and closed loop automation using Nokia NSP or 3rd party APIs.

Some example use cases:

- Node software upgrades
- Service activation tests
- Service fulfillment invoking pre and post deployment workflows
- Customizable policy logic for other applications
- Mass migration of services from one tunnel type to another
- One place scheduling for all NSP REST APIs and node CLI
- Centralized orchestration for Nokia NSP and 3rd party vendor APIs

NSP uses Mistral DSL v2 which is based on YAML. YAML defines expressions in workflow and action definitions. The formatting of the YAML must comply with Mistral DSL v2 specifications. For more information about Mistral DSL v2, the workflow hierarchy and associated attributes, refer to the Mistral DSL v2 specification online.

Certain functions within NSP workflows use Python, such as triggers and the nsp.python system action. NSP uses Python version 3.11. Any custom actions created by a user must be compliant with this version.

8.1.2 Developer information

The Network Automation Guide provides an overview for operators.

For detailed information about the following topics for developers, see the Workflows tutorial on the Network Developer Portal.

- Using Workflow APIs
- Creating workflows, including working with schema forms
- Working with Jinja2 templates
- Working with environments
- Working with ad-hoc actions
- Working with Kafka triggers
- Workflow signatures

8.1.3 API support

NSP Workflow functions are available for OSS using programmable APIs. For general information about developer support, visit the Network Developer Portal

For specific documentation about APIs for Workflows, click on API Reference in the Workflow Manager row.

8.1.4 Developer mode

If developer mode is disabled in the NSP, users cannot create, import, modify, or delete workflows, actions, or Jinja2 templates.

See the *NSP Installation and Upgrade Guide* for more information.

8.1.5 Access control

User groups are assigned access to menus, network resources, and Analytics resources by assigning roles in the NSP. Action permissions are assigned to roles.

Note: Disabling developer mode overrides access control settings. If developer mode is disabled, restricted functions are restricted for all users.

The following table describes scopes that are specific to workflows.

Scope	Available operations
Import	Import all available file types
Write Workflow Artifacts	Create and Edit workflows
Publish Workflow Artifacts	Publish workflows and related artifacts
Manage Executions	Pause, resume, cancel and delete workflow and action executions
Manage Environments	Perform actions in the Environments view
Manage Triggers	Manage Kafka triggers and workflow schedules
Debug	Debug using Flow views, the YAQL evaluator, or code definition views
Execute Workflow Artifacts	Execute workflows and actions. Pause, resume, or cancel workflow executions created by the same user.

The scopes are combined to create the following roles:

- Developer:
  - Import
  - Write Workflow Artifacts
  - Publish Workflow Artifacts
- Troubleshooter:
  - Execute Workflow Artifacts
  - Manage Executions
  - Debug
- Operator:
  - Manage Executions
  - Manage Environments
  - Manage Triggers

  - Execute Workflow Artifacts
- Network Engineer
  - All scopes

Depending on your access settings, some of these pages or operations may not be available to you. See the *NSP System Administrator Guide* for more information.

9 Using workflows

9.1 What is a workflow?

9.1.1 Workflow

A workflow represents a process. Each workflow consists of at least one task.

The **Workflows**, **All Workflows** view shows the workflows created in or imported to NSP. If the Status field shows PUBLISHED, the workflow is ready to run.

Double-click on a workflow to open an Info page. From the Info page, choose from the drop-down list to navigate to workflow-specific pages, such as the YAML definition and Flow diagram.

9.1.2 Input form

The input form is one of the pages available from the Info page of a workflow.

The user input parameters required by a workflow are declared in the input form. The value of each parameter is a JSON-compliant type such as number or string, a dictionary, or a list. It can also be an expression to retrieve a value from a task context, or any of the mentioned types containing inline expressions.

When the workflow is executed, the configured input form is presented in the Create Execution form. You can use the JSON schema to create form properties such as lists, choices, and fields with autocomplete.

If the developer has included execution mode options in the workflow script, the user input form includes the option for the operator to enable the mode. The following options can be included:

- Dry Run: execute the workflow without pushing any information to the network.
- Compare: execute the workflow and provide a summary of what has changed, or, if Dry Run is also enabled, what would be changed if you set Dry Run to false
- Force: execute the workflow with predefined user input values.

See the Workflows tutorial on the Network Developer Portal for detailed information.

9.1.3 Tags

Tags are used to filter and group objects.

If a workflow is compliant with a set of format requirements, such as Service Fulfillment or Kafka triggers, a tag must be used to indicate what the workflow is configured for. For example, for a Kafka trigger to launch a workflow, the workflow must have a KafkaTrigger tag.

You can add tags to group and filter workflows according to your needs, for example, add a vendor tag to easily find workflows for the vendor’s NEs.

9.1.4 Deleting workflows

You can delete a workflow from the Table row actions menu; see “How do I navigate a list?” in the *NSP User Guide*. If there is a Kafka Trigger associated with a workflow, the Kafka Trigger must be removed before the workflow can be deleted.

9.2 How do I add a workflow?

9.2.1 Workflow creation options

You have several options for adding workflows to NSP.

- Create a workflow. This procedure provides general steps. See the Network Automation tutorial on the Network Developer Portal for detailed information.
- Import a workflow from your computer.
- Import a workflow from Git.
- Install a workflow using Artifacts; see 2.3 “How do I install an artifact bundle?” (p. 20).
  Note: if an installed workflow appears with a red signature icon, there is a problem with the workflow. Try downloading and installing again. Contact Nokia if the problem persists.
- Clone a workflow and edit the clone as needed.

Note: Here are some things to know when setting up workflows.

- A workflow must pass validation before it can be saved in NSP. If you import a workflow that is not valid or clone a workflow and make changes that don’t pass validation, you must update it to fix the errors before you can add it to the workflow list.
- Workflows are imported in the DRAFT state. The state must be changed to PUBLISHED before the workflow is available to all users.

9.2.2 Example workflow

Workflow examples are available in the creation form. You can choose the best example for your needs and use it as a guide, or delete it and insert your own YAML code.

The example workflows `cleanup_WFM` and `createNspUser` are available by default. Any workflow with the `example` tag will also be available in the creation form.

9.3 How do I create a workflow?

Attention: Workflow names must only contain alphanumeric characters.

9.3.1 Steps

1

Open **Workflows**, **All Workflows**.

2

Click **+ WORKFLOW**.

3

In the **Create Workflow** form that opens, enter your code in the YAML panel:

1. If necessary, click in the Select Workflow field to change the 9.2.2 “Example workflow” (p. 88) provided.
2. Update the YAML as needed.

4

Click **VALIDATE & UPDATE FLOW**. Resolve any validation errors that appear.

After the YAML has been validated, you can click on the **Flow** tab to see the Flow diagram if needed.

5

Click **CREATE**.

The workflow is created in DRAFT status.

END OF STEPS

9.4 How do I import files from my computer?

9.4.1 Purpose

Files with various filename extensions can be imported:

- packages (.zip)
- workflows (.yaml)
- environment files (.env)
- actions (.action)
- Jinja2 templates (.jinja)

Multiple files of different types can be imported in the same operation. Workflows are imported in DRAFT status.

Workflow zip files can contain input files in JSON format. The JSON file should have the same name as the workflow, for example, workflow1.json.

9.4.2 Steps

1

Open one of the following:

a. **Workflows**, **All Workflows**

b. **Workflows**, **Actions**

c. **Workflows**, **Environments**

d. **Workflows**, **Jinja2**

Note: You can import any accepted file type from any page that supports importing.

2

Open the Import form:

a. From **Workflows**, **All Workflows**, click the **IMPORT** drop-down list and choose **File System**.

b. From another view, click **IMPORT** .

3

In the **Import** form, perform one of the following steps to select your file.

a. Navigate to a file or files on your computer and drag them to the panel at the top of the form.

b. Click in the panel at the top of the form to navigate to the file you want. Click **Open**.

The file names appear in the Files to Import panel.

4

Click **IMPORT**.

The Import Results panel displays the status of the import of each file, with explanations for failed imports.

5

Click **CLOSE**.

END OF STEPS

9.5 How do I export files?

Tip: Configured files can be exported: workflows, ad-hoc actions, environments, and Jinja2 templates.

9.5.1 Steps

1

Open one of the following:

a. **Workflows**, **All Workflows**

b. **Workflows**, **Actions**

c. **Workflows**, **Environments**

d. **Workflows**, **Jinja2**

Note: You can import any accepted file type from any page that supports importing.

2

Select the object you need to export.

3

Choose ⋮ (Table row actions), **Export** or ⋮ (Table row actions), **Export Zip**.

Depending on your browser settings, the file is downloaded or a confirmation dialog opens.

4

To export the Flow diagram of a workflow:

1. Open **Workflows**, **All Workflows**.
2. Double-click on a workflow to open the Info page
3. Choose Flow from the drop-down list.
4. Click **Screenshot Flow** to download the Flow diagram as a .png file.

5

To export the Flow diagram of a workflow execution:

1. Open **Workflows**, **Workflow Executions**.
2. Double-click on a workflow execution to open the Info page
3. Choose Flow from the drop-down list.
4. Click **Screenshot Flow** to download the Flow diagram as a .png file.

6

Save the file as needed.

END OF STEPS

9.6 How can I interact with the Nokia Git repository from NSP?

9.6.1 Purpose

The Nokia nsp-workflow repository on GitHub hosts sample workflows to be used with NSP. Use these workflows to get to know workflows in NSP, or as blueprints for your own workflows. You can also check for updates to workflows you have imported, or open the Nokia nsp-workflow repository.

The default environment uses the Nokia repository on GitHub. You can update the default environment to point to your own Git if needed.

9.6.2 Prerequisite

To import from the Nokia GitHub repository, you must add the following URLs to the allowed hosts list on the NSP server:

- *raw.githubusercontent.com*
- *api.github.com*

You can update the allowed hosts via postman, or using `curl` from the command line on the NSP server:

```
# curl -kv --request POST https://VM
IP/session-manager/api/v1/whitelist/allowedHosts --header 'Content-Type:
application/json' --data-raw '{ "host": "url" }' --header "Authorization:
Bearer authorization token ↵
```

where *VM IP* is the IP address of the NSP server, *url* is the URL you are adding, and *authorization token* is a bearer token requested from NSP.

9.6.3 Steps

1

Open **Workflows**, **All Workflows**.

2

Click the **IMPORT** drop-down list and choose **GitHub** to open the **Import from Git** form.

3

To import a workflow from Git:

1. In the **Import from Git** form, choose **Import Packages** from the panel at the left of the screen.
2. Explore the list of workflows, or filter the list using the filter field.

   If a workflow has not been added to your NSP, the **IMPORT** button is available. If updates are available for a workflow you have imported, the **UPDATE** button is available.
3. Click the web link to view the workflow in the Nokia repository, or click **More actions** ••• for details.
4. Click **IMPORT**. The workflow is imported and the **IMPORT** button changes to an indicator that your copy of the workflow is up to date.

4

To import updates to workflows you have saved:

1. In the **Import from Git** form, choose **Updates** from the panel at the left of the screen.
2. Click **Check for Updates** to refresh the list of available updates.
3. Explore the list of updates, or filter the list using the filter field.
4. Choose an update from the list and click **UPDATE**, or click **UPDATE ALL** at the top of the form.

   The updates are completed.

5

To open the repository, in the **Import from Git** form, choose **OPEN REPOSITORY** from the panel at the left of the form.

The Nokia nsp-workflow repository opens in a new browser tab.

End of steps

9.7 How do I clone a workflow?

9.7.1 Purpose

Use this procedure to copy a workflow, for example, if you need a new workflow that is similar to an existing one but with a different value for an attribute., or if you want to test a change to a workflow.

Note: You can clone a signed workflow, but the clone will not be signed.

9.7.2 Steps

1

Open **Workflows**, **All Workflows**.

2

Choose a workflow and choose ⋮ (Table row actions), **Clone**.

3

In the form that opens, enter a name for the cloned workflow and click **CLONE**.

4

See 9.8 “How do I edit a workflow?” (p. 92) to edit the clone.

End of steps

9.8 How do I edit a workflow?

9.8.1 Purpose

Use this procedure to modify a workflow.

When you put a workflow into DRAFT status to edit it, it is locked for your edits. Other users can’t modify the workflow, or see the changes being made. When the workflow is PUBLISHED, it’s no longer locked.

If any schedules or executions are running at the time you put the workflow into Draft status, they are completed according to the last published version of the workflow. That is, the executions are not affected by your changes.

Workflows that are imported from GitHub and signed workflows cannot be edited. You can clone these workflows and make changes but the cloned versions are not signed.

You must have write privileges for the workflow you want to modify.

9.8.2 Steps

1

Open **Workflows**, **All Workflows**.

2

Open the Definition page of the workflow:

a. Click on a workflow and choose ⋮ (Table row actions), **Definition**.

b. Double-click on a workflow and choose **Definition** from the **Info** drop-down list.

3

Change the workflow state to Draft:

1. Click **Modify State**.
2. In the form that opens, choose Draft and click UPDATE.

4

In the **YAML (DRAFT)** tab, make changes as needed.

You can click on the **PUBLISHED** tab at any time to see the previous published YAML description.

5

Click **VALIDATE & UPDATE FLOW**.

If the validation fails, fix any problems and validate again.

6

When the validation succeeds, click **PUBLISH** to publish the edited workflow, or **SAVE DRAFT** to keep it in DRAFT status.

7

When you complete your edits and publish the workflow, check to see if any schedules or Kafka triggers are affected by your changes. For example, if you added a parameter to a workflow, any schedules that run the workflow must be updated to provide input for the new parameter.

END OF STEPS

9.9 How do I update an input form?

Tip: You can add or change the form input of a workflow in either DRAFT or PUBLISHED status.

9.9.1 Steps

1

Open **Workflows**, **All Workflows**.

2

Double-click on a workflow.

3

Choose **Input Form** from the **Info** drop-down list.

4

Choose a task from the following options.

a. To update the input form manually, enter JSON or YAML details in the editor and click **UPDATE INPUT FORM**.

b. To auto-generate JSON, click **AUTO GENERATE UI**. Click **OK** to confirm.

The input form is auto-generated. The auto-generated form overwrites any existing form input.

c. To import JSON, click **Import JSON from Filesystem** at the top of the screen. Navigate to the file you want and click **Open**.

The JSON details appear in a JSON form. Click **UPDATE INPUT FORM**.

The input form is updated in the workflow.

END OF STEPS

9.10 How do I change the status of a workflow?

Tip: When multiple workflows are selected, **Modify State** and **Export All** become available at the top of the screen.

9.10.1 Steps

1

Open **Workflows**, **All Workflows**.

2

Open the Update Workflow State form:

a. Click on a workflow and choose (Table row actions), **Modify Status**.

b. From the **Definition** page of a workflow, click **Modify State**.

3

Select the radio button for the new state.

4

Click **UPDATE**.

END OF STEPS

9.11 How do I execute a workflow?

9.11.1 Purpose

Use this procedure to run a workflow manually. Executing a workflow creates a workflow execution.

9.11.2 Steps

1

Open the Create Execution form:

a. From **Workflows**, **All Workflows**, choose a workflow and click ⋮ (Table row actions), **Execute** .

b. From **Workflows**, **Workflow Executions**, click + **EXECUTION**.

2

In the **Create Execution** form, verify that the environment is correct.

Click in the Environment field to change the environment if needed.

3

Enter a description if necessary.

The description appears in the **Workflows**, **Workflow Executions** view when the execution is created.

4

Enable the **Kafka Notification** check box as needed to enable Kafka notifications.

If Kafka notifications are enabled, when the workflow enters a new state (for example, Running, Success, or Waiting) a Kafka event is created on the WFM topic with details.

5

From the **Input Format** drop-down list, choose the format and provide inputs.

Note: File inputs cannot exceed 1 MB in size.

a. **Direct Input**

b. **File Input**

Click and choose the file to import. The Input Type is detected.

c. **URL Input**

Type or paste the URL and click Enter ↵.

The Input panel is populated.

6

Configure input parameters as needed.

7

Click **EXECUTE** to execute the workflow.

8

If an action in the workflow requires user input during execution, such as a confirmation message, the action enters Waiting On User Input status.

1. Choose **Executions**, **Workflow Executions** from the drop-down list at the top left of the page and double-click the workflow execution you created.
2. Click **WAITING ON INPUT ACTION EXECUTIONS**.
3. Choose the action execution and click ⋮ (Table row actions), **Update** . The user input form opens.
4. Configure the parameters.

The action execution status is updated and the workflow execution proceeds.

9

Return to the **Workflows**, **Executions**, **Workflow Executions** view to view the workflow execution status.

END OF STEPS

9.12 What is a flow?

9.12.1 Flow diagrams

A Flow diagram is a graphical representation of a workflow or workflow execution. Tasks are shown as boxes, and dependencies as lines between the boxes.

To open a Flow page, double-click on a workflow or workflow execution to open an information page. Choose Flow from the drop-down list, beside the workflow name.

When available, icons are shown on the task boxes to indicate the category of action the task performs:

- File Action : The task performs a file operation, such as delete or create.
- Folder Action : The task performs a folder operation, such as delete or create.
- Template Action : The task implements a Jinja2 template.
- Message Action : The task makes REST API calls or sends a message, including email or Kafka.

- Generic System Action
- CLI Action : The task uses CLI.
- User Action : The task requires user input.
- Javascfript Action : The task includes JavaScript.
- Python Action : The task includes Python.
- Checksum Action : The task returns the checksum value of a filename or path.
- Delay Action : The task incorporates a delay, for example, “perform this action, then wait two s”.
- Subworkflow : The task calls another workflow.

Click **Screenshot Flow** to download the Flow diagram as a .png file.

9.12.2 The Flow diagram of a workflow

The **Flow** diagram of a workflow shows all of the tasks that could be run by the workflow and their dependencies. For example: on success of Task 1, perform Task 2; on failure of Task 1, perform Task 3. For this example, the dependency is indicated by a solid line, colored green for success or red for failure. If the dependency is conditional, the diagram shows a dotted green line. Hover your mouse over the word **Condition** to see the condition.

Tasks are all in Preview status because the diagram is showing a workflow, not a completed execution.

Choose a task to see the path to the task; for example, the success of two previous tasks. With a task selected, click **Details** on the right of the page to open the Details panel and view information about the task, including the action performed, the task definition, and the information published by the action, if any.

9.12.3 The Flow diagram of a workflow execution

The Flow diagram of a workflow execution shows the tasks that ran for the chosen execution, including their status.

For detailed information about a task, choose the task and click **Details** on the right of the page to see the information panels.

The information panels show the following:

- **Details** : name of the task, the action performed, information published, and time stamps
- **Action Executions** : the list of action executions performed when the task ran, with their action IDs, status, input and output

  From the Action Executions panel, click **Debug** to test or debug an action.
- **Run Time**: the list of tasks performed in the workflow execution, the run time of each, and their timespan

9.13 How do I configure user access to a workflow?

9.13.1 Purpose

Use this procedure to grant or restrict access to a workflow for a user group. Before you can configure user access, user groups must be created; see the *NSP System Administrator Guide*.

This procedure requires an administrator role.

9.13.2 Steps

1

Log in to the NSP as an administrator.

2

Open **Workflows**, **All Workflows**.

3

Select one or more workflows.

Press and hold the Shift key to select multiple workflows.

4

Open the User Access form:

- If you have selected one workflow, click ⋮ (Table row actions), **User Access**
- If you have selected multiple workflows, click **User Access** at the top right of the page.

5

In the User Access form, configure access to the selected workflows:

1. Choose an action from the drop-down list at the top right of the form.
2. If you chose **Choose access**, configure the drop-down lists for each group in the panel at the right.

6

Click **SAVE**. The user access is updated.

END OF STEPS

10 Actions and action executions

10.1 Overview

10.1.1 What is an action?

An action is the smallest unit of instruction in a workflow. System actions are included with Mistral or downloaded to the NSP, and ad-hoc actions are customized by the user.

Actions can be installed using Artifacts; see 2.3 “How do I install an artifact bundle?” (p. 20).

For example, std.http is a system action: it sends an HTTP request. If this action is incorporated into a workflow, the user must specify the URL as part of executing the workflow. An ad-hoc action can be created that sends an HTTP request to a specific URL. A workflow that incorporated this ad-hoc action would not require the URL as part of the user input.

The Actions page shows the actions saved to NSP. Both system actions and ad-hoc actions are listed. You can create, edit, or delete an ad-hoc action. System actions are read-only.

Double-click on an action to open an Action Detail page, which provides a description of the action including parameters and usage examples.

10.1.2 What is an action execution?

An action execution is a record of an executed action. The Action Executions view shows the list of actions that were executed by workflows, the names of the tasks that initiated them, and their status.

Choose **Action Executions** from the drop-down list at the top left of the page to open the **Action Executions** view.

10.1.3 What can I do with actions or action executions?

Viewing the lists of actions and action executions can be useful to developers for troubleshooting and evaluating whether and how workflows might need to be modified.

Double-click on an action on the Actions page to open an Action Detail page. The Action Details page provides documentation about the action, including a description, identifying details, and, the YAML definition of the action.

See the Workflows tutorial on the Network Developer Portal for more information.

To evaluate a YAQL expression, click .* to open the Yaqulator.

10.2 How do I create an adhoc action?

10.2.1 Steps

1

Open **Workflows**, **Actions**.

2

Click **+ ADHOC ACTION**.

3

In the **Create Action** form that opens, enter, copy or paste your code in the YAML panel.

4

Click **VALIDATE**. Resolve any validation errors that appear.

5

Click **CREATE**.

END OF STEPS

10.3 How do I create an action execution?

10.3.1 Steps

1

Open one of the following:

a. **Workflows**, **Actions**

b. **Workflows**, **Action Executions**

2

Open the **Run Action Execution** form.

a. From **Workflows**, **Actions**, choose an action and click ⋮ (Table row actions), **Run**.

b. From **Workflows**, **Action Executions**, click **Run Action Execution**.

The Run Action Execution form opens.

3

Choose the Action Name and provide input as needed.

If an example is provided in the documentation for the action, you can select the example from the Select Example drop-down list. The example code is provided in the input field.

4

Click **RUN**.

The action executes.

END OF STEPS

11 Environments

11.1 What is an environment?

11.1.1 Environment

An environment file is a set of environment variables for use by workflows. When a workflow is executed, the environment variables are fetched and applied to the workflow where needed.

For example, you can set frequently used values such as hostnames as environment variables to reduce the amount of user input required by workflows.

See the Workflows tutorial on the Network Developer Portal for details about working with environments.

NSP is installed with a default environment, DefaultEnv. The default environment provides the information that NSP needs to reach the Git repository. Access to the NSP GitHub repository is configured by default. If you want the **Import from Git** button to point to a different location, update the DefaultEnv environment.

11.2 How do I create an environment?

Attention: Environment names must only contain alphanumeric characters.

11.2.1 Steps

1

Open **Workflows**, **Environments**.

2

Click **+ ENVIRONMENT**

3

In the **Create Environment** form that opens, enter a name in the Environment Name field.

4

a. If **Show Form** is on:
   1. Click **+ ADD**
   2. Enter the variable name and value and click **ADD**.
   3. Repeat for each variable you need to add.

b. If **Show Form** is off:
   1. Enter the variables in the panel.
   2. Click **VALIDATE** and resolve any validation errors.

5

Click **CREATE**.

END OF STEPS

11.3 How do I clone an environment?

11.3.1 Steps

1

Open **Workflows**, **Environments**.

2

Choose an environment and click ⋮ (Table row actions), **Clone**. NSP creates a copy of the environment called *environment_name*_CLONE and adds it to the list.

3

See 11.4 “How do I modify an environment?” (p. 103) to edit the clone.

END OF STEPS

11.4 How do I modify an environment?

11.4.1 Steps

1

Open **Workflows**, **Environments**.

2

Choose an environment and click ⋮ (Table row actions), **Update**.

3

a. If **Show Form** is on, click + **ADD** to add a variable, or select a variable and click ⋮ (Table row actions), **Edit**, or **Delete** to edit or remove one.

b. If **Show Form** is off, enter your updates in the panel and click **VALIDATE**. Resolve any validation errors.

4

Click **UPDATE**.

END OF STEPS

12 Jinja2 templates

12.1 What is a Jinja2 template?

12.1.1 Template engine

Jinja2 is a template engine for the Python programming language. A Jinja2 template is a structured file containing template tag blocks, or variables, and instructions for how to replace the variables with data. The template engine uses a Python script to read a YAML data file, insert the data into the template tag blocks, and generate the file. For example, Jinja2 templates can be used to convert a data set to a more reader-friendly format such as an HTML document.

Jinja2 templates can be created in NSP, imported, or installed using Artifacts; see 2.3 “How do I install an artifact bundle?” (p. 20).

See the Workflows tutorial on the Network Developer Portal for details about working with Jinja2 templates.

12.1.2 Sample template

The Jinja2 template below is a simple template that creates an HTML page.

```
name: myJinja
tags:
  - a
  - b
description: Sample Jinja2 Template
template: |
  <!doctype html>
  <title>{% block title %}{% endblock %}</title>
  <ul>
    {% for user in users %}
    <li><a href="{{ user.url }}">{{ user.username }}</a></li>
    {% endfor %}
  </ul>
  <p>
    Refer to https://jinja.palletsprojects.com/en/2.10.x/api/
  </p>
```

12.2 How do I create a Jinja2 template?

Attention: Jinja2 template names must only contain alphanumeric characters.

12.2.1 Steps

1

Open **Workflows**, **Jinja2**.

2

Click **+ JINJA TEMPLATE**.

3

In the **Create Jinja Template** form, enter a name for the template.

4

Add tags as needed:

1. Enter text in the **Tag** field and click **+**.
2. Repeat to add additional tags.

5

In the **Template** panel, enter or paste the template YAML.

6

Turn off the **Show Form** toggle at the top right of the form.

7

Click **VALIDATE** and resolve any validation errors that appear.

8

Click **CREATE**. The Jinja2 template is created.

END OF STEPS

12.3 How do I clone a Jinja2 template?

12.3.1 Steps

1

Open **Workflows**, **Jinja2**.

2

Choose a Jinja2 template and click ⋮ (Table row actions), **Clone**. NSP creates a copy of the environment called *template_name*_CLONE and adds it to the list.

3

See 12.4 “How do I edit a Jinja2 template?” (p. 106) to edit the clone.

END OF STEPS

12.4 How do I edit a Jinja2 template?

12.4.1 Steps

1

Open **Workflows**, **Jinja2**.

2

Select a Jinja2 template and click ⋮ (Table row actions), **Update** .

3

Enter your updates in the YAML panel and click **VALIDATE**. Resolve any validation errors.

4

Click **UPDATE**.

END OF STEPS

13 Workflow executions

13.1 What is a workflow execution?

13.1.1 Workflow execution

A workflow execution is an instance of an executed workflow.

In **Workflows**, **Workflow Executions**, you can see the list of executions that have been run since the last cleanup, including the inputs provided, their status (Idle, Running, Success, Error, or Canceled), their run time, and how they were triggered. The list can show you the status of scheduled and running workflows, and serve as a record of succeeded and failed executions.

The list shows the 1000 most recent executions by default. Statistics showing the status of workflows are based on the 4000 most recent executions.

Double-click on an execution to open an information page that shows the input entered, tasks that succeeded, any error messages that were produced, and a Flow diagram showing the tasks that executed.

13.1.2 Workflow tasks

A task defines a specific logical step in a workflow. Each task can take input data and produce output data.

A single task is a logical step that can perform an action; for example, performing a REST request using std.http or CLI request using std.ssh. A task can call other workflows. A task can also be iterated over a list of items performing bulk operations.

Tasks can be chained together based on the following criteria:

- on-success (runs only if the task ran successfully)
- on-error (runs only if the task failed)
- on-complete (always runs)

Tasks that are not chained are executed at the same time.

13.1.3 Execution workers

Execution workers are queues used by NSP to maximize the number of operations that can be executed at one time.

A workflow execution is automatically assigned one of the following workers, based on how the execution was initiated:

- default: executions initiated from the GUI or by an OSS
- trigger: executions initiated by a Kafka trigger or schedule
- LSO: executions initiated by large-scale operations performed by an NSP operator or Device Administrator APIs

Assigning executions to execution workers ensures that , for example, the execution of a large-scale operation will not prevent a scheduled execution from running on time.

Execution worker information is read-only.

13.1.4 Execution status

You can view the status of an execution from the list in **Workflows**, **Workflow Executions**. Select an execution and click ⋮ (Table row actions).

- Choose **Quick View** to view a summary of the execution, including input, output, and flow.
- Choose **Go to Tasks** to see the list of task executions.

The status of the execution can be any of the following:

- Idle: waiting to be run

  If the execution represents a child workflow and the parent workflow or task stops before the child runs, the child execution will remain in this state.
- Running
- Error: an error has been encountered by this execution or by a child workflow.

  If a child workflow is in an error state but the error is handled by the parent task, the parent workflow will not enter an error state.

  An error encountered by a related workflow execution will not cause another execution to enter an error state. For example, if Task 1 requires both Workflow 1 and Workflow 2 to succeed, and Workflow 2 encounters an error, Task 1 will fail, however, the status of Workflow 1 will be based only on the progress of that workflow.
- Success: completed successfully
- Failed
- Canceled: canceled by a user

13.2 How do I rerun a workflow execution?

13.2.1 Purpose

Rerun an execution to execute the workflow again with the same inputs.

13.2.2 Steps

1

To rerun a single execution manually:

1. Open **Workflows**, **Workflow Executions**.
2. Choose a workflow execution and click ⋮ (Table row actions), **Rerun** .

2

To automate running a workflow repeatedly with the same inputs, create a schedule.

End of steps

13.3 How do I update the description of a workflow execution?

13.3.1 Purpose

Use this procedure to add notes to the Description field of a workflow execution. You can filter on the Description field in the Executions list.

13.3.2 Steps

1

Open **Workflows**, **Workflow Executions**.

2

Choose a workflow execution and click ⋮ (Table row actions), **Update** .

3

In the form that opens, make changes as needed.

4

Click **UPDATE**.

END OF STEPS

13.4 How do I stop a workflow execution?

13.4.1 Purpose

You can cancel a workflow execution manually, or the execution can be canceled by an API action.

Note:

- When a workflow execution is canceled, no new actions are executed. Actions that are already running at the time of cancellation will complete.
- If a workflow contains a child workflow, canceling either the child or the parent will cancel the execution of both workflows.

13.4.2 Steps

1

Open **Workflows**, **Workflow Executions**.

2

Select the execution you need to cancel and click ⋮ (Table row actions), **Cancel**.

END OF STEPS

13.5 How do I change the state of a running or failed workflow execution?

13.5.1 Steps

1

Open **Workflows**, **Workflow Executions**.

2

Choose a workflow execution and click ⋮ (Table row actions), **Modify State** .

3

Choose the new state and click **UPDATE** to confirm.

END OF STEPS

13.6 How do I evaluate a YAQL expression?

13.6.1 Steps

1

Open **Workflows**, **Workflow Executions**.

2

Double-click on a workflow execution to open the information page.

3

Choose **Flow** from the drop-down list beside the workflow name.

4

From the Flow page, click **Yaqulator** .*.

5

In the **Yaqulator** form that opens, enter or paste your YAQL expression in the YAQL Expression field.

6

Enter or paste the YAML or JSON context for the YAQL expression to evaluate in the YAML/JSON Context field.

Verify that the radio button for the correct format is selected.

7

Click **EVALUATE**.

The Result field is populated with the results of your query.

END OF STEPS

13.7 How do I debug an action?

13.7.1 Steps

1

Open the information page for a workflow or a workflow execution:

a. Open **Workflows**, **All Workflows** and double-click on a workflow.

b. Open **Workflows**, **Workflow Executions** and double-click on a workflow execution.

2

Choose **Flow** from the drop-down list beside the workflow name.

3

From the Flow page, choose an action execution:

1. Choose a task and click **Action Executions** at the right of the screen to open the Action Executions panel.
2. Select an action execution from the list to show the execution information.

4

Click **DEBUG**.

5

In the **Run Action Execution** form that opens, update the contents as needed.

6

Click **RUN** to run the action with the updated details.

7

Click **RESULT** to verify the results of the updated action.

8

As needed, click **COPY** to copy the updated action to the clipboard, and modify the workflow to paste in the copied details.

END OF STEPS

14 Kafka triggers

14.1 What is a Kafka trigger?

14.1.1 Kafka trigger

A Kafka trigger is a method of event-driven automation. When a Kafka trigger is configured, a workflow is executed in response to a Kafka event notification. For example, if a Kafka notification is received indicating that a port is down, a workflow execution can be triggered to bring up another port and move the services from the down port to the new port.

A Kafka trigger consists of a Kafka topic and event type to monitor, and a trigger rule. The trigger rule is a statement in JSONPath syntax that declares the matching criteria for the workflow to run.

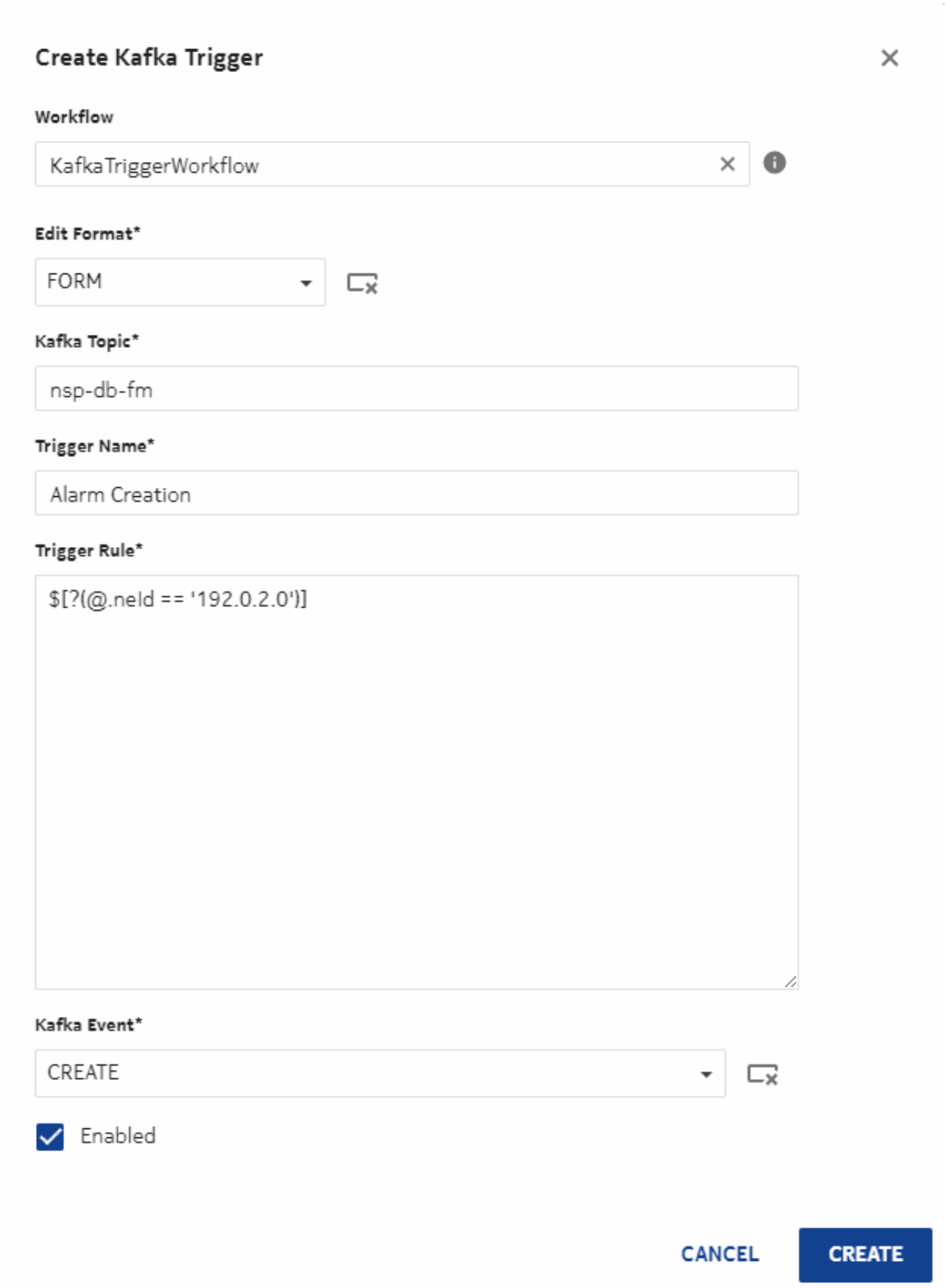
Restrictions apply on topics that can be used for triggering; see the Workflow Manager APIs tutorial on the Network Developer Portal.

By default, the Kafka trigger is limited to 10 executions per minute. This can be changed by updating the Kafka trigger.

See the Network Automation tutorial on the Network Developer Portal for details about working with Kafka triggers.

14.1.2 Sample Kafka trigger

The Kafka trigger below is configured to execute the WFM_example workflow if an alarm creation event is received on the nsp-db-fm (fault management) Kafka topic for a specified NE.



14.2 How do I create a Kafka trigger?

14.2.1 Before you begin

Before a Kafka trigger can be created to execute a workflow, the KafkaTrigger tag must be applied to the workflow definition.

Attention: Kafka trigger names must only contain alphanumeric characters.

14.2.2 Steps

1

Open **Workflows**, **Kafka Triggers**.

2

Click **KAFKA TRIGGER**.

3

In the **Create Kafka Trigger** form that opens, configure the parameters.

4

Click **CREATE**.

END OF STEPS

14.3 How do I edit a Kafka trigger?

14.3.1 Steps

1

To update the Kafka trigger definition:

1. Open **Workflows**, **Kafka Triggers**.
2. Choose a Kafka trigger and click (Table row actions), **Update** .
3. In the Update Kafka Trigger page, make changes as needed.
4. Click **UPDATE**.

2

To reset the Times Matched and Times Executed counters, select a Kafka trigger and click **Reset Counters** .

END OF STEPS

15 Schedules

15.1 What is a schedule?

15.1.1 Schedule

A schedule allows you to configure a workflow to execute in the future, either once or repeatedly.

When you create a schedule you will specify the input and parameters. If you want the input to change from one execution to another, you will need multiple schedules. For example, if you would like a cleanup workflow to clean up failed workflow executions once a week and all workflow executions once a month, create a weekly schedule and a monthly schedule.

15.2 How do I schedule a workflow?

Attention: Schedule names must only contain alphanumeric characters.

15.2.1 Steps

1

Open the **Create Schedule** form:

a. Open **Workflows**, **All Workflows** view, choose a workflow and click ⋮ (Table row actions), **Schedule**. The Create Schedule form opens for the workflow.

b. Open **Workflows**, **Schedules**, and click + **SCHEDULE**. The Create Schedule form opens with no workflow specified.

2

In the **CONFIGURATION** tab, configure the general parameters.

1. Select a workflow and environment as needed.
2. Enter a schedule name.
3. Update the information in the **Input and Parameters** area.

   When the schedule runs, it executes the workflow with the input you enter here.
4. For NSP to notify Kafka when the schedule runs the workflow, select Kafka Notification.

3

In the **SCHEDULE** tab, configure the number of executions and the scheduling details.

1. Configure the Start Time parameter. If you only want the schedule to execute once, this is the only parameter you need to configure.
2. Configure repetition if needed:

- To set a simple repetition pattern, such as once a week, choose Repeat and configure the parameters.
- To set a custom repetition, such as specific days of the month, choose **Advance Mode** and configure the parameters.

4

Click **CREATE**.

END OF STEPS

16 Workflow policies

16.1 What is a workflow policy?

16.1.1 Cleanup

A policy allows you to configure a workflow execution cleanup operation to execute on a perpetual schedule. You can configure a simple schedule, or create a custom schedule using a cron expression.

Choose **Policies** from the drop-down list at the top left of the page to open the **Policies** view.

When you create a policy you will specify the schedule, age of executions to clean up, and filters. You can filter based on tags and/or execution state. For example, you can configure one policy to delete failed workflow executions weekly, and another to delete successful workflow executions with the `testing` tag daily.

The default policy is pre-loaded with installation of NSP. By default, executions are retained for 30 days.

You can also manage cleanup policies using the Workflow Manager API; see the Network Automation tutorial on the Network Developer Portal.

16.2 How do I create a workflow execution cleanup policy?

16.2.1 Steps

1

Open **Workflows**, **Policies**.

2

Click **+ POLICY**. The Create Policy form opens.

3

Enter a policy name.

4

Define the workflow executions to be cleaned up by the policy:

1. Update the Older Than (Days) parameter as needed. By default, executions are retained for 30 days.
2. Select a tag to filter on if needed.
3. Select an execution state.

For example, if you enter 30, test, and SUCCESS, when the policy executes it will delete successful executions with the `test` tag that are older than 30 days.

5

Configure the schedule:

a. Click **Schedule** and configure the schedule parameters.

b. Click **Custom** and configure a cron expression in the format *minute hour day-of-month month day-of-week*.

Cron expression wildcards are supported. For example, the cron expression `0 0 1 * ?` indicates midnight on the first day of the month.

Note: Nokia recommends that schedules not be set to run more than once a day.

6

Click **CREATE**.

END OF STEPS

16.3 How do I edit a policy?

16.3.1 Steps

1

Open **Workflows**, **Policies**.

2

Choose a policy and choose ⋮ (Table row actions), **Update**. The Update Policy form opens.

3

Edit the parameters as needed and click **UPDATE**.

END OF STEPS

Part VI: Network Automation use cases

Overview

Purpose

Describes use cases for Network Automation.

Contents

17 Network intent use cases

17.1 Creating an intent

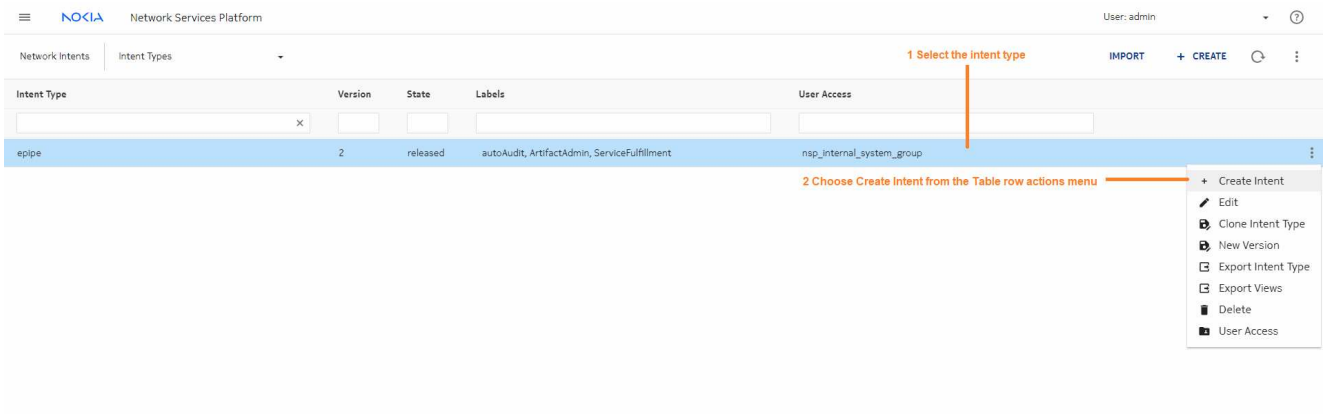
17.1.1 Purpose

This use case shows how to create an intent from an existing intent type.

17.1.2 Steps

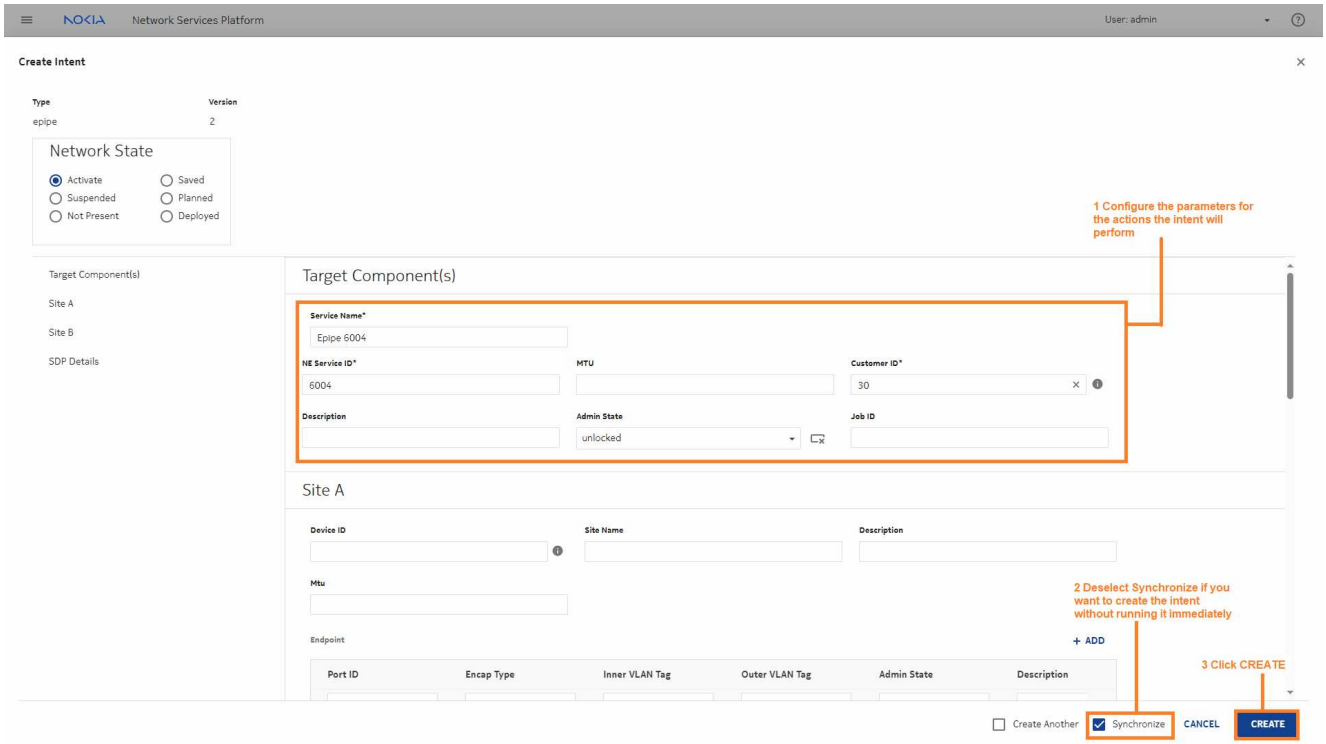
1

From the **Network Intents**, **Intent Types** view, select the intent type and click Create Intent.



2

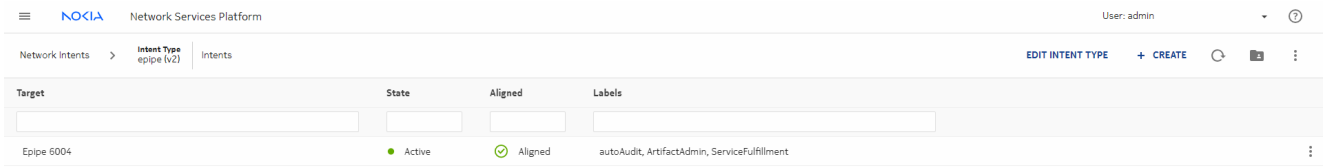
In the **Create Intent** form, configure the parameters and click Create. Choosing **Synchronize** will execute the intent immediately.



END OF STEPS

Result

Double click on the intent type to view the list of intents. The Aligned field shows that the intent parameters are aligned with the network: the intent has executed successfully.



18 Workflow use cases

18.1 Importing and executing a workflow from Git to NSP

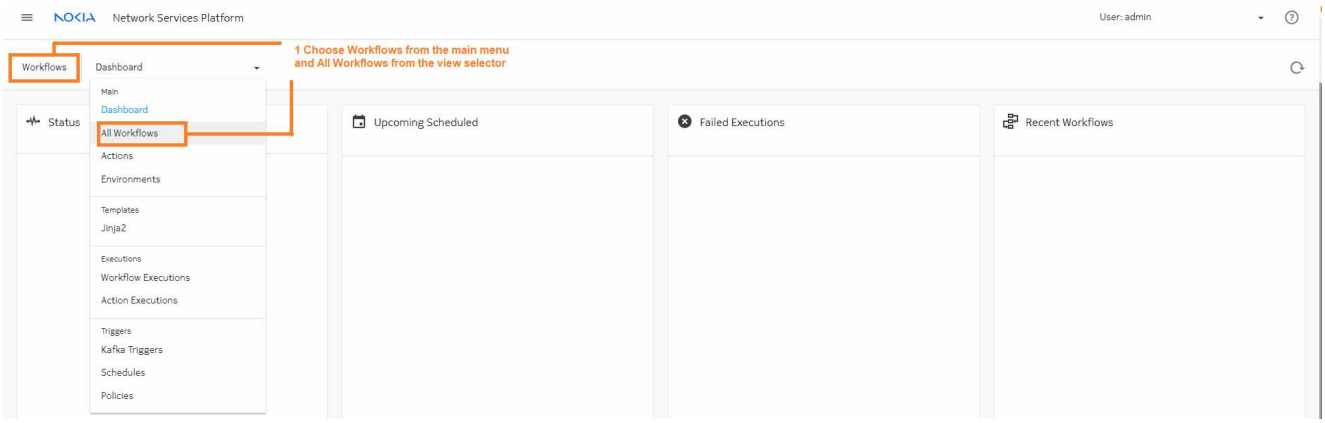
18.1.1 Purpose

This article shows how to import a workflow from the Nokia GitHub repository, publish and execute the workflow.

18.1.2 Steps

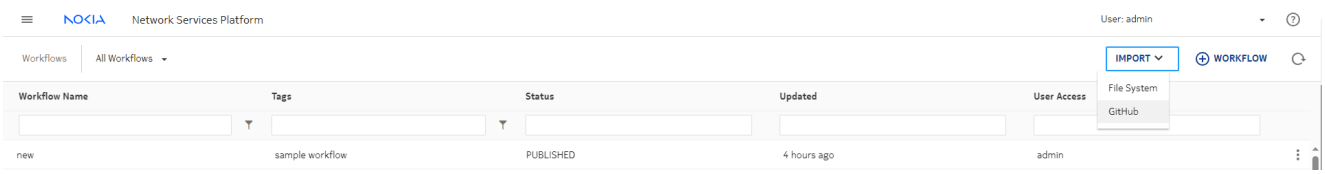
1

Open **Workflows**, **All Workflows**.



2

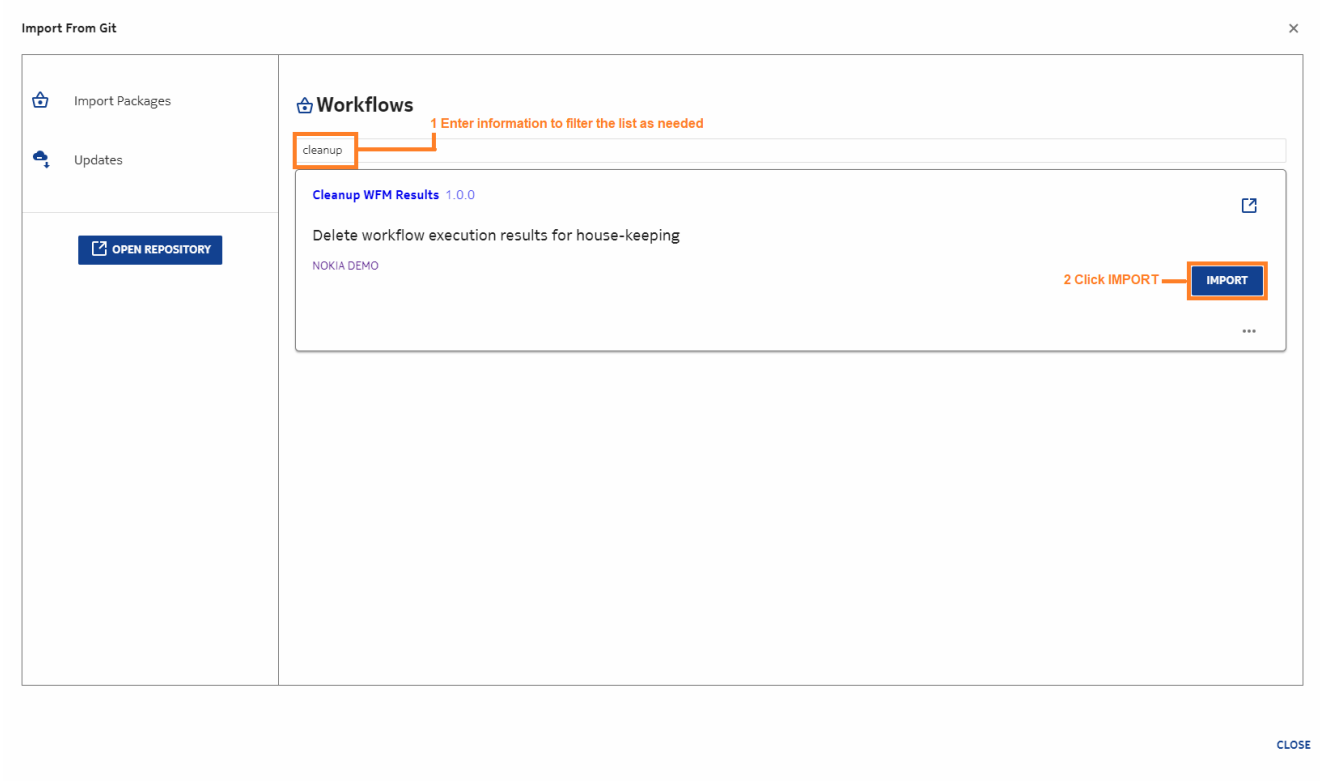
At the top of the view, choose **IMPORT**, **GitHub**.

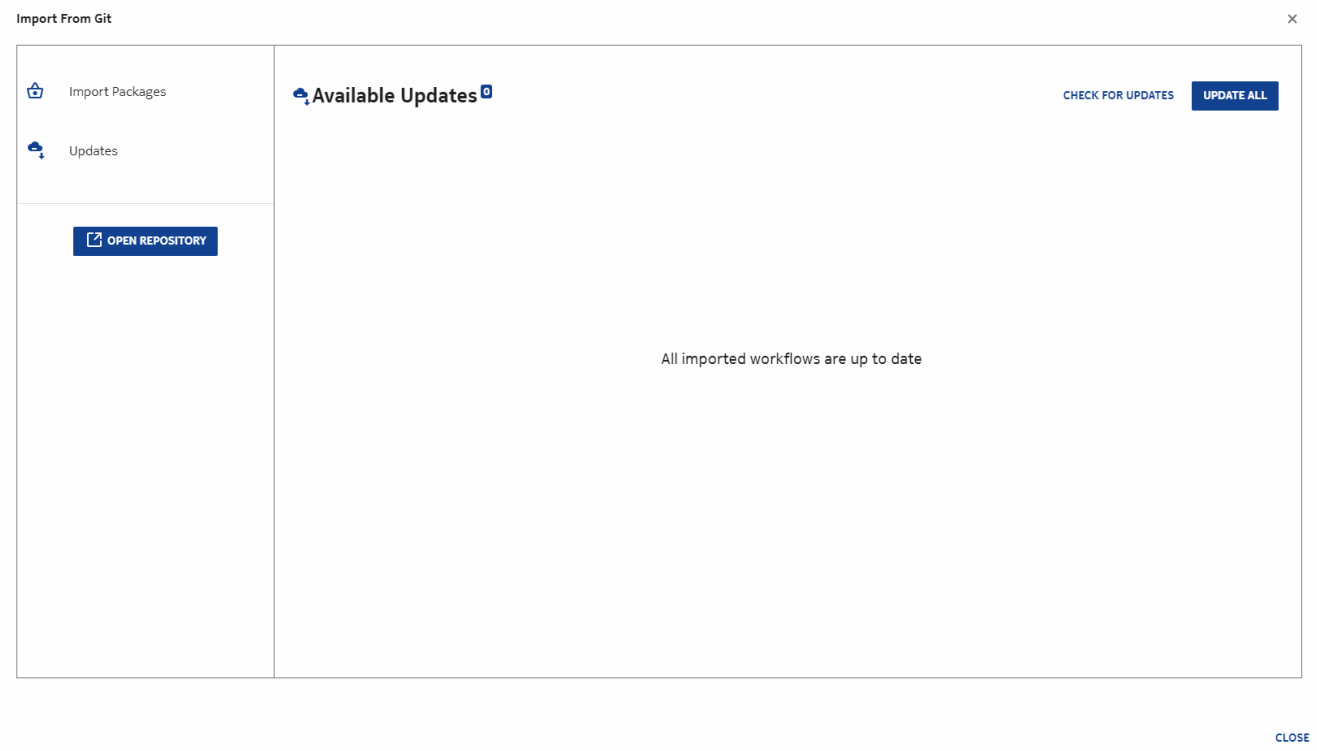


3

The Import from Git form shows the list of available workflows in the Nokia NSP repository. We'll import Cleanup WFM Results. Use the filter field to filter the workflows and click Import.

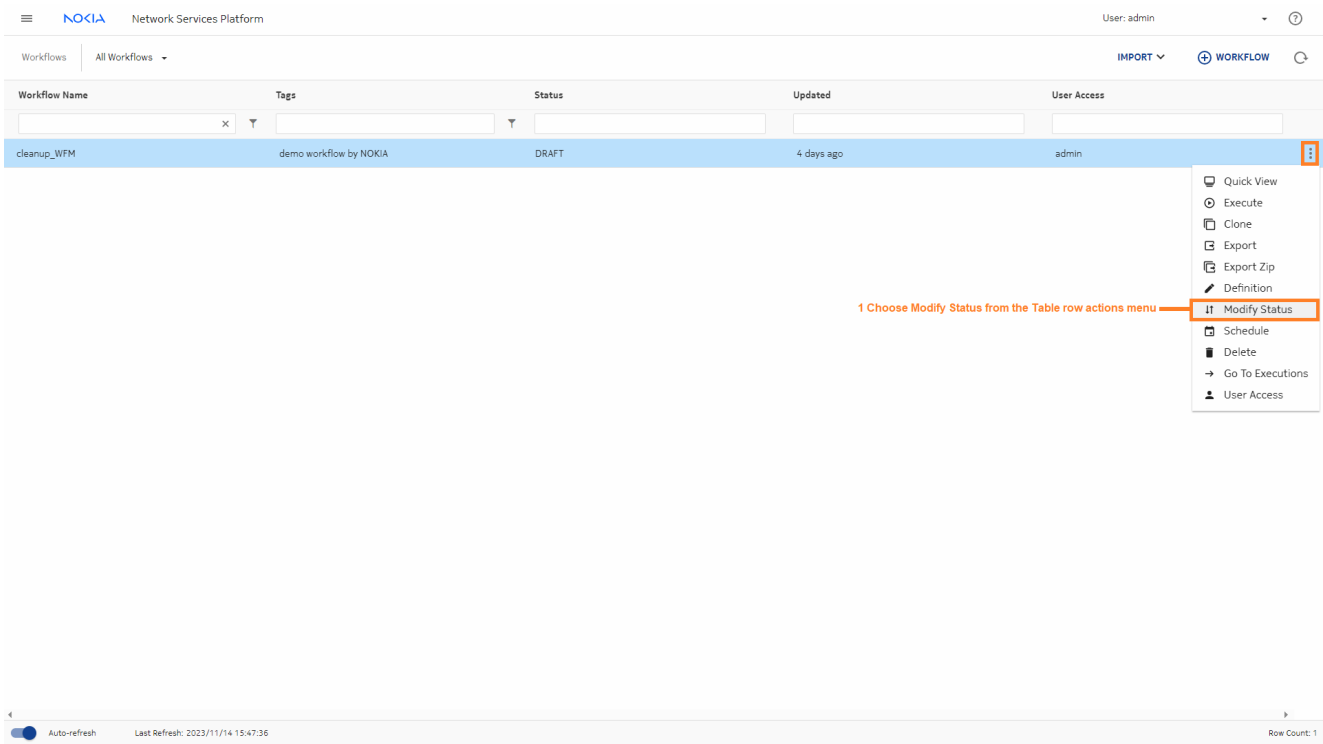
Click **Updates** to verify that the imported workflow is up to date.

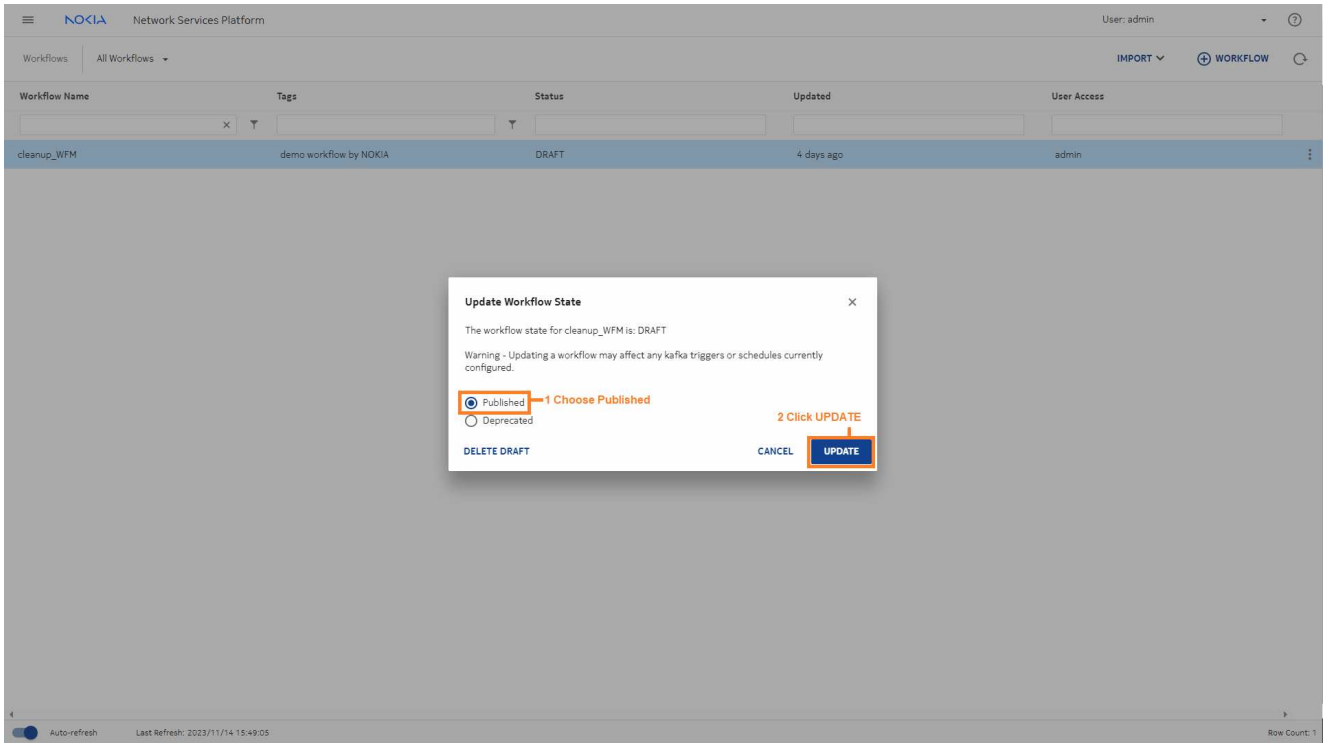




4

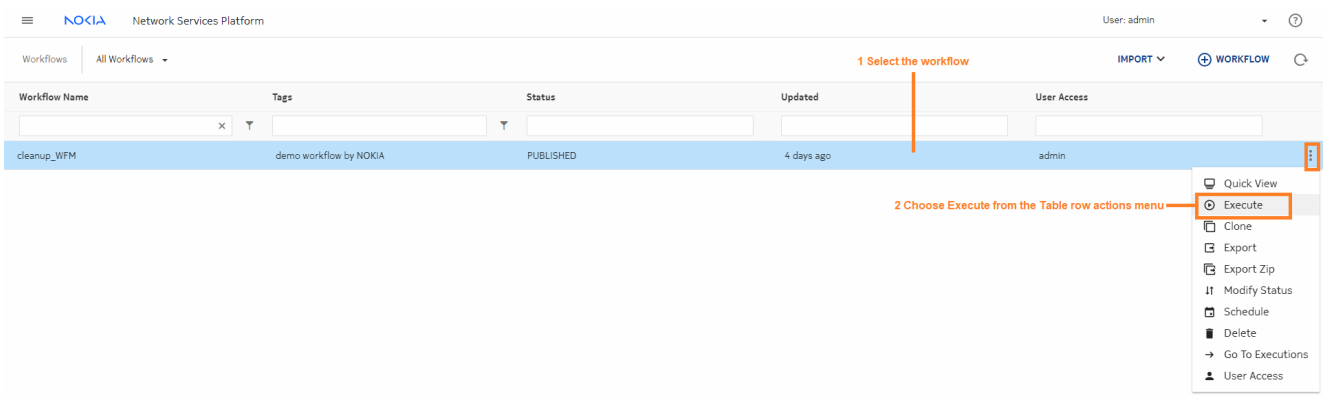
The imported workflow appears in the **Workflows**, **All Workflows** view, with a status of DRAFT. To execute the workflow, we'll need to change the status to PUBLISHED.

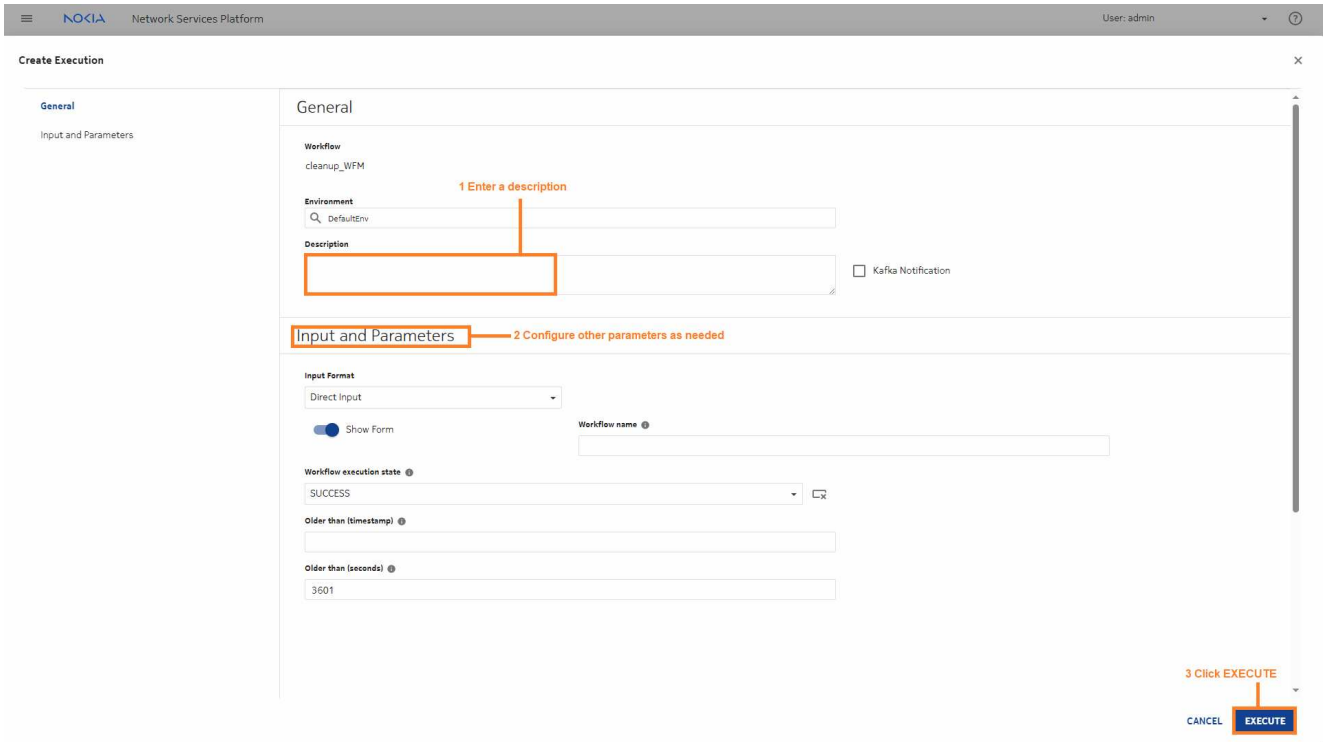




5

Now we can execute the workflow:





END OF STEPS

Result

Let's switch to the **Workflows**, **Workflow Executions** view to view the status. The green checkmark indicates that the workflow has executed successfully.