HP NFV Director

HP NFV Director

Version 1.0

User Guide

Edition: 1.1

For Linux (RHEL 6.4) Operating System

Legal Notices

Warranty

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

License Requirement and U.S. Government Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

Trademark Notices

Adobe®, Acrobat® and PostScript® are trademarks of Adobe Systems Incorporated.

HP-UX Release 10.20 and later and HP-UX Release 11.00 and later (in both 32 and 64-bit configurations) on all HP 9000 computers are Open Group UNIX 95 branded products.

Java™ is a trademark of Oracle and/or its affiliates.

Microsoft®, Internet Explorer, Windows®, Windows Server 2007®, Windows XP®, and Windows 7® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Firefox® is a registered trademark of the Mozilla Foundation.

Google Chrome® is a trademark of Google Inc.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

EnterpriseDB® is a registered trademark of EnterpriseDB.

Postgres Plus® Advanced Server is a registered U.S. trademark of EnterpriseDB.

UNIX® is a registered trademark of The Open Group.

X/Open® is a registered trademark, and the X device is a trademark of X/Open Company Ltd. in the UK and other countries.

Red Hat® is a registered trademark of the Red Hat Company.

Linux® is a registered trademark of Linus Torvalds in the U.S. and other countries.

Neo4j is a trademark of Neo Technology.

Contents

Tables

Figures

Chapter 1

In this Guide

This guide describes the NFV Director for users of the solutions that are based on the product, operators focused on the NFV Director processes as well as system administrators.

Chapter 1 and 2 is a general overview of the NFV Director for the audience of this guide. Rest of the manual focuses on the operational and troubleshooting aspects of NFV Director.

Chapter 3 provides the overview of the data modeling of the NFV Director and how the relationships are defined between the various model objects.

Chapter 4 describes the various data modeling operations that need to be performed using the NFV Director Inventory. This chapter explains how the NFV Director data model can be created, viewed, edited, deleted, uploaded, and downloaded.

Chapter 5 describes the South Bound Interfaces implemented to perform the NFV Director operations on VIMS and Hypervisors.

Chapter 6 describes the North Bound Interfaces available in order to send a request for creation of Virtual Machines (VM) and Virtual Network Functions (VNF), Scale in and out operations.

Chapter 7, 8 and 9 describes the Virtual Network Functions, Policies applied on the VNFs and VNF Scaling operations.

Chapter 10 and 11 describes in detail the activation processes and workflows.

Chapter 12 describes the administration of various NFV Director components in the solution.

Chapter 13 describes the VNF monitoring aspects of the NFV Directory. It involves configuring and administrating the HP SiteScope product to achieve the VNF monitoring.

Chapter 14 describes the NFV Director process on the alerts triggered by the KPI breach on the VMs and VNFs, including the correlation and the action taken to remedy the root cause of the alert.

Chapter 15 describes the various troubleshooting instructions that may be useful to refer if the user of the NFV Director runs into issues.

Note

Read this document before installing or using this software.

1.1 Audience

This is the only manual in the document set for NFV Director intended for the solution users and system administrators. All the other manuals are intended for system

integrators who build solutions based on the NFV Director core product. However, it is recommended for any user of the NFV Director to begin with this document before proceeding with installation or any other aspects of the solution.

1.2 Typographical Conventions

Courier Font:

- Source code and examples of file contents.

- Commands that you enter on the screen.

- Pathnames

- Keyboard key names

Italic Text:

- Filenames, programs and parameters.

- The names of other documents referenced in this manual.

Bold Text:

To introduce new terms and to emphasize important words.

1.2.1 Install Location Descriptors

Following names are used to define the install locations throughout this guide.

Descriptor	What the Descriptor represents
${OM_INSTANCE}	`/var/opt/openmediation-V62/containers/<instance-#>`
${UCA_AUTOMATION_CONSOLE_HOME}	This directory contains the UCA Automation UI deployment. The path refers to `/opt/UCA-ATM`
${UCA_EBC_HOME}	The root directory of UCA-EBC. The default value is `/opt/UCA-EBC`
${UCA_EBC_INSTANCES}	This directory may contain multiple instances of UCA-EBC where the value packs are deployed. The path refers to `${UCA_EBC_DATA}/instances/default`
${ACTIVATOR_OPT}	The base install of Service Activator is `/opt/OV/ServiceActivator`
${SITESCOPE_HOME}	`/opt/HP/SiteScope`

Table 1 Install Location descriptors

Chapter 2

Introduction

HP Network Functions Virtualization (NFV) Director is an ETSI MANO compliant NFV orchestrator that is responsible for lifecycle management of network services (NS) across the entire operator's domain, like multiple datacenters. The NFV Director performs the following functions:

- Manages network services.

- Manages NS and VNF packages (functions such as on-boarding new NS and VNF packages).

- Creates new NS during run-time.

- Lifecycle management of NS instantiation and NS instance—Includes functions such as update, scale-out/in, event collection and correlation, and termination.

- Manages integrity and visibility of NS instances through their lifecycle and also manages the relationship between the NS instances and the VNF instances.

- Manages NS instance topology across the entire operator domain.

- Monitors NS instances.

- Manages NS instance automation across the entire Operator domain.

- Manages policies and enforces them for the NS instances.

2.1 HP OpenNFV Program

The NFV Director is part of the HP OpenNFV Program, operationalizing NFV through an NFV orchestrator with embedded VNF manager capabilities.
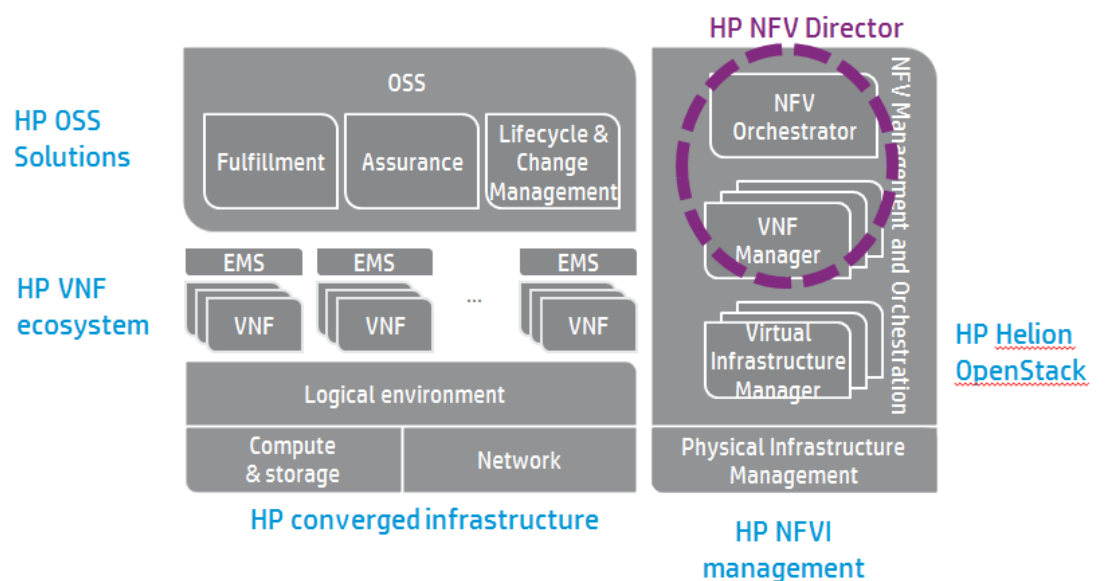
Figure 1: HP NFV Director

2.2 HP NFV Director Functionality

At a high level the HP NFV Director performs the following functions:

- Orchestrates the allocation and release of resources to be used by VNF.

- Applies policies for orchestration and resource allocation.

- Installs and instantiates VNF software images on virtual and physical machines.

- Configures virtual or physical networks to connect instantiated machines.

- Assumes the role of a VNF Manager in the absence of a VNF Manager for a VNF.

- Monitors and accounts NFVI resources for a specific VNF and network service.

- Checks the usage of a resource against policies.

- Collects performance data for NFVI compute, storage, and network resources.

- Detects and handles faults in NFVI.

- Provides the ability to consume fault data for a VNF and fault management.

- Proactively monitors NFVI resources and generates fault in the absence of fault reporting from NFVI.

- Interfaces with the VNF Manager to provide NFVI resources to be consumed by VNF managers for VNFs.

- Provides a VNF and NS Catalog for the operator to instantiate and manage VNF and NS.

- Provides a framework for fault correlation and root-cause analysis process to determine the reason for fault conditions and their impact on VNFs and network services.

- Provides a framework to determine corrective actions. It triggers such actions at one or more action points within the NFV framework or to the OSS.

- Defines and provides an interface to external entities (such as OSS and NMS) for the following functions:

    o VNF on-boarding

    o Lifecycle management of VNF instances (in co-ordination with VNF Managers)

    o NFV Policy Management

    o Performance Data of VNF and NS

    o NFVI resource usage accounting

For features supported in version 1.0, refer to the *NFV Director Release Notes*.
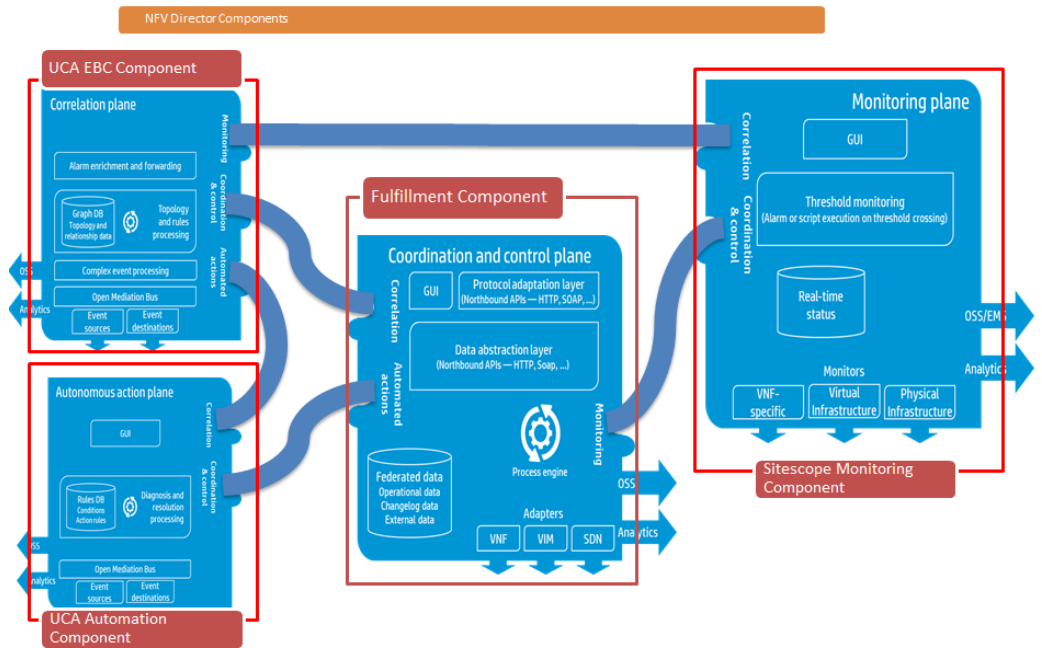
Figure 2: HP NFV Director Component Architecture

NFV Director consists of the following components:

- Co-ordination and Control plane
- Monitoring plane
- Correlation plane
- Autonomous action plan

2.2.1 Co-ordination and Control plane

The NFV Director fulfillment is based on four components as described in the following table.

Interfaces	GUI	Data model	Automation
Northbound Generic (Web service) OpenStack (REST) Custom (project based through Protocol adaptors)	HPSA standard JSPs	HPSA Standard Beans	HPSA standard workflows
Southbound HPSA plug-in (using Templates)	HPSA XPMAPS Dynamic forms	Generic Artifact/Relationships Beans	GPM Process

Table 2: Fulfillment Components

It is based on the HPSA platform.

2.2.2 Monitoring plane

The monitoring plane consists of three parts:

- An integration component called Assurance Gateway

- Monitoring that is agentless

- Mediation to collect events from external sources

2.2.2.1 Assurance Gateway

The Assurance Gateway receives and processes the VNF topology information from fulfillment, and updates UCA-EBC graph database for correlation.

It also receives and processes monitoring notifications from fulfillment and delegates appropriate action with all the data to agentless monitoring component.

2.2.2.2 Monitoring

The NFV Director includes the agentless monitoring component. Built using HP SiteScope, this component can monitor a wide variety of monitoring points, issuing events or executing commands when pre-defined thresholds are crossed. As different VNFs have different monitoring needs, the monitoring points and thresholds are automatically configured by NFV Director as the VNF is provisioned or modified. As an agentless solution, the NFV Director does not require installation of monitoring agents on the target systems.

The individual VNF managers are responsible for monitoring their own internal faults and performance, possibly augmented with traps and KPIs provided by the NFV Director. When VNF management functionality is provided out of the NFV Director, through an embedded VNF manager, it might be necessary to collect application-specific monitoring information. HP Sitescope provides the capability to develop custom monitors for VNF resources and VNF applications.

SiteScope templates are used to standardize a set of monitor types and configurations into a single structure. This structure can then be repeatedly deployed as a group of monitors targeting multiple elements of the monitored environments.

Templates accelerate the deployment of monitors across the enterprise through a single-operation deployment of groups, monitors, alerts, remote servers, and configuration settings.

2.2.2.3 Mediation

HP NFV Director includes HP Open Mediation, which provides the following capabilities:

- Allows the integration of multiple products.

- Defines common communication patterns:

    o Alarm flow

    o Resynchronization

    o Action invocation

    o Topology notification

- Provides numerous connectivity features:

    o Web Services—SOAP/HTML, SOAP/JMS, HTML/REST

    o Files—Local file access, FTP/FTPS/SFTP

    o Database—JDBC

    o Enterprise Java—JMS , JMX , RMI

         o   Other—TCP/UDP, HTTP/HTTPS, IRC, LDAP, SMTP/POP3/IMAP, RSS, SMPP, SNMP, XMPP

In the context of the NFV Director, OpenMediation is used to integrate with Sitescope, EMS, VNF Manager, or any other source of events.

2.2.3 Correlation plane

The Unified Correlation Analyzer for Event Based Correlation product (also known as UCA Expert by analogy with the legacy TeMIP Expert software) offers a new and generalized event based correlation solution.

Based on the JBoss Drools 5.5.0.Final, rule engine, UCA for EBC offers the capability to create comprehensive functional correlation sets called Value packs that implement the correlation logic. This correlation is performed by running certain rules (the rules are written in a Java-based language). Any Value Pack can support or use predefined functionalities such as Alarm collection, filtering, lifecycle, as well as Generic Actions.

The UCA for EBC can perform the following functions:

- Collect alarms and map them into Operator Alarm model (Alarm).

  The Alarm model is based on a mix of X733 and OSS/J Fault Management Model.

- Run several scenarios (rule engines) in parallel and in sequence to implement complex correlation algorithms. Each set of scenarios implementing a single correlation solution is grouped inside a UCA for EBC Value Pack.

- Dispatch Alarm objects for different scenarios.

- Execute rules based on the scenario input stream and generate suitable output. For example, actions to external systems.

- Control the scenario input stream using an alarm-based filtering layer.

- Execute actions such as storing in a database, creating a Trouble Ticket, creating a new Alarm, group alarms, forward an alarm to another scenario, or execute a Generic Action through the OSS Open Mediation v6.2 (NOM v6.2) layer.

Rule files are JBoss Rule files. Hence, both JBoss Expert and Fusion rules are supported in the UCA for EBC rule files. JBoss Drools Expert and JBoss Drools Fusion are JBoss Drools basic modules.

Over this basic functionality, UCA for EBC also provides a software development kit (SDK) that allows solution developers to easily build UCA for EBC Value Packs (Functional Correlation block). Administration tools (both command line and a GUI) are also available to manage, monitor, and troubleshoot the product.

UCA for EBC can be connected with a mediation bus (OSS Open Mediation v6.2), providing the capability to collect alarms from any source (NMS) and performing actions in return.

2.2.4 Autonomous Action plane

UCA Automation software, which is a combination of both business rule engine and workflow engine, enables a clear separation of what to automate and how to automate. All complexities of automation, such as how to access a network resource (can be a network element, an element component, an EMS, or NMS), what its credentials are, which specific transport mechanism should be used to connect to the resource, what specific OS versions of the device should be supported, what specific commands need to be sent, are abstracted from the business rules.

This software enables administrators to create, update, and read the business rules with utmost clarity and to maintain them efficiently. It also empowers the administrators to store the knowledge gained regarding the automation in the form of business rules focusing on what part without bothering about the how part.

Another advantage of UCA Automation software is that for most of the resolution automations, the operator should know only the business rules and not the business rules technologies to implement day-to-day operational changes to the decisions.

Thus, UCA Automation System is a platform for building value added resolution automations based on a judicious combination of business rules and workflows.

<div align="right">

Chapter 3

</div>

Overview of NFV Director

This chapter provides an overview of the architecture of the NFV Director.

3.1 Four pillars of NFV Director

NFV Director is based on four basic pillars that support all features within the product.
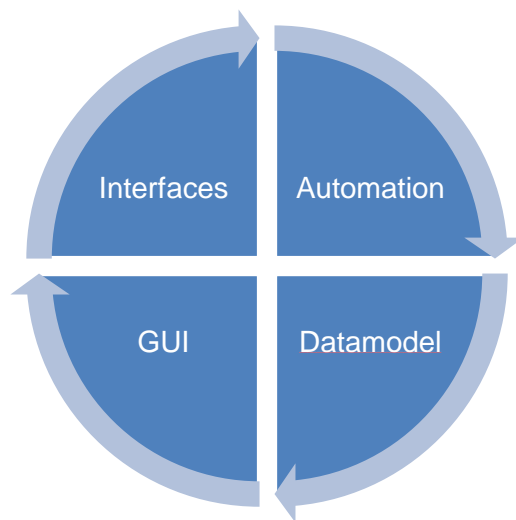
Figure 3: Four pillars of NFV-D

3.2 Interfaces

The NFV Director provides the following two types of interfaces:

- Northbound interfaces
- Southbound interfaces

3.2.1 Northbound interfaces

The Northbound interfaces of NFV director are based on a module called SOSA (Service Order Smart Adapter) that allows to map (Smart Adapter) any request sent to the NFV Director to the internal operations defined on the catalog of the product (Service Orders).

Any request can be queued allowing to protect the system from errors and to set appropriate priority for each request based on their origin or type of operation.
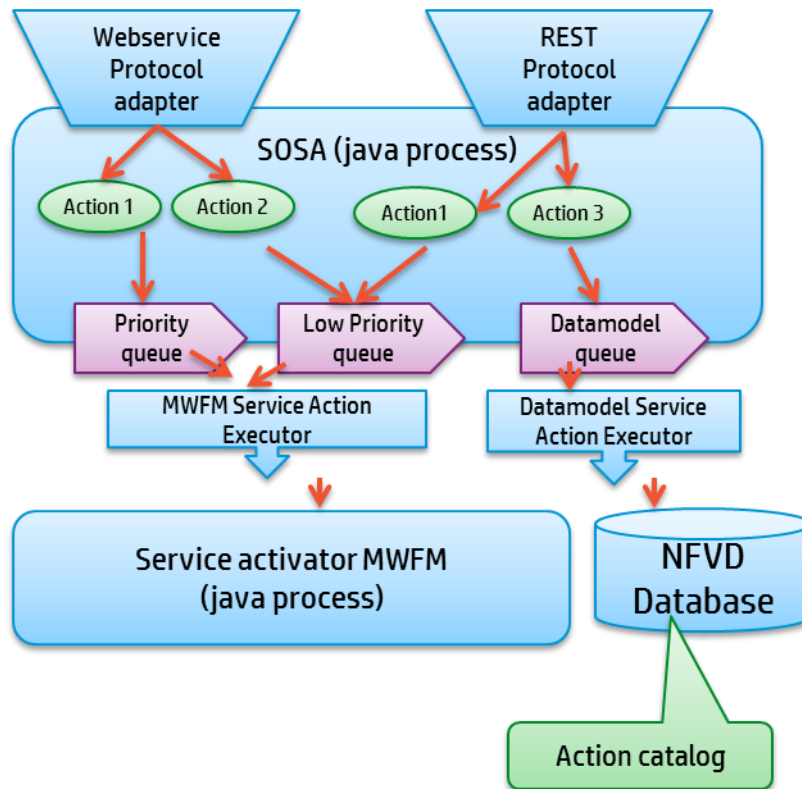
Figure 4: SOSA Mapping behavior

This mapping function allows to expose the same operations (for example, create VNF) using different protocols or syntaxes (for example using REST API and Web services).

The product provides out-of-the-box generic web service interface that allows calling any operation in the catalog using the dynamic service orders.

Each service order is basically identified using three parameters--Name, Type, and Action. For example:

- NFVD | CREATE | VNF (for creating a VNF)

- NFVD | DELETE| VNF(for deleting a VNF)

- NFVD | SCALE_IN| VM (for scale in a VNF)

- NFVD | SCALE_OUT| VM (for scale out a VNF)

Each service order might be required as a set of characteristics or parameters for running it successfully.

```
<dyn:characteristics>
<!--1 or more repetitions:-->
    <dyn:characteristic>
    <dyn:name>?</dyn:name>
    <dyn:value>?</dyn:value>
  </dyn:characteristic>
 </dyn:characteristics>
```

For more information on SOSA module, refer to the SOSA related documentation.

3.2.2 Southbound interfaces

The NFV Director can connect to any system using an extensible engine that uses plug-ins.

Each plug-in allows to connect in a certain way (web services, SSH, Telnet, REST, and so on) and execute a set of instructions/commands.

The instructions/commands are stored in the database in the form of templates and allow variables for providing the reusability of those templates.

Each template can be organized in sections and can optionally have rollback instructions for each section.

3.3 Data model

The NFV Director data model is a technology based on XML objects and allows modeling any kind of entity and any kind of relationship between those entities.

For more details, see the *NFV Director Data model Overview* section.

3.4 GUI

NFVD provides an administration portal and an operational portal.

The operational portal is called Solution Container and allows you to deploy new solutions on the portal to enrich the out-of-the-box functionality of the product.

Also, the product provides a new HTML5 display technology called Xmap that allows representing network diagram artifact models and huge amount of visualization to improve the GUI experience.

3.5 Automation

NFVD automation is based on HP Service Activator workflow engine. For more details of the workflow engine, refer to the *HP Service Activator Workflows and the Workflow Manager Guide*.

Chapter 4

Overview of NFV Director Data Model

4.1 NFV Director modeling philosophy

The NFV Director does not follow the regular database modeling rules. Instead, it uses artifacts that can be related between them using relationships.

The NFV director model is an XML based model that changes the current way of working with HPSA.

Each operation (GUI, Northbound, or automation) is translated into XML data after it is stored in the corresponding persistent technology.

The model is not tied to any specific persistent technology.
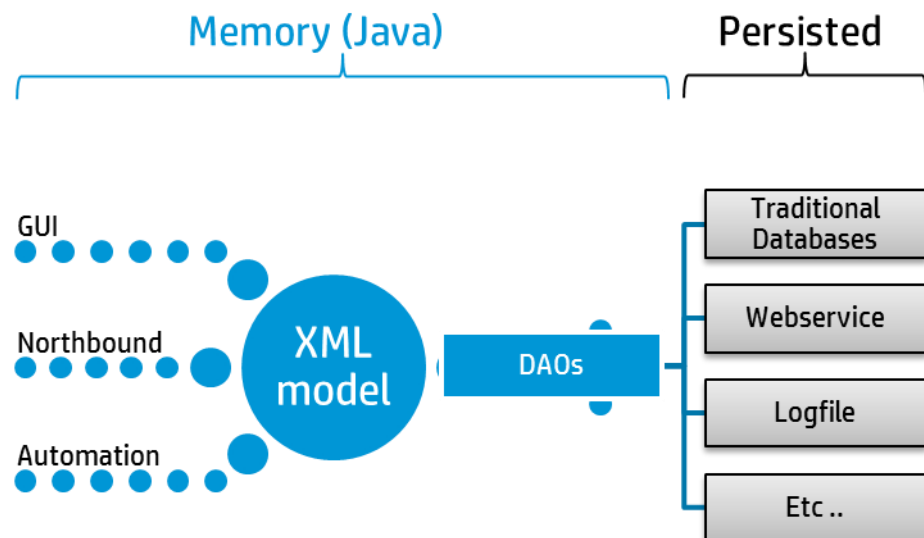
Figure 5: XML data model architecture

The NFV Director uses two master extended beans to model most of the systems.

4.1.1 Artifact

An artifact is an entity of any type that can have any number of attributes.

4.1.2 Relationship

A relationship is a parent-child relationship from one artifact to another. It can be of different types and contain any number of attributes.
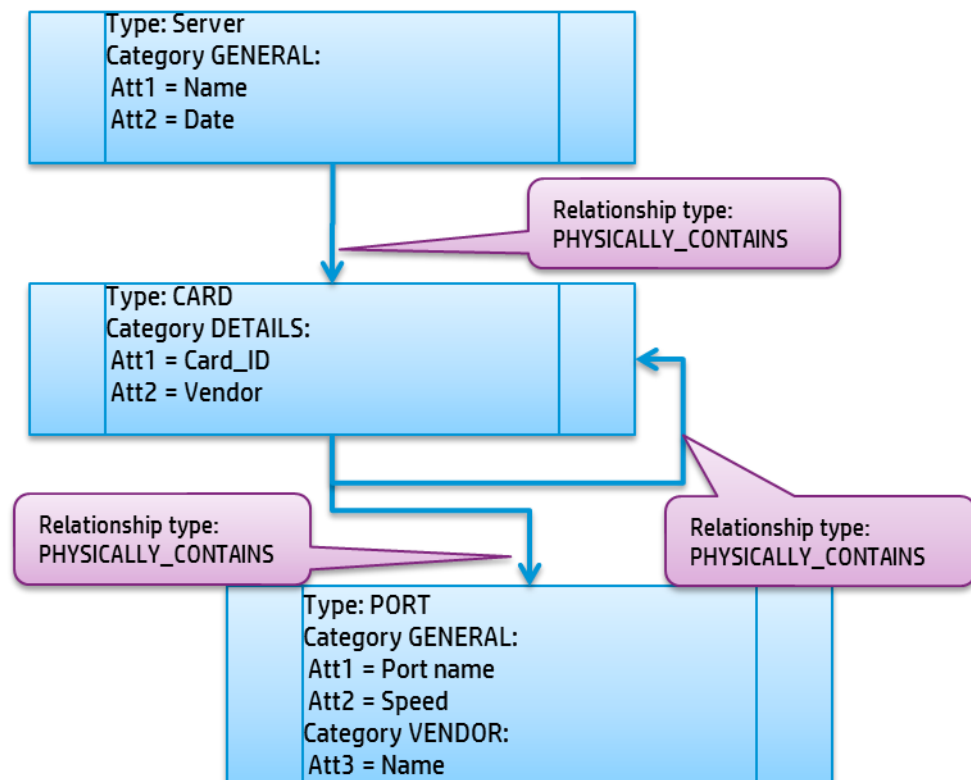
Figure 6: Example of artifact relationship

4.2 Flavors of artifacts and relationships

The three different flavors of artifacts and relationships are the following:

- Definitions
- Template
- Instance

4.2.1 Definition

The possible types of artifacts and their attributes along with the possible types of relationships and their attributes are defined in the DEFINITION tables.

The table also defines the artifacts that can be parents of different types of relationships to other artifacts.

The DEFINITION tables define what is possible. A set of definitions are shipped with the product. However, you can easily add new artifacts and relationships, or modify the existing ones.
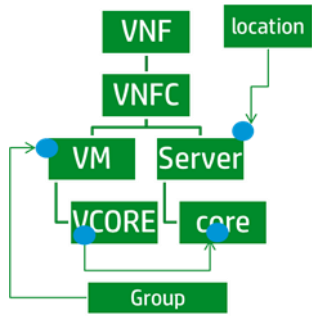
Figure 7: Example of artifact relationship definition

4.2.2 Template

A template is an instance that might contain policies and rules to fix the way it performs, when creating an instance based on a template. Templates can contain the catalog of the VNF and NS that are known by the NFV Director.
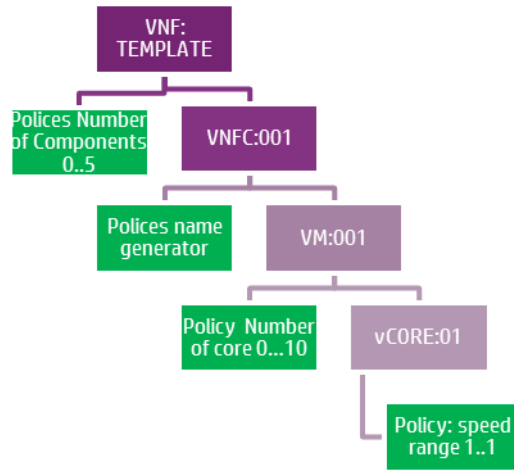
Figure 8: Example of artifact relationship template

4.2.3 Instance

Each definition can have numerous instances with different or equal values for each one of its attributes. The instance information contains the inventory of all provisioned services and resources.
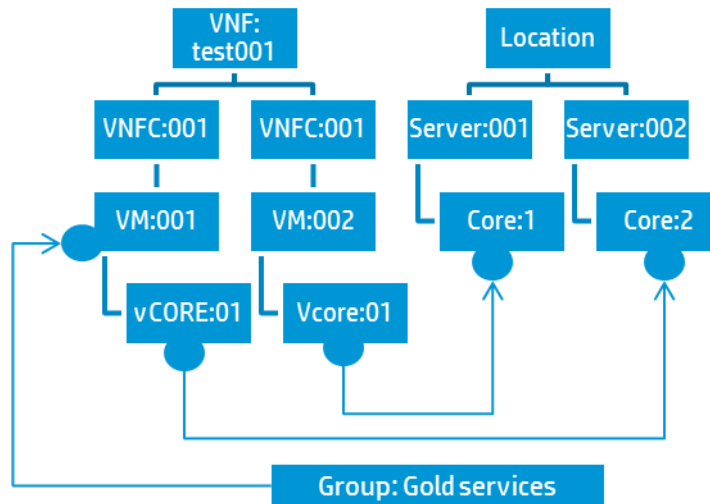
Figure 9: Example of artifact relationship instance

4.3 Out-of-the-box models

The product provides a set of out-of-the-box models, which allows you to model Virtual Network Functions, Network services, and datacenters.

4.3.1 Virtual Network Function model

The VNF model is composed of a set of definitions of artifacts and their relationships are described in the following illustration.
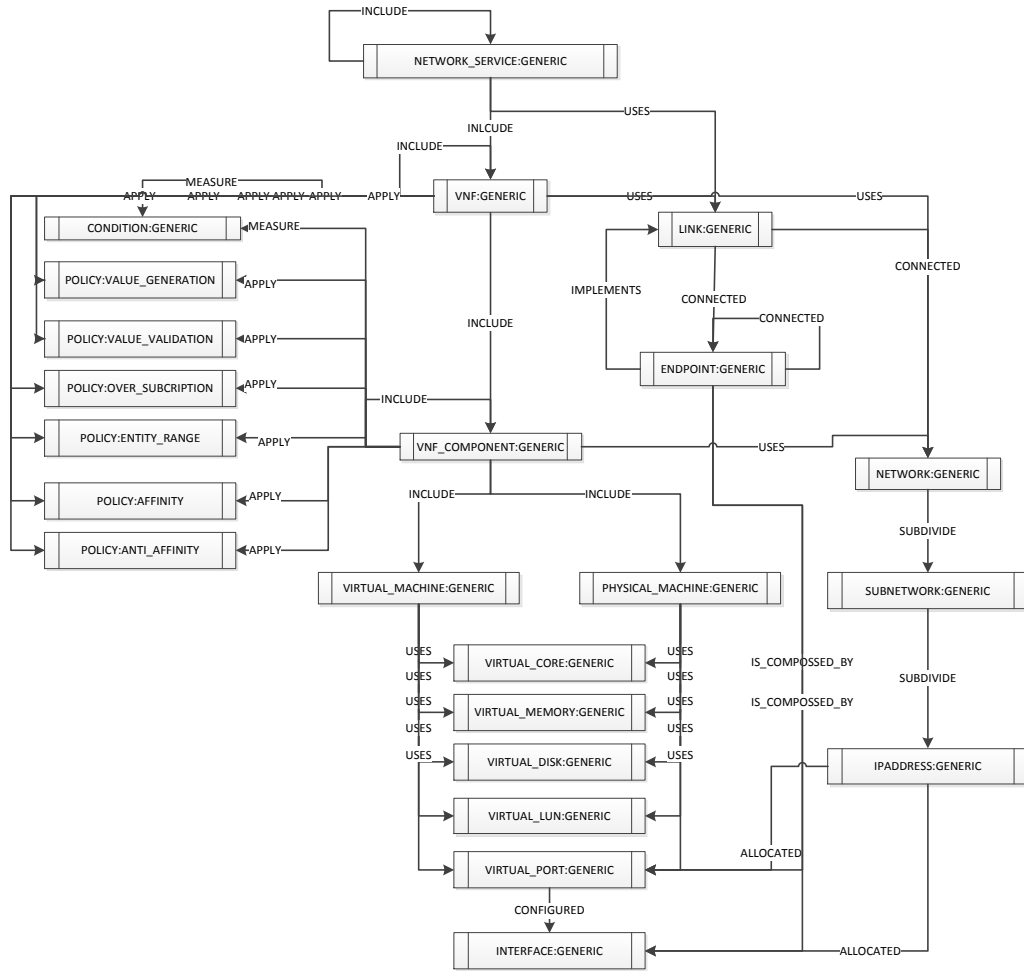
Figure 10: VNF Model

4.3.2 Resources model

The resources model is composed of a set of definitions of artifacts and their relationships are described in the following illustration.

Figure 11: Resources Model

4.3.3 Monitoring model

The monitoring model is composed of a set of definitions of artifacts and their relationships are described in the following illustration.
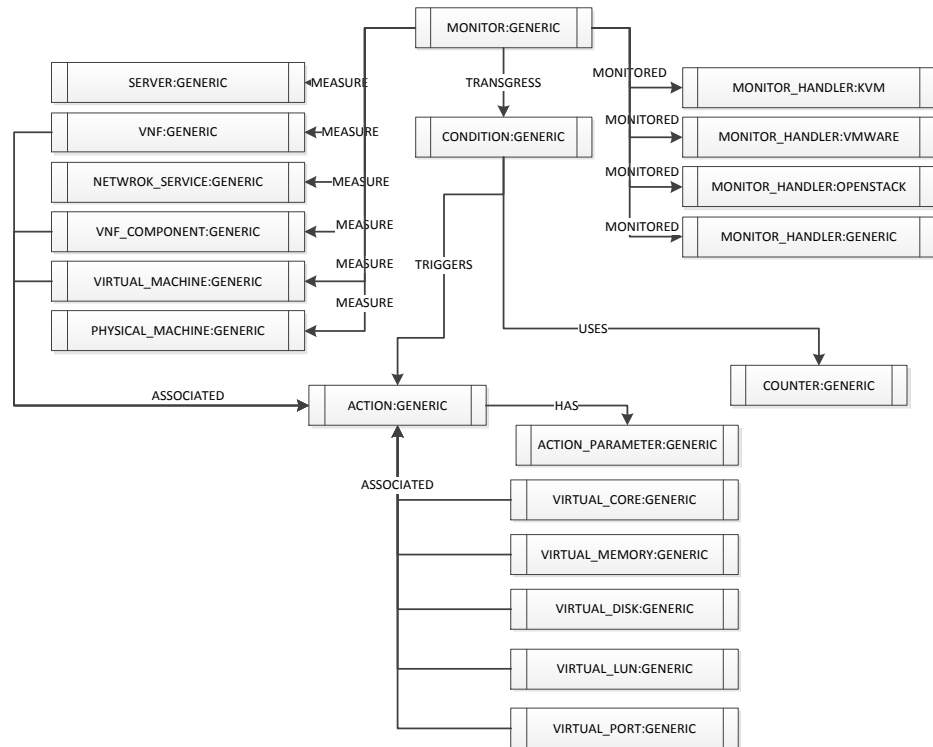
Figure 12: Monitoring Model

4.4 Particularities

4.4.1 Relationships

As all relationships are stored in the same database table (when using a database as storage mechanism), create your queries by using type or ID (PK) as filters. The tables are partitioned using types. Hence, filtering using a type is similar to querying only a table with the data of that type and not all the relationships in the system.

Chapter 5

Overview of NFV Director Operations

The NFV Model solution is a framework that provides certain capabilities for modeling network structures and elements. The use of these capabilities in data modeling is also explained in this section.

5.1 Data modeling and operations

5.1.1 Artifacts

Artifacts are objects that you can define from every nature. They represent the main unit of management in the NFV model. Artifacts can be interrelated through relationships.

5.1.2 Definitions

Definitions are base structures or the skeleton of an artifact. They have variable number of categories, attributes, and relationships. Definitions are a combination of different values for a structure:
`Family:Category:Group:Type:Subtype:Version`. This uniquely identifies an artifact and serves as the `primary key`. The following actions can be performed:

- Create Definitions
- Update Definitions
- View Definitions
- Delete Definitions

5.1.3 Templates

Templates are definitions with values of attributes set as required. They can be related with another template that fits the definition specification. Templates are also used to create instances from them. These instances contain set values of attributes from templates instead of default values from definitions. The following actions can be performed:

- Create Templates (From definition)
- Update Templates
- View Templates
- Delete Templates

5.1.4 Instances

Instances are definitions with values of the attributes set as required. They can be related to another instance that fits the definition specification. As a definition child (or sub child), instances inherit the combination from its parent. However, in this case, the combination can be repeated (the other way forces one definition to one instance). The following actions can be performed:

- Create Instances (From definition, from template)
- Update Instances
- View Instances
- Delete Instances

5.1.5 Relationships

Relationships are connections between artifacts and can be defined by any type. A definition (A) can be related with other definition (B), by a relationship type, and the templates and instances created from definition A can be related with the templates or instances (respectively) created from definition B. The following actions can be performed:

- Create Relationship (Create Definition process)
- Delete Relationship (Delete Definition process)

5.2 HPSA web - NFV inventory

This tool allows you to execute all available operations in the NFV Model.
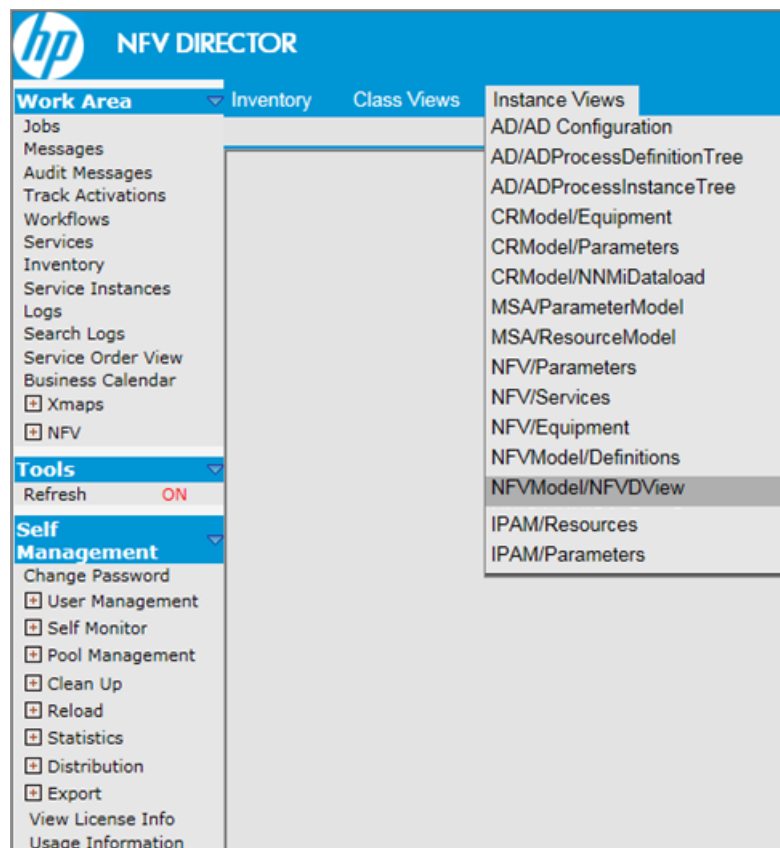
Figure 13: Inventory view - instance views
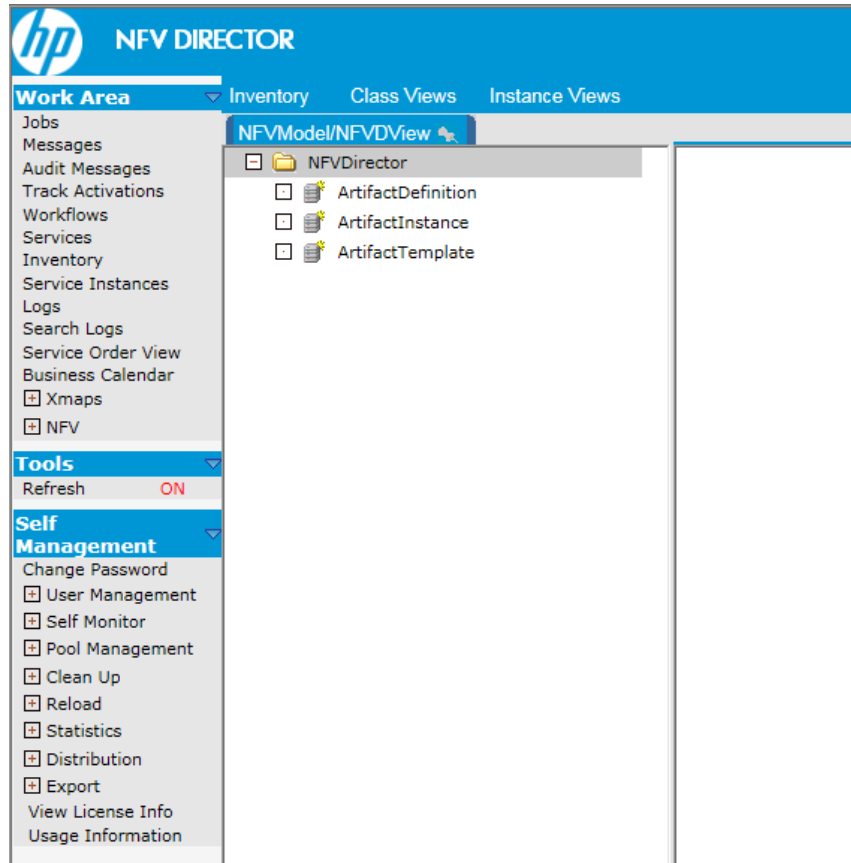
Figure 14: Inventory view-NFVModel/NFVDView

5.3 Definitions operations

The following image shows the various operations allowed for Definitions.
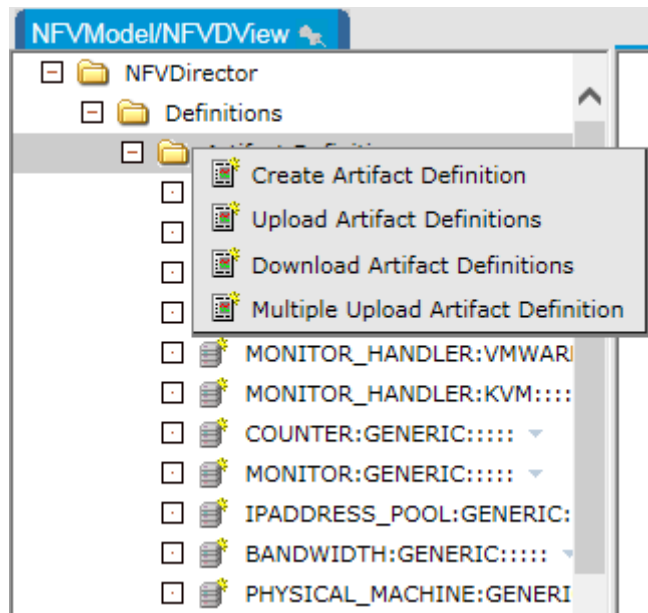
Figure 15: NFD Definition operations
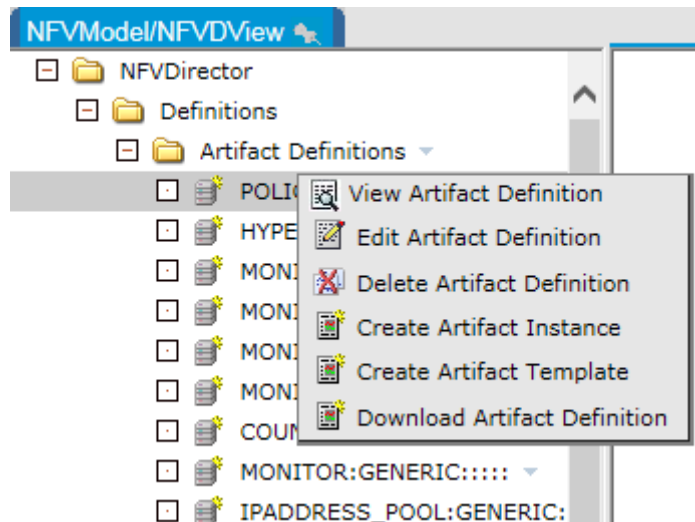
Figure 16: NFD artifact definition operations

5.3.1 Create Definition

This process creates an artifact definition and if specified, it creates a relationship with another definition that was previously created.

Fill the combination values, especially for the **Family** field, which is mandatory. The combination cannot be repeated. After creating some definitions, you can select values from the combo boxes.

You can set available statuses for each definition. These statuses are displayed as a list of eligible elements when creating and modifying instances/templates.

To add a new status into the list, enter the status description and click the **Add** button.

Available statuses entered:

- ENABLED

- DISABLED

- CHECKED

- DESIGNED

- RESERVED

- PROVISIONED

- ACTIVE

- TERMINATED

- INSTANTIATED

- LOCKED

You can define categories and sub categories of attributes. For each category or sub category, you can create as many attributes as required.

You can also select a relationship type and artifact related to the creation. Following is an example:

1. Enter the following details in the **Create Artifact Definition** pane.

Figure 17: Create Artifact Definition

Field	Description
Family	Set the value. For example, `VNF`.
Category	Set the value. For example, `GENERIC`.
Available Status Entered:	Enter `ON` and click **Add**. Enter `OFF` and click **Add**.
Add Category	1. Enter a category name. For example, `General`. 2. Click **Add**.

Table 3: Create Artifact Definition fields – an example

A category panel that contains **Add Attribute** and **Add Category** commands is generated. This **Add Category** inside the category panel creates a sub category from the **General** category.

Figure 18: Create Artifact Category panel

2. In the **General** pane, the category panel, enter the following details:

Field	Description
Add Attribute	Enter an attribute name. Enter the default value for attribute.

Field	Description
	Click **Add**. (Repeat this process for creating more attributes).
	Enter the Subcategory Name.
	Click **Add**.
Mandatory	Select the checkbox.
Type	Select a type from the drop-down list.
Unit	Select a unit from the drop-down list.
Name	Enter a name for the attribute and click **Add**. Repeat this process for creating more attributes.

Table 4: General category fields

3. Enter the sub category name and click **Add**.

4. In the **General_subcategory** pane, enter the required values similar to the values way in the **General** category pane fields.

5. Click the **Create Artifact** button.

6. To add parent-child relationship, select a **Parent Relationship: Type**.

Figure 19: Parent Child Relationship

7. Select the artifact related from the list of all artifacts.

8. Click the **Add Parent / Child Relationship Type**.

   For the parent relationship set as father, the artifact definition is being created and the artifact selected from the list of artifact definition is set as child.

9. Enter the category description and click **Add Category**.

10. In the **Category** pane, enter the required details.

    These categories and attributes defined in this window belong exclusively to the Relationship module.

Figure 20: Parent child relationship category

You can do the following tasks as well. It is possible to:

- Add new statuses.
  - Add new categories.
  - Add new attributes in a category.
  - Add new subcategories in a category.
  - Add new relationships.
  - Update the default values of attributes in a category / subcategory.
  - Update **is Physical** and **Enabled** values.

5.3.2 View Definitions

You can view all artifact properties by following this procedure:

1. Click the definition in the tree.
2. (Optional) Right-click the tree and select **View Artifact Definition**.

5.3.3 Delete Definitions

You can delete an artifact definition:

Right-click the definition in the tree and select **Delete Definition**.

5.3.4 Download Definitions

Use the following procedure to download the definitions.

1. Right-click the definition object and select the **Download definition**.
2. (Optional) Right-click the definitions branch and select **Download definitions.**

The NFVD generates an XML file with definitions which can be uploaded later.

5.3.5 Upload Definitions

Use the following procedure to upload the definitions.

Right-click the definitions branch and select **Upload definitions**.

You can upload definitions into NFV Model using the XML file, which you have downloaded or created.

5.3.6 Upload multiple Definitions

Use the following procedure to upload multiple definitions.

Right-click the definitions branch and select **Upload definitions**.

You can upload definitions into NFV Model using the XML file, which you have downloaded or created.

5.3.7 Other operations in Definitions

You can perform the following operations in addition to the ones mentioned above.

- Create Artifact Template.

- Create Artifact Instance.

These operations generate such Instances or Templates, setting the combination for primary values (family, category, group, type, subtype, and version) according to the original definition.

5.4 Templates operations

The following are the operations allowed on an artifact template:

- Upload artifact templates

- Download artifact templates
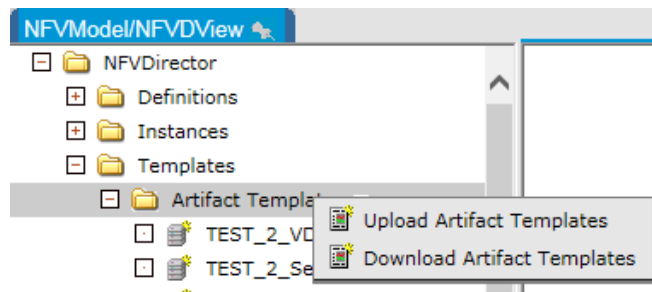
Figure 21: Download Artifact Templates

The following are the operations allowed on an artifact template:

- View artifact templates

- Edit artifact templates

- Delete artifact templates

- Create artifact instance

- Download artifact template

- Create instance from template
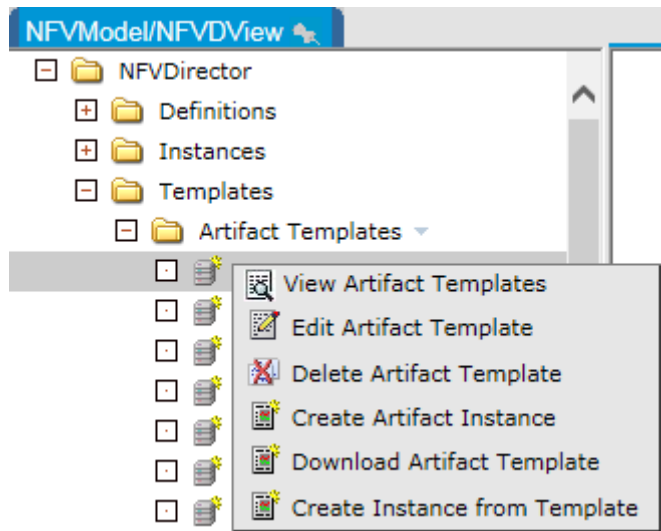
Figure 22: Create Instance from Template

5.4.1 Create Template

You should right-click the **Definition** and select the **Create Artifact Template** option to create an artifact template.

This operation can only be done from definitions branch of the tree. The template will inherit all properties (artifact values, relationships) from definition. The default values for attributes, properties "is physical", "status" (choose from definition list of statuses) and "Enabled" can simply be changed forming a new base for future instances.

Figure 23: Create Artifact Template from Definition

5.4.2 Edit Template

This process allows the user to change default values of the template.
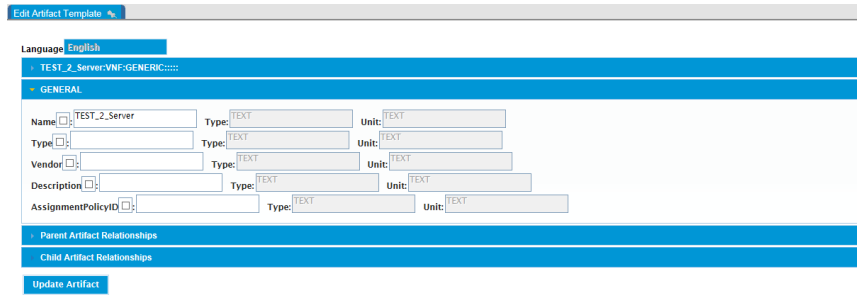
Figure 24: Edit Template

5.4.3 View Template

It is possible to view all artifact properties by left-clicking the instance in the tree or right-clicking it and selecting View Artifact Template.

5.4.4 Delete Template

It is possible to delete an artifact template by right-clicking the instance in the tree and selecting delete instance.

5.4.5 Upload and Download Artifact Templates

This is exactly the same process as Upload and Download Artifact Definitions. Obviously here are downloaded / uploaded templates.

5.4.6 Create artifact instance

This action will create an artifact instance with the values of current template. (Instance will take default values for attributes, primary combination, and relationships from template)

5.4.7 Create instance from template

To create an instance from a template, use the following procedure.

1. Select **Templates** > **Artifact Templates**.
2. Check the templates and their relationships.
3. Right-click the parent template and select **Create Instance from Template**.
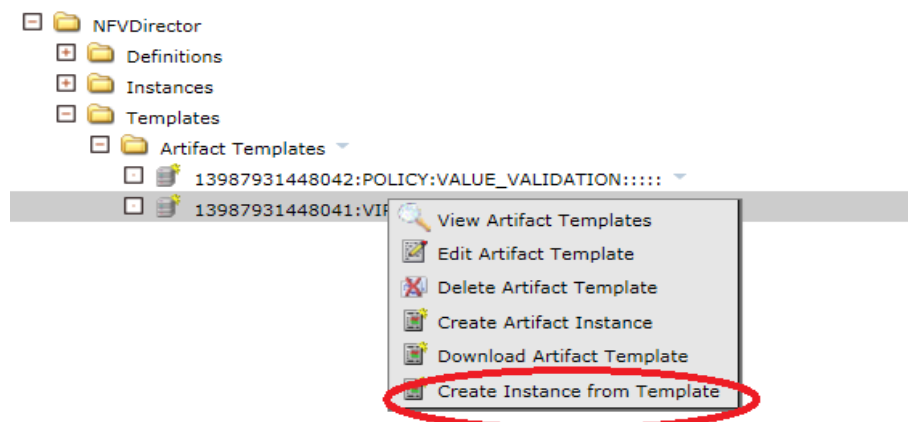
Figure 25: Create Instance from Template

4. Check the Instances, Artifact Instances.

   All instances should be created with the same attributes and relationship as the templates.
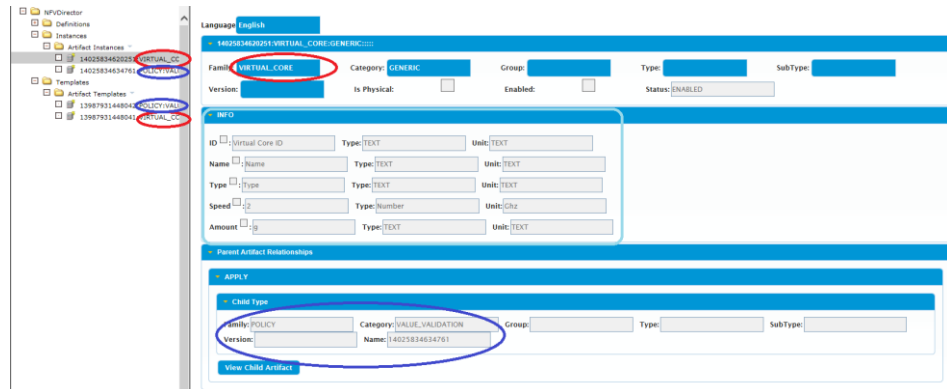
Figure 26: Verify the Artifact instance created

If the flow did not work properly, errors might appear on the screen with the 1XXX ID format.

To check different kind of errors, refer to the NFV Director documentation.

5.5 Instances operations

The following are the artifact instance operations:

- Create artifact instance
- Upload artifact instances
- Download artifact instances
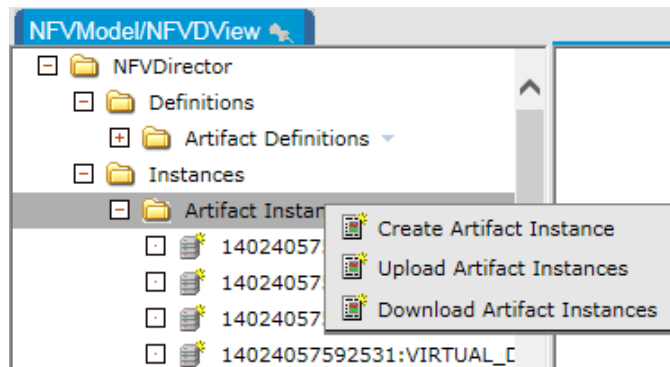
Figure 27: Download Artifact Instances

The following are the operations available for a specific instance:

- View artifact instances
- Edit artifact instances
- Delete artifact instances
- Create instance child

- Child of…

- Download artifact instance

- New view of artifact instance

- Delete artifact instance tree

- Scale-out

- Scale-in

- Scale up/down
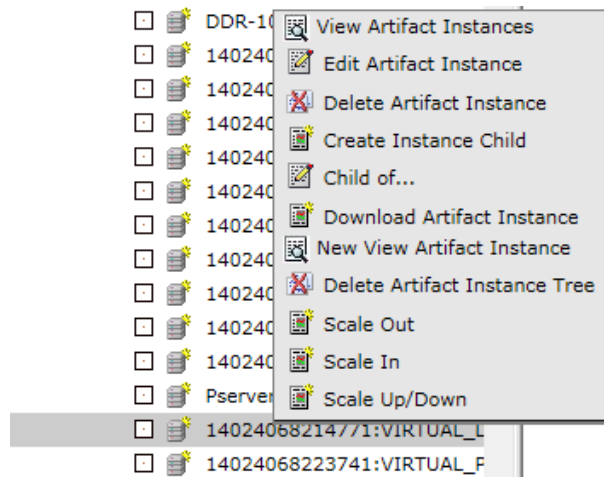
Figure 28: Artifact Instance operations

5.5.1 Create instance

Two ways of create an instance: from itself and defining one by one the values of the "primary combination" or from a definition (mentioned before) in which the values are preselected. Creation of an instance is mainly the same process as creation of a definition and does not require further explanation.

5.5.2 Edit instance

Editing an instance is similar to editing a definition, but you can edit only the values of the attributes in the instance.

5.5.3 View instance

It is possible to view all artifact properties by left-clicking the instance in the tree or right-clicking it and selecting View Artifact Instance.

5.5.4 Upload and download artifact instances

This is exactly the same process as Upload and Download Artifact Definitions. Obviously here are downloaded / uploaded Instances.

5.5.5 Delete artifact instance

It is possible to delete an artifact instance by right-clicking the instance in the tree and select delete instance.

5.5.6 Create instance child

Create an instance child is to instantiate a relationship definition, where the selected instance takes the role of the parent.

1. Select one instance from the tree and right-click the **Create Instance Child**.

2. The user should select the type of relationship among all types of relationships for which the selected instance can be the parent.

    The possible families are reloaded at family combo. The user-selected family and combos are refreshed with updated available categories.

3. Select the categories.

    The refreshing occurs again when the definition combination of values is completed.

4. Click the **Create Relationship** button.

5.5.7 Scale-out

This operation dynamically creates as many instances of one type (definition) as entity-range policy determines. You should define required relationships. For more details on the scale-Out operation, refer to the section *10.2 Scale-out*.

1. Right-click the instance and select the **Scale-Out** option.
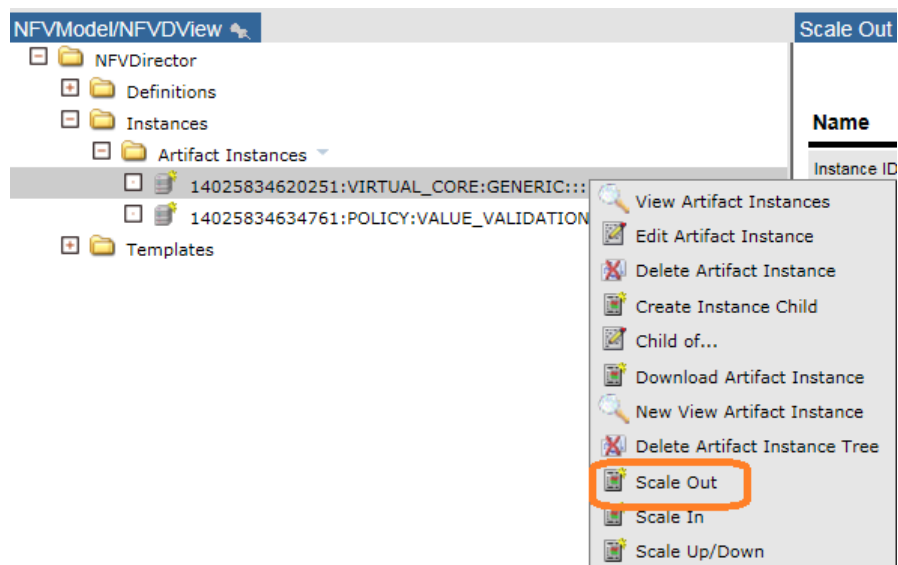
Figure 29: Scale out artifact instance

2. Check the Instance ID.
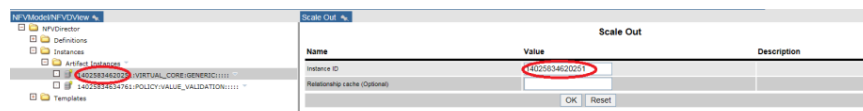
Figure 30: Verify Instance Id for scale out

3. Click **OK**.

    The workflow is launched and the scale performed.

If the flow did not work properly, errors might appear on the screen with the 4XXX ID format.

To check different kind of errors, see the *10.2.4.1 Errors* section.

5.5.8 Scale-in

This operation deletes dynamically as many instances of one type (definition) as entity-range policy determines. It is necessary to have defined relationships that are required. For more detailed information about Scale-In operation, refer to section *10.1 Scale-in*.

1. Select **Instance** > **Artifact Instances**.
2. Right-click the instance template and select the **Scale In** option.
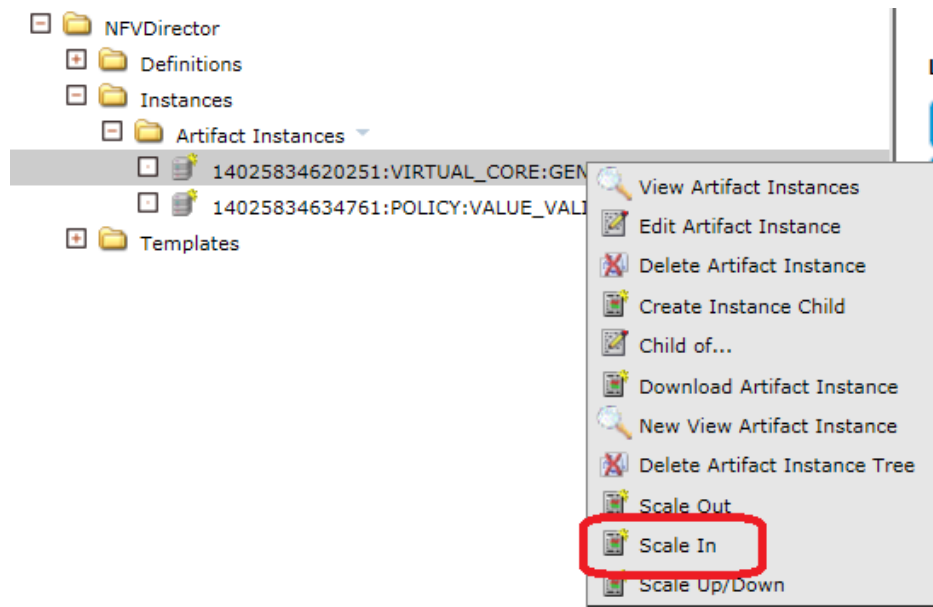
Figure 31: Scale in artifact instance

3. Check the Instance ID.
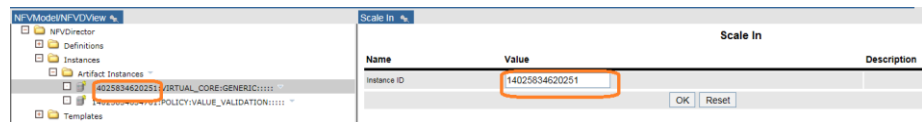
Figure 32: Verify Instance ID for scale in

4. Click **OK** to scale.
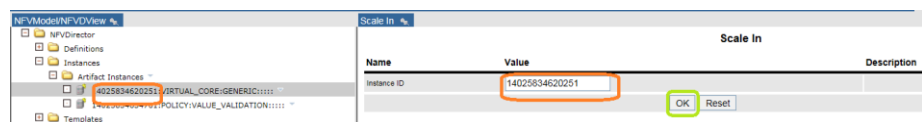
Figure 33: Confirm Scale In operation

The workflow is launched and the scale performed.

If the flow did not work properly, errors might appear on the screen with the 5XXX ID format.

To check different kind of errors, see the *10.1.4.1 Errors* section.

5.5.9 Scale up

On the instance VNF or VNF_COMPONENT, right-click and select the **Scale Up** option.
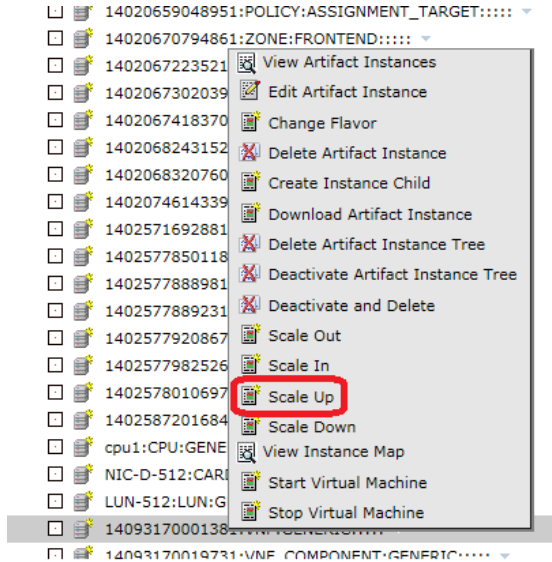
Figure 34: Scale up artifact instance

This operation enlarges attributes for an instance based on the entity-scale policy (see *8.3 Policies*). It is necessary to have defined relationships that are required. You can set the value of "Scale All Tree" and "Force Stop of Virtual Machines" options. If these fields are empty, the predefined values are "true" and "false" respectively.

The operation has an impact both in DB and on the Openstack side, changing the VM flavor, according to the attributes set in DB.

Scale Up

Name	Value	Description
Service Name	VNF	
Service Type	NFVD	
Service Operation	SCALE_UP	
Instance ID	14093170001381	
Scale All Tree?		
Force Stop of Virtual Machines?		
	OK Reset	

Figure 35: Scale up operations

5.5.10 Scale down

On the instance VNF or VNF_COMPONENT, right-click and select the **Scale Down** option.
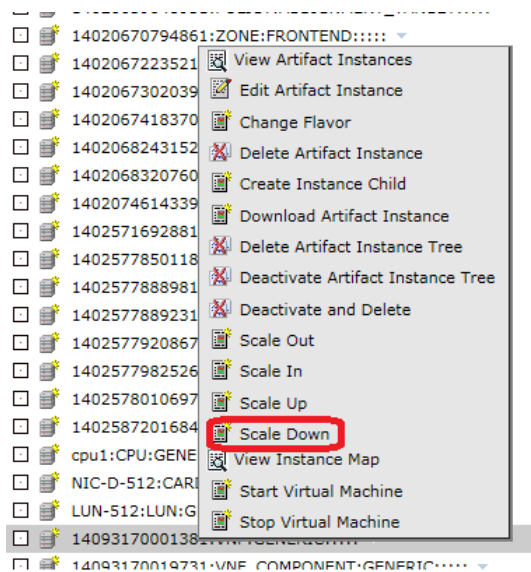
Figure 36: Scale down artifact instance

It is the opposite operation of **Scale Up**. Its function is reducing attributes for an instance based on the entity-scale policy (see *8.3 Policies*). It is necessary to have defined relationships that are required. You can set the value of "Scale All Tree" and "Force Stop of Virtual Machines" options. If these fields are empty, the predefined values are "true" and "false" respectively.

The operation has an impact both in DB and on the Openstack side, changing the VM flavor, according to the attributes set in DB.

Scale Down

Name	Value	Description
Service Name	VNF	
Service Type	NFVD	
Service Operation	SCALE_DOWN	
Instance ID	14093170001381	
Scale All Tree?		
Force Stop of Virtual Machines?		

OK   Reset

Figure 37: Scale down operations

5.5.11 Start Virtual Machine

Start virtual Machine operation has been built in order to start the virtual machines and make the assurance monitors active.
It works at VNF, VNF_COMPONENT and VIRTUAL_MACHINE, starting all the virtual machines associated to the operation launching level.
After the operation is executed, the virtual machines are started and they can be seen in active status in CS8/OpenStack web environment and active status in the database, with the labels active in the status fields respectively.

1. Select **Instance** > **Artifact Instances**.

2. Right-click the instance template and select the **Start Virtual Machine** option.
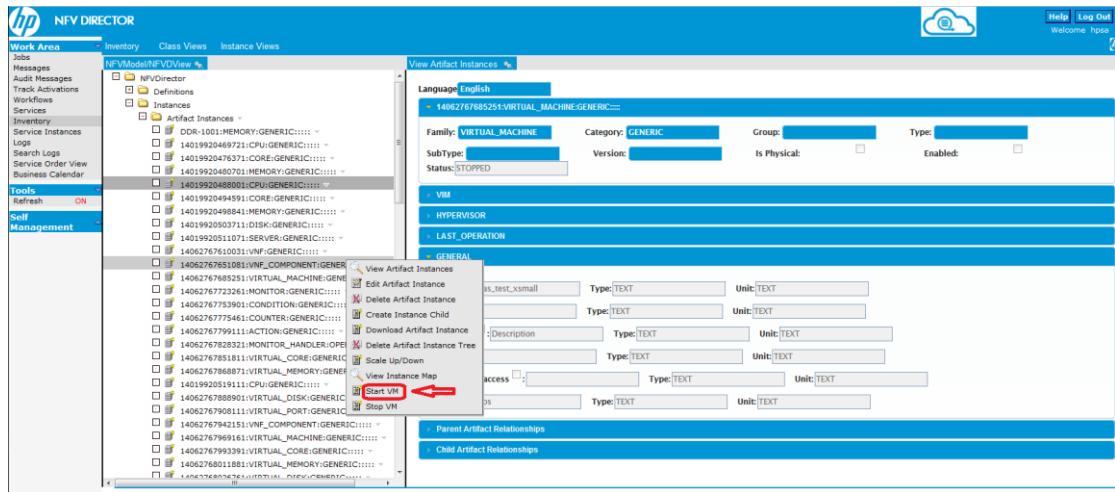
Figure 38: Start virtual machine

3. Check the Instance ID.
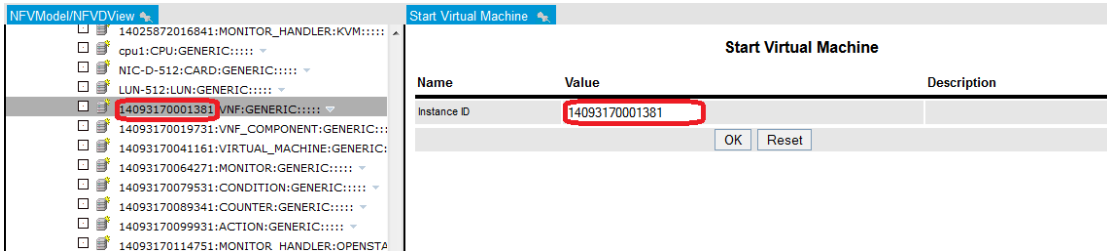
Figure 39: Verify instance ID for Start Virtual Machine

4. Press **OK** button to launch the Start Virtual Machine operation.
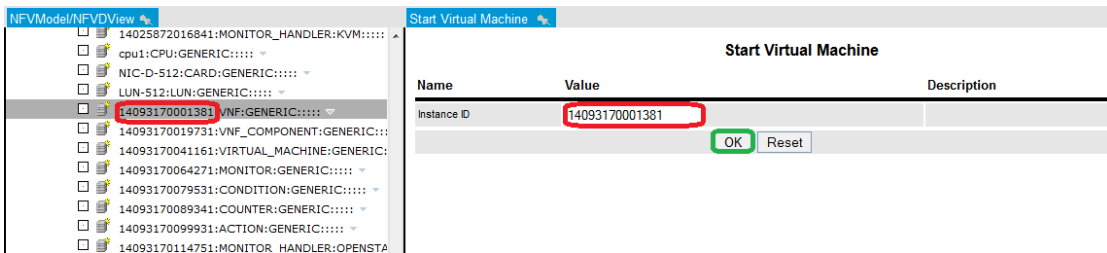
Figure 40: Confirm Start Virtual Machine operation

The workflow is launched and the Start Virtual Machine performed.

If the flow did not work properly, errors might appear on the screen with the 19XXX format.

To check different kind of errors, see the NFV Director documentation.

5.5.12 Stop Virtual Machine

Stop virtual Machine operation has been built in order to stop the virtual machines and the assurance monitors.

It works at VNF, VNF_COMPONENT and VIRTUAL_MACHINE, stopping all the virtual machines associated to the operation launching level.

After the operation is executed, the virtual machines are stopped and they can be seen in shutdown state in CS8/OpenStack web environment and stopped state in the database, with the labels shutdown and stopped in the status fields respectively.

1. Select **Instance** > **Artifact Instances**.

2. Right-click the instance template and select the **Stop Virtual Machine** option.
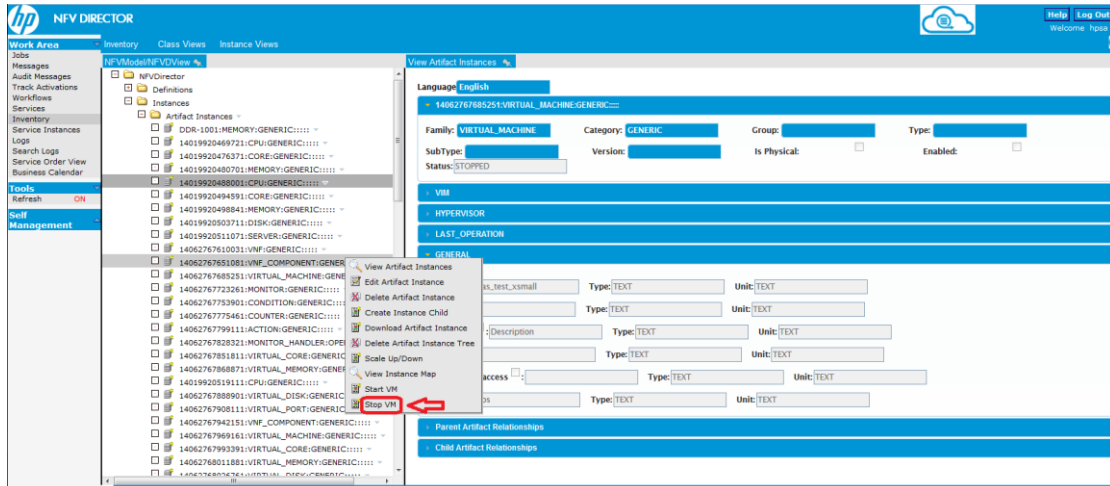
Figure 41: Stop virtual machine

3. Check the Instance ID.
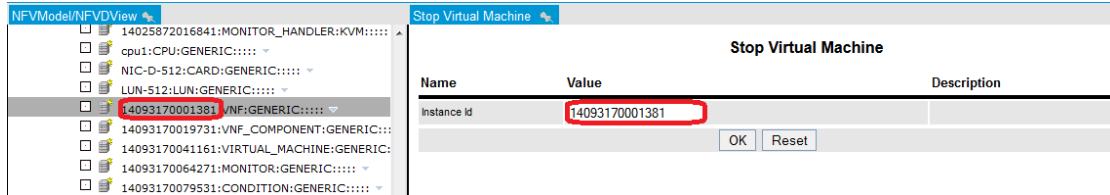
Figure 42: Verify instance ID for Stop virtual machine

4. Press **OK** button to launch the Stop Virtual Machine operation.
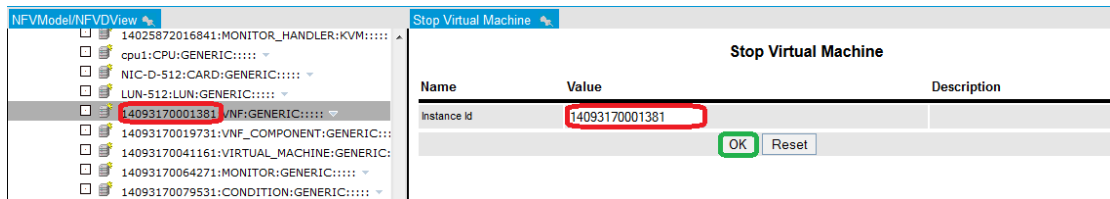
Figure 43: Confirm Stop Virtual Machine operation

The workflow is launched and the Stop Virtual Machine performed.

If the flow did not work properly, errors might appear on the screen with the 191XX format.

To check different kind of errors, see the NFV Director documentation.

5.6 Synchronizing NFVD Assurance and Fulfillment

When NFVD Assurance Gateway application starts, you may have to synchronize with NFVD Fulfillment on the infrastructure operations that Fulfillment is carried out and which the Assurance might have missed out.

You can synchronize them using the resynchronization feature of NFVD Assurance Gateway.

When starting, the Assurance Gateway reads the following parameters from `/var/opt/HP/nfvd/conf/fulfillment.properties` file to determine whether to synchronize at start-up.

S.No	Parameter	Comments
1.	RESYNC_AT_STARTUP	A *Boolean* flag that controls the behavior at startup. A value of *true* enables resynchronization at start-up. Default: *false*
2.	FULFILLMENT_URL	The fulfillment URL pointing to the northbound web services interface. Default: `http://12.0.0.1:8071/ngws/service`
3.	FULFILLMENT_CONNECTION_TIMEOUT	An Integer value in milliseconds that determines the connection timeout interval for the fulfillment web services connection. Default: 90000
4.	FULFILLMENT_RESPONSE_TIMEOUT	An Integer value in milliseconds that determines the response timeout interval for the fulfillment web services calls. Default: 90000

If the `RESYNC_AT_STARTUP` flag is set to `true`, the Assurance Gateway makes the Web service call exposed by Fulfillment to get the details, and it synchronizes the topology database.

```
# Configure RESYNC_AT_STARTUP as true/yes, for synchronization during Assurance startup
RESYNC_AT_STARTUP=false
# Provide the fulfillment URL to sync the data
FULFILLMENT_URL=http://127.0.0.1:8071/ngws/service
# Fulfillment URL connection timeout limit in millisecond, default 1.5 min
FULFILLMENT_CONNECTION_TIMEOUT=90000
# Fulfillment URL response for query timeout limit in millisecond, default 1.5 min
FULFILLMENT_RESPONSE_TIMEOUT=90000
```

Figure 44: fulfillment._properties

You can perform the Resync operation manually as well using the following steps:

1. Launch JConsole.
2. Select the `jboss-modules.jar` process in JConsole and click **Connect**.
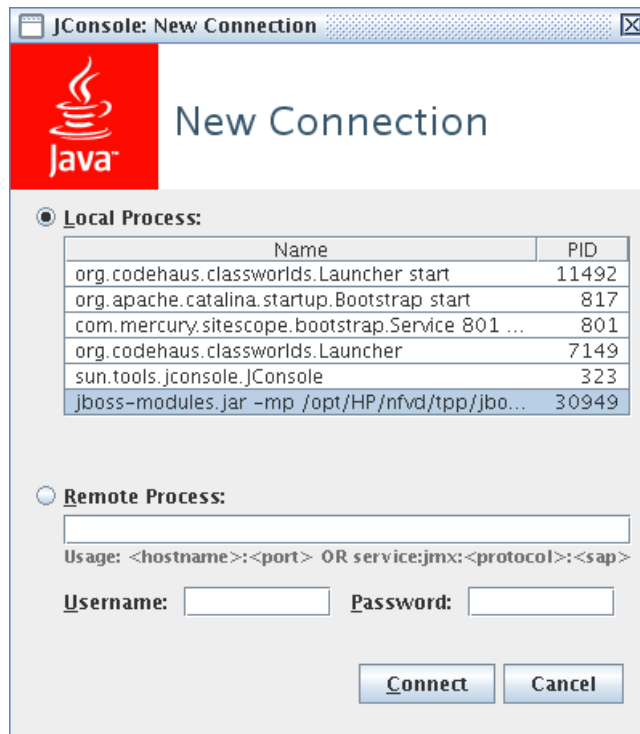
Figure 45: JBoss-modules.jar process in JConsole

3. Click the **MBean** tab to get the list of available operations.

4. Select **TopologyResync** > **startTopologyResync** operation.
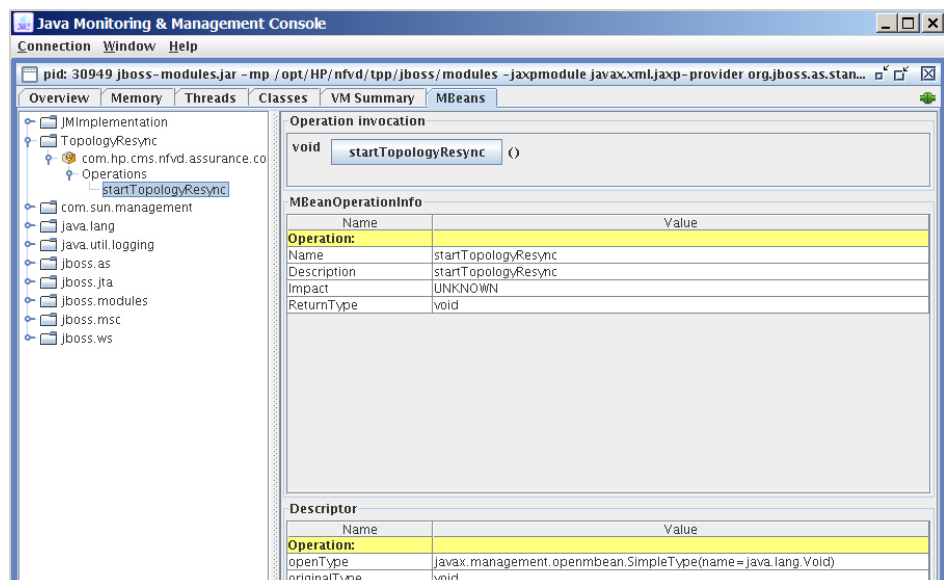
Figure 46: Choosing startTopologyResync in JConsole

5. Click the **startTopologyResync** button to start the synchronize operation.

   This operation uses the `FULFILLMENT_URL` as set in the
   `fulfillment.properties` file.

Chapter 6

Southbound interface

6.1 OpenStack plug-in

The OpenStack is the tool for final activation. The resource that is used to set up a virtual machine, modify a network, or query a system image as other operations. It provides the complete process for creating or modifying any operation from the beginning.

6.1.1 OpenStack CS8 user interface

HP Cloud System has implemented a Web environment to make user's interaction easier with the system. Although NFVD does not interact with this tool, it helps the user to check if NFVD OpenStack operations are performed without problems, as the NFVD operations are made using CS8.

6.1.2 OpenStack plug-in operations

The following operations are possible.

- Create, Edit, Query, and Delete Virtual Machine.

- Query for an Image.

- Create, Query and Delete a Flavor.

- Create, Query, Edit and Delete Networks and Subnets.

All listed operations can be executed from CS8 User Interface. However, automation is not available for these processes. The REST Northbound Interface is implemented for that purpose.

6.1.3 OpenStack templates

OpenStack plug-in uses templates for communicating the commands to the hypervisor. These templates should be of the same format as the JSON request for OpenStack. These expected request formats can be found in the OpenStack API documentation at: http://developer.openstack.org/api-ref-compute-v2.html.

You should create a new template for each new operation to be implemented. Templates can be created / listed and edited through the HPSA Solution Container: **Administrator**> **ECP** > **Activation Commands Template** > **Template**.
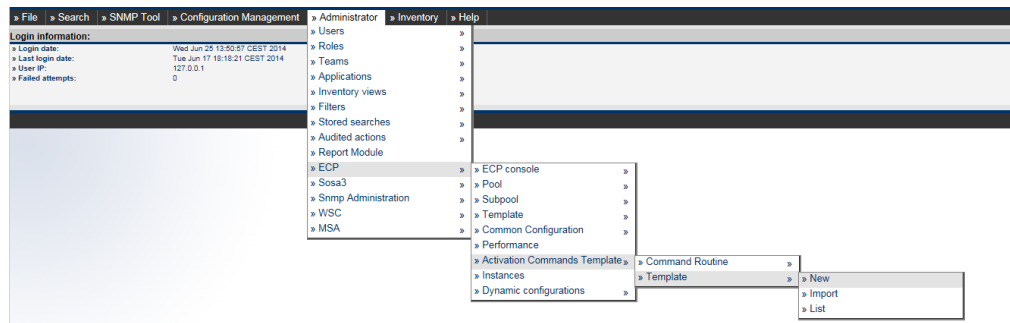
Figure 47: HPSA Solution Container ECP command template

The following is an example of creating a server template.

```
[TEMPLATE:Config]
http.operation=POST

#http.url.sufix=/v2/${tenant_id}/servers

http.url.sufix=/servers

openstack.endpointtype=compute

[TEMPLATE:Do]

[TEMPLATE:Section 0]
{

    "server":{

        "name":"${SERVER_NAME}",

        "imageRef":"${IMAGE}",

        "flavorRef":"${FLAVOR}",

        "max_count":1,

        "min_count":1,

        "networks":[

            {

                "uuid":"${SERVER_NETWORK_ID}"

            }

        ],

        "security_groups":[

            {

                "name":"default"

            }
```

Figure 48: Example server template

The template is organized in two sections:

- Config
  In the Config section, the following details are provided.
  Operation:

    - POST for creating

    - PUT for editing

    - GET for querying

    - DELETE for deleting

    - http.url.sufix—Refers to the path in the API

    - Type of endpoint (depending on the operation):

- Compute for virtual machines and images

- Network for networks

- DO—Section 0 is specified as the concrete JSON request expected by the OpenStack API. The JSON is a compounded structure with pairs (variable:value). Variables like ${SERVER_NAME} can be inserted in the template and the plug-in replaces it with a value passed through the specific workflow.

6.1.4 OpenStack Workflows

In the current version, a unique activation workflow is deployed for each necessary operation. The structure in the workflows is always the same:

- Get values from outside

  - Authentication Values

  - Activation Values (Server Name, Network UUID)

- Add Activation Values inside a HashMap

- Invoke the plug-in with those values

  - Authenticate

  - Execute concrete operation

- Check correct activation

- If activation was OK get the OpenStack Response into an object

- Send the object to the workflow caller
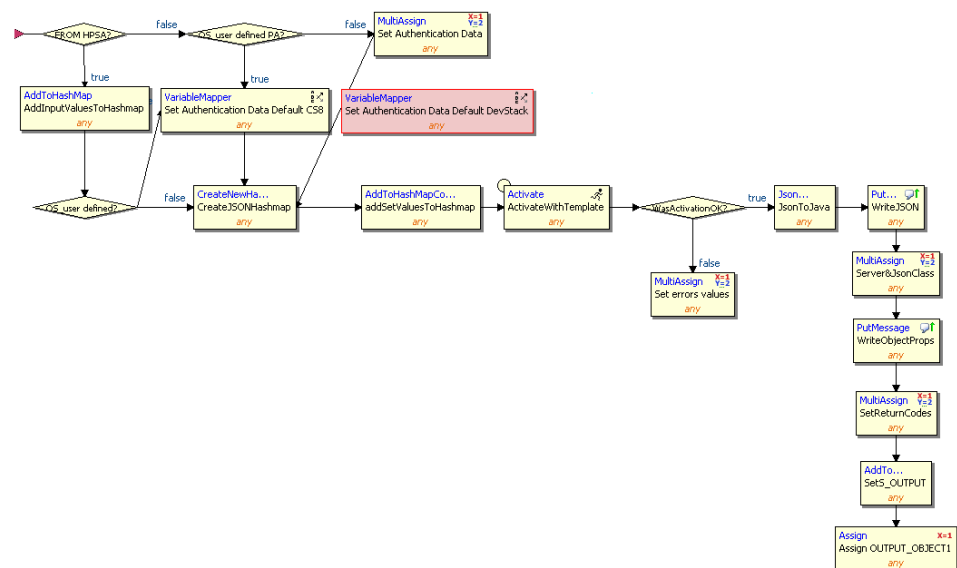
Figure 49: Example: Create Server Workflow

6.2 CS8 – REST Interface

The Rest Interface helps automate OpenStack operations. A Rest client is required to run those operations. This section explains the procedure to call operations using the Firefox Rest Client.

56
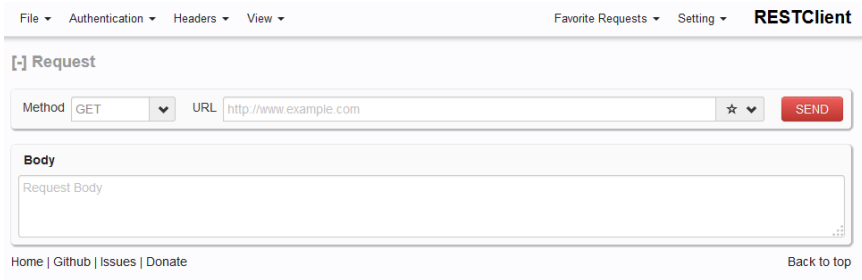
Figure 50: CloudSystem Firefox Rest Client

You should indicate proper headers.
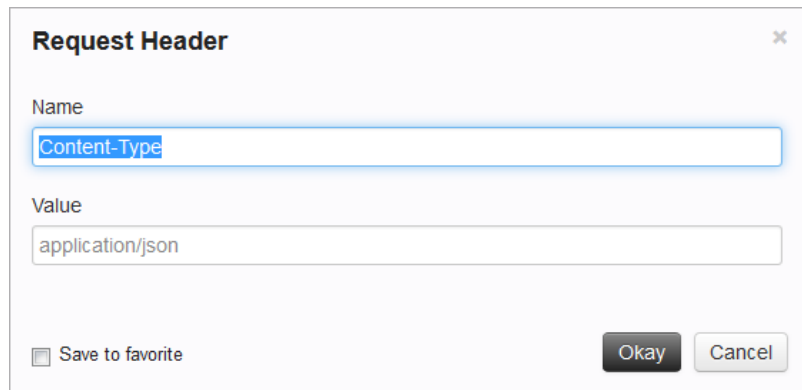
Figure 51: CloudSystem Rest Client Request Header

6.2.1 Operations

6.2.1.1 Create Server
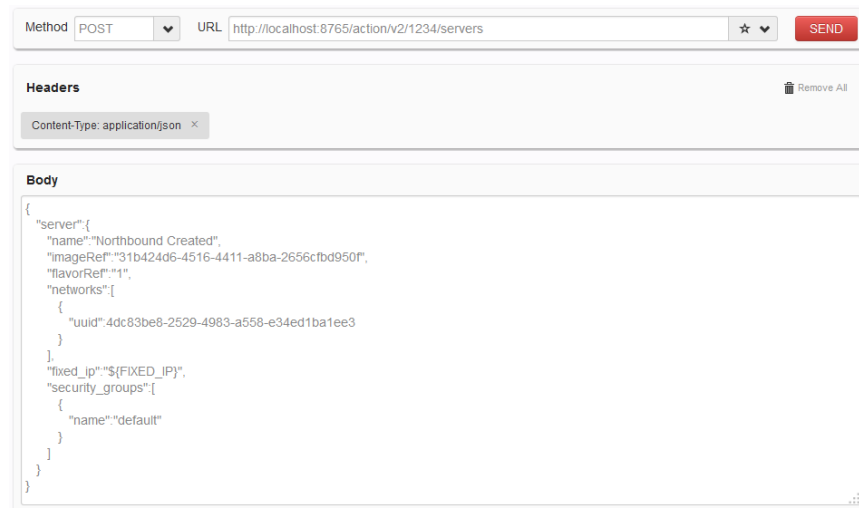
Figure 52: CloudSystem Create Server operation

URL: `http://localhost:8765/action/v2/1234/servers`
`http:$host:$Protocol_Adapter_port/action/$version_api/$tenant/s`
`ervers`

6.2.1.2 Edit Server
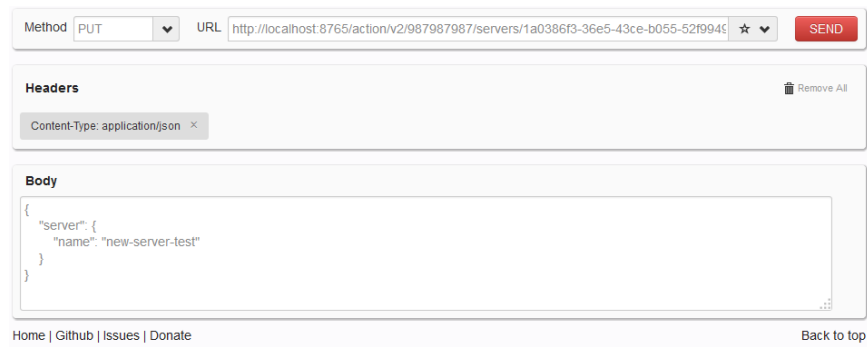
Figure 53: CloudSystem Edit Server operation

URL: `http://localhost:8765/action/v2/987987987/servers/1a0386f3-36e5-43ce-b055-52f994924e36`

`http:$host:$Protocol_Adapter_port/action/$version_api/$tenant/servers/$serverId`

6.2.1.3 Get Server by Id

Figure 54: CloudSystem Query Server operation

URL: `http://localhost:8765/action/v2/987987987/servers/1a0386f3-36e5-43ce-b055-52f994924e36`

`http:$host:$Protocol_Adapter_port/action/$version_api/$tenant/servers/$serverId`

6.2.1.4 Delete Server

Figure 55: CloudSystem Delete Server operation

URL: `http://localhost:8765/action/v2/987987987/servers/1a0386f3-36e5-43ce-b055-52f994924e36`

`http:$host:$Protocol_Adapter_port/action/$version_api/$tenant/servers/$serverId`

6.2.1.5 Start Server
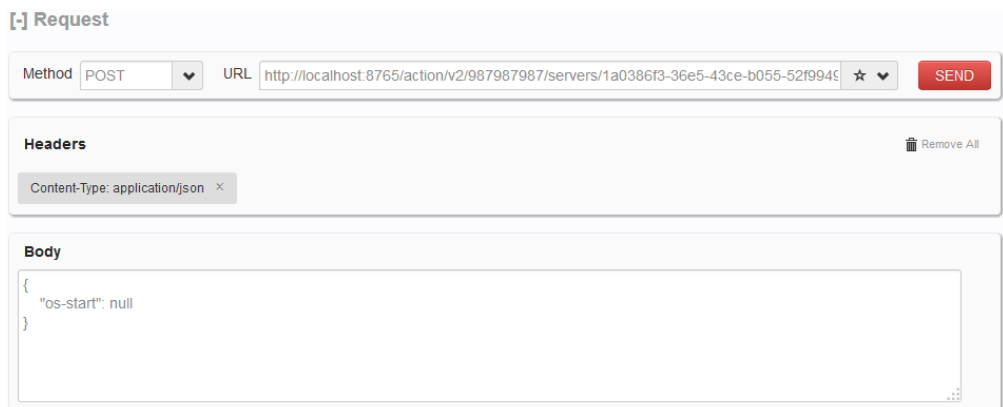
Figure 56: CloudSystem Start Server operation

URL: `http://localhost:8765/action/v2/987987987/servers/1a0386f3-36e5-43ce-b055-52f994924e36`

`http:$host:$Protocol_Adapter_port/action/$version_api/$tenant/servers/$serverId`

6.2.1.6 Stop Server

Figure 57: CloudSystem Stop Server operation

URL: `http://localhost:8765/action/v2/987987987/servers/1a0386f3-36e5-43ce-b055-52f994924e36`

`http:$host:$Protocol_Adapter_port/action/$version_api/$tenant/servers/$serverId`

6.2.1.7 Query Image
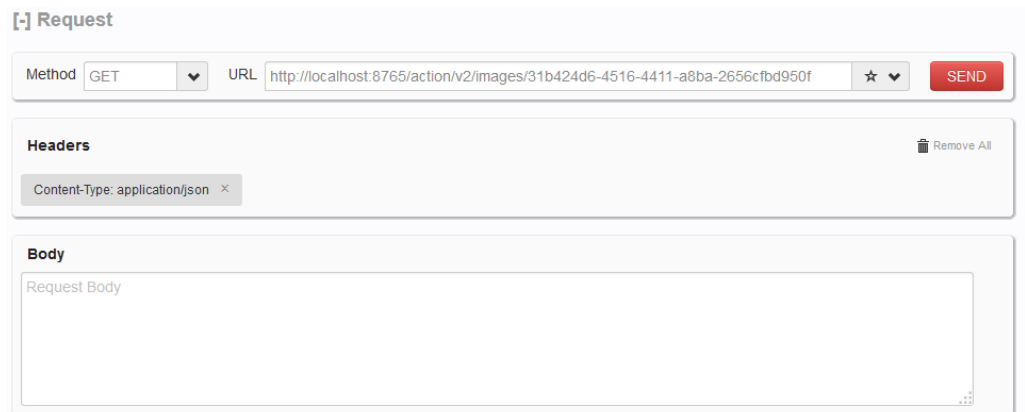
Figure 58: CloudSystem Query Image operation

URL: `http://localhost:8765/action/v2/images/31b424d6-4516-4411-a8ba-2656cfbd950f`
`http:$host:$Protocol_Adapter_port/action/$version_api/images/$imageId`

6.2.1.8 Create Flavor

Figure 59:CloudSystem Create Flavor operation

URL: `http://localhost:8765/action/v2/flavors`
`http:$host:$Protocol_Adapter_port/action/$version_api/flavors`

6.2.1.9 Get Flavor

Figure 60: CloudSystem Query Flavor operation

URL: `http://localhost:8765/action/v2/flavors/10`
`http:$host:$Protocol_Adapter_port/action/$version_api/flavors/$flavorId`

6.2.1.10 Get Flavor by Parameters
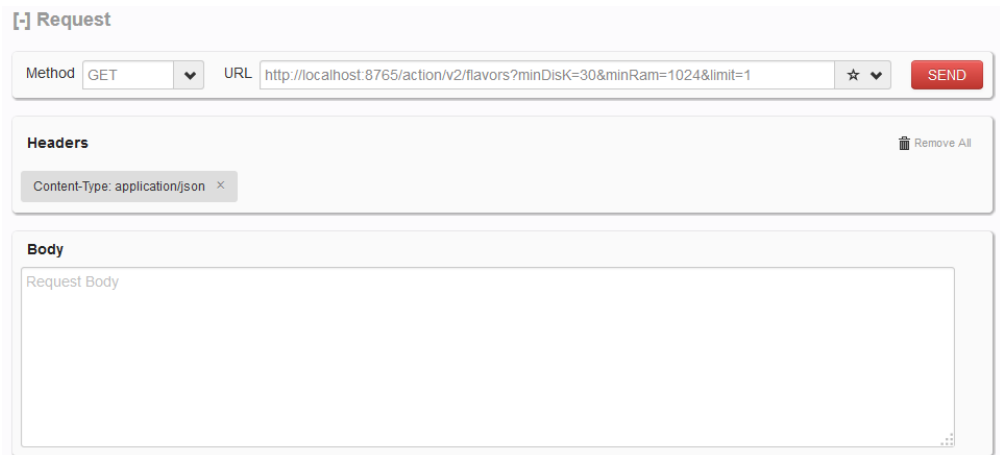
Figure 61: CloudSystem Query Flavor by Parameters operation

URL:
```
http://localhost:8765/action/v2/flavors?minDisK=30&minRam=1024&
limit=1
http:$host:$Protocol_Adapter_port/action/?$param1=$value1&$para
m2=$value2
```

6.2.1.11 Delete Flavor
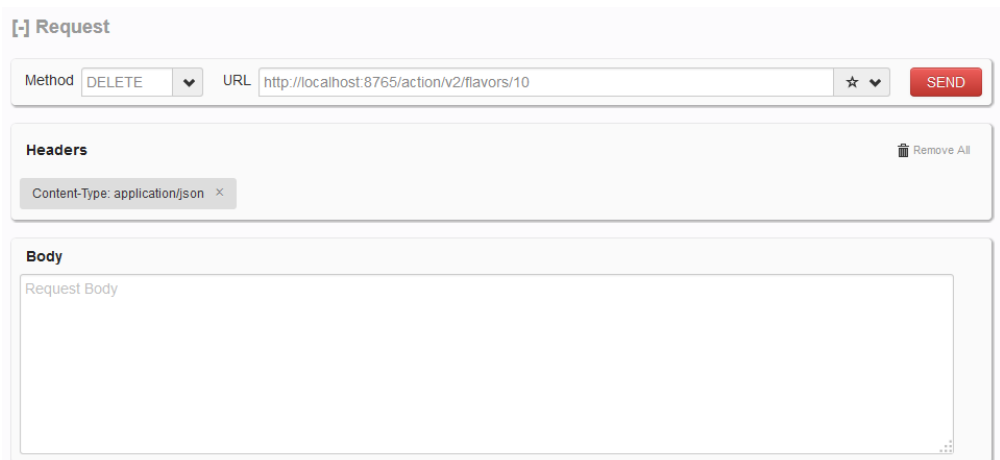
Figure 62: CloudSystem Delete Flavor operation

URL: `http://localhost:8765/action/v2/flavors/10`
```
http:$host:$Protocol_Adapter_port/action/$version_api/flavors/$
flavorId
```

6.2.1.12 Create Network

Figure 63: CloudSystem Create Network operation

URL: `http://localhost:8765/action/v2.0/networks`
`http:$host:$Protocol_Adapter_port/action/$version_api/networks`

6.2.1.13 Edit Network

Figure 64: CloudSystem Edit Network operation

URL: `http://localhost:8765/action/v2.0/networks/4dc83be8-2529-4983-a558-e34ed1ba1ee3`
`http:$host:$Protocol_Adapter_port/action/$version_api/networks/$network_id`

6.2.1.14 Get Network By ID

Figure 65: CloudSystem Query Network by ID operation

URL: `http://localhost:8765/action/v2.0/networks/4dc83be8-2529-`
`4983-a558-e34ed1ba1ee3`
`http:$host:$Protocol_Adapter_port/action/$version_api/networks/`
`$network_id`

6.2.1.15 Get Network by Parameters

Figure 66: CloudSystem Query Network by Parameters operation

URL:
`http://localhost:8765/action/v2.0/networks?name=MyNetworkTestin`
`g`
`http:$host:$Protocol_Adapter_port/action/$version_api/networks?`
`$param1=$value1`

6.2.1.16 Delete Network

Figure 67: CloudSystem Delete Network operation

URL: `http://localhost:8765/action/v2.0/networks/4dc83be8-2529-4983-a558-e34ed1ba1ee3`
`http:$host:$Protocol_Adapter_port/action/$version_api/networks/$network_id`

6.2.1.17 Create Subnet

Figure 68: CloudSystem Create Subnet operation

URL: `http://localhost:8765/action/v2.0/subnets`
`http:$host:$Protocol_Adapter_port/action/$version_api/subnets`

6.2.1.18 Edit Subnet

Figure 69: CloudSystem Edit Subnet operation

URL: `http://localhost:8765/action/v2.0/subnets`
`http:$host:$Protocol_Adapter_port/action/$version_api/subnets/$subnet_id`

6.2.1.19 Get Subnet

Figure 70: CloudSystem Query Subnet operation

URL: `http://localhost:8765/action/v2.0/subnets`
`http:$host:$Protocol_Adapter_port/action/$version_api/subnets/$`
`subnet_id`

6.3 VIMs configuration

NFV Director needs to know the appropriate URL and credentials for the VIM and the appropriate tenants so they can be configured at the plugin level, workflow level and monitoring level.

If the credentials in CS8 do not match the operations NFVD triggers some errors will be raised at runtime execution.

Out of the box there are no Workflows to end2end create Networks although the atomic operation is provided in the OpenStack southbound plugin so they should be created upfront on the VIM or a Workflow developed to create them when needed.

If NFV director is going to choose the server where the VMs are going to be created then OpenStack regions and availability zones need to be created to the level of server so that the NFV director can specify the server using the region or availability zone.

Chapter 7

Northbound Interface

7.1 SOSA as NFVD operator

Apart from Web user interface NFVD operations can be executed using SOSA web services interface also by sending appropriate commands. In this section xml format and "SOAP UI" tool are used for illustration.

7.1.1 Understanding SOAP requests

Identify the operation with its service action:

```
<dyn:name>
<dyn:type>
<dyn:action>
```

The three parameters identify uniquely the service action to be executed.

Request body:

```
<dyn:name>REQUEST</dyn:name>
<dyn:value>REQUEST-BODY</dyn:value>
```

It has to be set here as XML structure that SOSA can read and convert into an NfvModel object (artifact definition, instance, and template or artifact relationship).

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws" xmlns:dyn="http://www.hp.com/sosa/dynamicserviceorder">
   <soapenv:Header/>
   <soapenv:Body>
      <ngws:startDynamicOrderSync>
         <dyn:serviceRequest>
            <dyn:services type="NFVD" name="DEFART" action="CREATE">
               <dyn:service>
                  <dyn:name>DEFART</dyn:name>
                  <dyn:type>NFVD</dyn:type>
                  <dyn:action>CREATE</dyn:action>
                  <dyn:characteristics>
                     <!--1 or more repetitions:-->
                     <dyn:characteristic>
                        <dyn:name>REQUEST</dyn:name>
                        <dyn:value><![CDATA[
                              <REQUEST>
                        ]]></dyn:value>
                     </dyn:characteristic>
                  </dyn:characteristics>
               </dyn:service>
            </dyn:services>
         </dyn:serviceRequest>
         <ngws:user>foo</ngws:user>
      </ngws:startDynamicOrderSync>
      <ngws:startDynamicOrderAsync><dyn:serviceRequest/><ngws:user/></ngws:startDynamicOrderAsync></soapenv:Body>
</soapenv:Envelope>
```

7.1.2 Operations

Different Requests body, different service action.

7.1.2.1 Artifact Definition Create

(DEFART-CREATE Test)

```
<dyn:services type="NFVD" name="DEFART" action="CREATE">
```

```
Service Name: DEFART
Service Action Name: NFVD
Operation: CREATE
Starts with: <createArtifactDefinitions xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Language of the artifact/s: <language></language>
List Tag (more than one artifact at a time): <artifactDefinitions>…</artifactDefinitions>
Artifact Tag: <artifactDefinition>…</artifactDefinition>
Primary combination of values (never repeated in definitions):
<artifactFamily>Family_name</artifactFamily>
<artifactCategory>category_name</artifactCategory>
<artifactGroup>group_name</artifactGroup>
<artifactType>type_name </artifactType>
<artifactSubtype>subtype_name </artifactSubtype>
<artifactVersion>version_name </artifactVersion>
Physical: <isPhysical>true/false</isPhysical>
Enabled:<enabled>true/false</enabled>
List of categories:<categories>…</categories>
Category:
<category>
                <label>category_label</label>
                <version>version_of_category</version>
                <order>1 to n</order>
                <attributes>…</attributes>
                <subcategories>…</subcategories>
<category>
```

It sets the label, version, and order of the category. Two more tags for attributes and categories inside categories are available.

- Attributes—Need to define label, type, unit, defaultValue, mandatory and order.

```
<attribute>
        <label>attribute_label</label>
        <type>
        <label>type_label</label>
                <descriptionLabel>type_description_label</descriptionLabel>
                <description>type_description</description>
        </type>
        <unit>unit_name</unit>
        <defaultValue>default_value</defaultValue>
        <mandatory>true/false</mandatory>
        <order>1 to n</order>
</attribute>
```

- Subcategories—Same as categories, repeated inside the inner `<categories>` tag.

  List of available statuses: `<availableStatus>…</ availableStatus>`

  Status label, visible label and whether it is enabled or not:
```
<status>
        <label>status_label</label>
        <visibleStatusLabel>visible_status_label</visibleStatusLabel>
        <enabled>true/false</enabled>
</status>
```

Ends with: `</createArtifactDefinitions>`.

7.1.2.2 Artifact Definition Update

(DEFART-UPDATE Test)
```
<dyn:services type="NFVD" name="DEFART" action="UPDATE">
Service Name: DEFART
Service Action Name: NFVD

Operation: UPDATE
```

This is the same operation as Artifact Definition Create in terms of the request body.

Starts with:

```
<updateArtifactDefinitions  xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
```

Ends with:

```
</updateArtifactDefinitions>
```

7.1.2.3 Artifact Definition Get (DEFART-GET Test)

```
<dyn:services type="NFVD" name="DEFART" action="GET">
Service Name: DEFART
Service Action Name: NFVD
Operation: GET
Starts with:
<query xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Language:
<languageCode>language_code</languageCode>
Query criteria: is always the same structure (in all artifacts)
<criteriaSet>
          <criteria>
                    <paramName>paramName(family, category…) </paramName>
                    <paramValue>paramValue</paramValue>
                    <comparator>comparatorType(EQ, LIKE…)</comparator>
          </criteria>
</criteriaSet>

Ends with: </query>
```

7.1.2.4 Artifact Definition Delete (DEFART-DELETE Test)

```
<dyn:services type="NFVD" name="DEFART" action="DELETE">
Service Name: DEFART
Service Action Name: NFVD
Operation: DELETE
```

Starts with:

```
<deleteArtifactDefinitions xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
```

List of artifact definitions to be deleted:

```
<artifactDefinitions>…</artifactDefinitions>
```

This attribute is required to specify uniquely the definition to be deleted. As mentioned earlier, the combination of Family:Category:Group:Type:Subtype:Version is the only way to do this.

```
<artifactDefinition>
          <artifactFamily>family_value</artifactFamily>
          <artifactCategory>category_value </artifactCategory>
          <artifactGroup>group_value </artifactGroup>
          <artifactType>type_value </artifactType>
          <artifactSubtype>subtype_value </artifactSubtype>
          <artifactVersion>version_value </artifactVersion>
</artifactDefinition>
```

Ends with:

```
</deleteArtifactDefinitions>
```

7.1.2.5 Definition Relationship Create (DEFREL - CREATE)

```
<dyn:services type="NFVD" name="DEFREL" action="CREATE">
Service Name: DEFREL
Service Action Name: NFVD
Operation: CREATE
Starts with:

<createRelationshipDefinitions xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
```

```
Language: <languageCode>language</languageCode>
```

List of relationship definitions:

```
<relationshipDefinitions>…</relationshipsDefinitions>
```

Artifact Tag:

```
< relationshipDefinition>…</ relationshipDefinition>
```

Primary combination of values from parent and child:

```
<parentArtifactFamily>parentFamily_value</parentArtifactFamily>
<parentArtifactCategory>parentCategory_value</parentArtifactCategory>
<parentArtifactGroup>parentGroup_value</parentArtifactGroup>
<parentArtifactType>parentType_value </parentArtifactType>
<parentArtifactSubtype>parentSubtype_value </parentArtifactSubtype>
<parentArtifactVersion>parentVersion_value </parentArtifactVersion>
<childArtifactFamily>childFamily_value</childArtifactFamily>
<childArtifactCategory>childCategory_value</childArtifactCategory>
<childArtifactGroup>parentGroup_value</childArtifactGroup>
<childArtifactType>parentType_value </childArtifactType>
<childArtifactSubtype>parentSubtype_value </childArtifactSubtype>

<childArtifactVersion>parentVersion_value </childArtifactVersion>
```

Physical:

```
<isPhysical>true/false</isPhysical>
```

Enabled:

```
<enabled>true/false</enabled>
```

List of categories for relationship:

```
<categories>…</categories>
```

Category relationship (not Artifact):

```
<category>
        <label>category_label</label>
        <version>version_of_category</version>
        <order>1 to n</order>
        <attributes>…</attributes>
        <subcategories>…</subcategories>
<category>
```

It sets the label, version, and order of the category. Two more tags for the attributes and sub categories inside categories are available.

Attributes—Define the label, type, unit, defaultValue, mandatory, and order.

```
<attribute>
        <label>attribute_label</label>
        <type>
                <label>type_label</label>
```

```
                    <descriptionLabel>type_description_label</descriptionLabel>
                    <description>type_description</description>
         </type>
         <unit>unit_name</unit>
         <defaultValue>default_value</defaultValue>
         <mandatory>true/false</mandatory>
         <order>1 to n</order>
</attribute>
```

Subcategories—Same as the categories, repeated inside inner `<categories>` tag.

List of available statuses: `<availableStatus>…</ availableStatus>`

Status label, visible label, and whether it is enabled or not:
```
<status>
         <label>status_label</label>
         <visibleStatusLabel>visible_status_label</visibleStatusLabel>
         <enabled>true/false</enabled>

</status>
```

Ends with:
```
</createRelationshipDefinitions>
```

7.1.2.6 Definition Relationship Update (DEFREL - UPDATE)

```
<dyn:services type="NFVD" name=" DEFREL" action="UPDATE">

Service Name: DEFREL

Service Action Name: NFVD

Operation: UPDATE
```

This operation is the same operation as the Relationship Definition Create, in terms of the request body.

Starts with:
```
<updateRelationshipDefinitions  xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
```

Ends with:
```
</ updateRelationshipDefinitions >
```

7.1.2.7 Definition Relationship Get (DEFREL-GET Test)

Operation for searching.
```
<dyn:services type="NFVD" name="DEFREL" action="GET">
Service Name: DEFREL
Service Action Name: NFVD
Operation: GET
Starts with:
<query xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Language:
<languageCode>language_code</languageCode>
Query criteria: is always the same structure (in all relationships)
<criteriaSet>
          <criteria>
                    <paramName>paramName(ie:type)</paramName>
                    <paramValue>paramValue(ie:contains)</paramValue>
                    <comparator>comparatorType(ie:EQ)</comparator>
          </criteria>
</criteriaSet>
Ends with: </query>
```

7.1.2.8 Relationship Definition Delete (DEFREL-DELETE Test)

```
<dyn:services type="NFVD" name="DEFREL" action="DELETE">
Service Name: DEFREL
Service Action Name: NFVD
Operation: DELETE
Starts with:
<deleteRelationshipDefinitions xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
List of artifact definitions to be deleted:

<artifactDefinitions>…</artifactDefinitions>
```

Specify the relationship that should be deleted.

As mentioned earlier, the combination of Family:Category:Group:Type:Subtype:Version in each artifact (both child and parent) is the only way to specify the relationship.

For example, the relationship between VIRTUAL_MACHINE and VIRTUAL_MEMORY is as follows:

```
<relationshipDefinition>
        <parentArtifactFamily>ie: VIRTUAL_MACHINE</parentArtifactFamily>
        <parentArtifactCategory>ie: GENERIC</parentArtifactCategory>
        <parentArtifactGroup></parentArtifactGroup>
        <parentArtifactType></parentArtifactType>
        <parentArtifactSubtype></parentArtifactSubtype>
        <parentArtifactVersion></parentArtifactVersion>
        <childArtifactFamily>VIRTUAL_MEMORY</childArtifactFamily>
                <childArtifactCategory>GENERIC</childArtifactCategory>
                <childArtifactGroup></childArtifactGroup>
                <childArtifactType></childArtifactType>
                <childArtifactSubtype></childArtifactSubtype>
        <childArtifactVersion></childArtifactVersion>
        <type>ie:USES</type>
</relationshipDefinition>
Ends with: </deleteRelationshipDefinitions>

Running SOAPUi Tests
```

7.1.2.9 Artifact Instance Create (INSART-CREATE Test)

```
<dyn:services type="NFVD" name="INSART" action="CREATE">
Service Name: INSART
Service Action Name: NFVD
Operation: CREATE
Starts with: <createArtifactInstances xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Language of the artifact/s: <language></language>
List Tag (more than one artifact at a time): <artifactInstances>…</artifactInstances>
Artifact Tag: <artifactInstance>…</artifactInstance>
Instance Id (Optional):
<id>ie:1234567890</id>
Primary combination of values (never repeated in definitions):
<artifactFamily>ie:VNF</artifactFamily>
<artifactCategory>ie:GENERIC</artifactCategory>
<artifactGroup></artifactGroup>
<artifactType></artifactType>
<artifactSubtype></artifactSubtype>
<artifactVersion></artifactVersion>
Physical: <isPhysical>true/false</isPhysical>
Enabled:<enabled>true/false</enabled>
List of categories:<categories>…</categories>

Category:
<category>
        <label>ie:GENERAL</label>
        <version>version_of_category</version>
        <order>1 to n</order>
```

```
            <attributes>…</attributes>
            <subcategories>…</subcategories>

<category>
```

Set the label, version, and order of the category. Two more tags are available for attributes and categories inside the categories.

- Attributes—Define label, type, unit, defaultValue, mandatory, and order.

```
<attribute>
        <label>ie:Name</label>
        <type>
                <label>ie:TEXT</label>
                <descriptionLabel>ie:TEXT</descriptionLabel>
                <description>ie:TEXT</description>
        </type>
        <unit>unit_name</unit>
        <defaultValue>default_value</defaultValue>
        <mandatory>true/false</mandatory>
        <order>1 to n</order>
</attribute>
```

Sub categories—They are same as categories repeated inside inner `<categories>` tag.

Status label, visible label, and whether it is enabled or not:
```
<status>
        <label>status_label</label>
        <visibleStatusLabel>visible_status_label</visibleStatusLabel>
        <enabled>true/false</enabled>
</status>

Ends with: </createArtifactInstances>
```

7.1.2.10 Artifact INSTANCE Update (INSART-UPDATE Test)

```
<dyn:services type="NFVD" name="INSART" action="UPDATE">
Service Name: INSART
Service Action Name: NFVD

Operation: UPDATE
```

Required instance ID inside body: <id>1234567890</id>

Not need to specify combination (Family: Category…): search by ID.

Rest of tag structure is like Definition Update.
```
Starts with:
<updateArtifactInstances xmlns="http://www.model.bll.nfv.activator.ov.hp.com">

Ends with: </ updateArtifactInstances>
```

7.1.2.11 Artifact INSTANCE GET (INSART-GET Test)

```
<dyn:services type="NFVD" name="INSART" action="GET">
Service Name: INSART
Service Action Name: NFVD
Operation: GET
```

This is the same operation as Artifact Definition Get but, naturally, instances have one more criteria search: id (represented by <id> tag). It does not need additional explanation.

```
Starts with:
<query xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Ends with: </query>
```

7.1.2.12 Artifact INSTANCE DELETE (INSART-DELETE Test)

```
<dyn:services type="NFVD" name="INSART" action="DELETE">
Service Name: INSART
Service Action Name: NFVD
Operation: DELETE
Starts with:
<deleteArtifactInstances xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Required instance ID inside body: <id>1234567890</id>
No need to specify combination (Family:Category…): search by ID.
Rest of tag structure is like Definition Delete.
Ends with: </deleteArtifactInstances>
```

7.1.2.13 RELATIONSHIP INSTANCE Create (INSREL-CREATE Test)

```
<dyn:services type="NFVD" name="INSREL" action="CREATE">
Service Name: INSREL
Service Action Name: NFVD
Operation: CREATE
Starts with:
<createRelationshipInstances xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Required parent and child instances IDs inside body:
<parentId>1234</parentId><childId>4321</childId>
No need to specify parent and child combination (Family:Category…): search by IDs.
Rest of tag structure is like Relationship Definition Create.
Ends with: </createRelationshipInstances>
```

7.1.2.14 RELATIONSHIP INSTANCE UPDATE (INSREL-UPDATE Test)

```
<dyn:services type="NFVD" name="INSREL" action="UPDATE">
Service Name: INSREL
Service Action Name: NFVD
Operation: UPDATE
Starts with:
<updateRelationshipInstances xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Required parent and child instances IDs inside body:
<parentId>1234</parentId><childId>4321</childId>
No need to specify parent and child combination (Family:Category…): search by IDs.
```

Rest of tag structure is like Relationship Definition Update.

```
Ends with: </updateRelationshipInstances>
```

7.1.2.15 RELATIONSHIP INSTANCE GET (INSREL-GET Test)

```
<dyn:services type="NFVD" name="INSREL" action="GET">
Service Name: INSREL
Service Action Name: NFVD
```

```
Operation: GET
```

This is the same operation as Relationship Definition Get but, naturally, instances have one more criteria search: id (represented by <parentId>, <childId> tags). It does not need additional explanation.

```
Starts with:
<query xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Ends with: </query>
```

7.1.2.16 RELATIONSHIP INSTANCE DELETE (INSREL-DELETE Test)

```
<dyn:services type="NFVD" name="INSREL" action="DELETE">
Service Name: INSREL
Service Action Name: NFVD
Operation: DELETE
Starts with:
<deleteRelationshipInstancesxmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Required instance ID inside body: <id>1234567890</id>
Required parent and child instances IDs inside body:
<parentId>1234</parentId><childId>4321</childId>
No need to specify parent and child combination (Family:Category…): search by IDs.
Rest of tag structure is like Definition Delete.
Ends with: </deleteRelationshipInstances>
```

7.1.2.17 Artifact TEMPLATE Create (TEMART-CREATE Test)

```
<dyn:services type="NFVD" name="TEMART" action="CREATE">
Service Name: TEMART
Service Action Name: NFVD
Operation: CREATE
Starts with:
<createArtifactTemplates="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Artifact Instance Create.
Ends with: </ createArtifactTemplates >
```

7.1.2.18 Artifact TEMPLATE UPDATE (TEMART-UPDATE Test)

```
<dyn:services type="NFVD" name="TEMART" action="UPDATE">
Service Name: TEMART
Service Action Name: NFVD
Operation: UPDATE
Starts with:
<updateArtifactTemplates="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Artifact Instance Update.
Ends with: </ createArtifactTemplates >
```

7.1.2.19 Artifact TEMPLATE GET (TEMART-GET Test)

```
<dyn:services type="NFVD" name="TEMART" action="GET">
Service Name: TEMART
Service Action Name: NFVD
```

```
Operation: GET
Starts with:
<query xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Artifact Instance Get.
Ends with: </ query >
```

7.1.2.20 Artifact TEMPLATE DELETE (TEMART-DELETE Test)

```
<dyn:services type="NFVD" name="TEMART" action="DELETE">
Service Name: TEMART
Service Action Name: NFVD
Operation: DELETE
Starts with:
<deleteArtifactTemplates="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Artifact Instance Delete.
Ends with: </ deleteArtifactTemplates >
```

7.1.2.21 RELATIONSHIP TEMPLATE Create (TEMREL-CREATE Test)

```
<dyn:services type="NFVD" name="TEMREL" action="CREATE">
Service Name: TEMREL
Service Action Name: NFVD
Operation: CREATE
Starts with:
<createRelationshipTemplates="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Relationship Instance Create.
Ends with: </ createRelationshipTemplates>
```

7.1.2.22 RELATIONSHIP TEMPLATE UPDATE (TEMREL-UPDATE Test)

```
<dyn:services type="NFVD" name="TEMREL" action="UPDATE">
Service Name: TEMREL
Service Action Name: NFVD
Operation: UPDATE
Starts with:
<updateRelationshipTemplates="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Relationship Instance Update.
Ends with: </updateRelationshipTemplates>
```

7.1.2.23 RELATIONSHIP TEMPLATE GET (TEMREL-GET Test)

```
<dyn:services type="NFVD" name="TEMREL" action="GET">
Service Name: TEMREL
Service Action Name: NFVD
Operation: GET
Starts with:
<query xmlns="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equals to Relationship Instance Get.
Ends with: </query>
```

7.1.2.24 RELATIONSHIP TEMPLATE DELETE (TEMREL-DELETE Test)

```
<dyn:services type="NFVD" name="TEMREL" action="DELETE">
Service Name: TEMREL
Service Action Name: NFVD
Operation: DELETE
Starts with:
<deleteRelationshipTemplates="http://www.model.bll.nfv.activator.ov.hp.com">
Details are equal to Relationship Instance Delete.
Ends with: </ deleteRelationshipTemplates>
```

Running a Test: Open that test in Soap-UI and click the **Submit** (like play) green button:
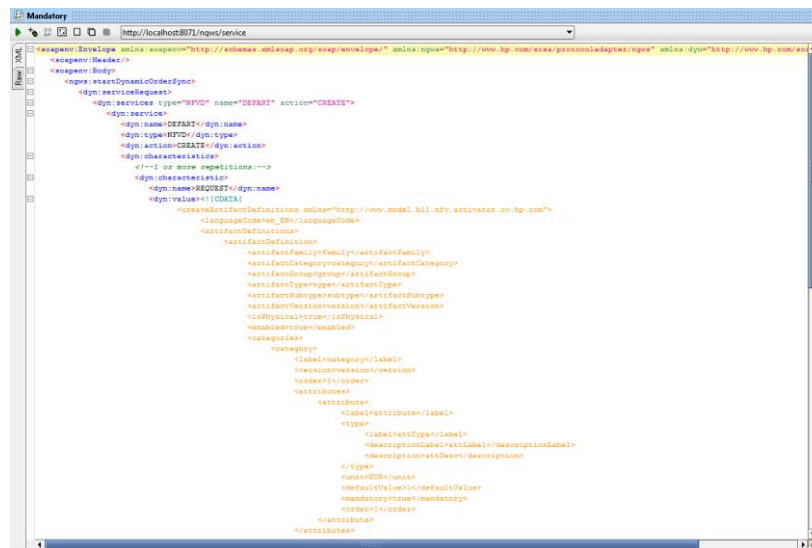
Figure 71: Running a Test

7.2 Northbound Interface – Automation

7.2.1 Scale-out: Create a new VM

Scale-out process takes the policies nodes and creates as many instances of a VM as these policies determine.
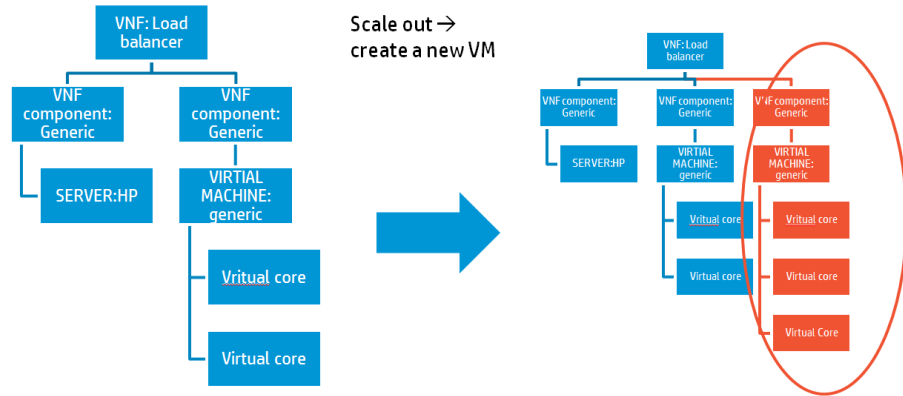
Figure 72: NBI: Scale-Out

The following is a typical Soap call for a scale–out process:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws"
xmlns:dyn="http://www.hp.com/sosa/dynamicserviceorder">

  <soapenv:Header/>

  <soapenv:Body>

    <ngws:startDynamicOrderSync>

      <dyn:serviceRequest>

        <dyn:services type="NFVD" name="INVENTORY" action="SCALE_OUT">

          <dyn:service>

            <dyn:name>INVENTORY</dyn:name>

            <dyn:type>NFVD</dyn:type>

            <dyn:action>SCALE_OUT</dyn:action>

            <dyn:characteristics>

              <dyn:characteristic>

                <dyn:name>ArtifactInstanceId</dyn:name>

                <dyn:value>14032664806161</dyn:value>

              </dyn:characteristic>

            </dyn:characteristics>

          </dyn:service>

        </dyn:services>

      </dyn:serviceRequest>

      <ngws:user>foo</ngws:user>

    </ngws:startDynamicOrderSync>


<ngws:startDynamicOrderAsync><dyn:serviceRequest/><ngws:user/></ngws:startDynamicOrderAsync></soapenv:Body>

</soapenv:Envelope>
```

7.2.2 Scale-in: Delete an Existing VM

Scale-in process takes the policies nodes and deletes as many instances of a VM as these policies determine.
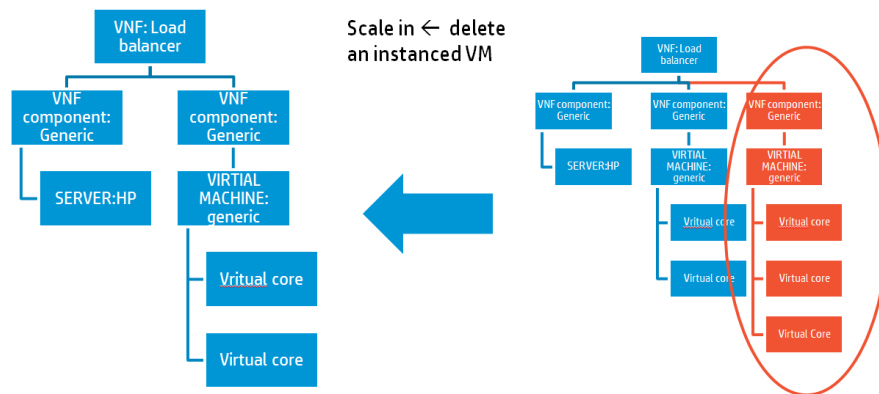
Figure 73: NBI: Scale-In

The following shows a typical Soap call for a scale–in process:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws"
xmlns:dyn="http://www.hp.com/sosa/dynamicserviceorder">

  <soapenv:Header/>

  <soapenv:Body>

    <ngws:startDynamicOrderSync>

      <dyn:serviceRequest>

        <dyn:services type="NFVD" name="INVENTORY" action="SCALE_IN">

          <dyn:service>

            <dyn:name>INVENTORY</dyn:name>

            <dyn:type>NFVD</dyn:type>

            <dyn:action>SCALE_IN</dyn:action>

            <dyn:characteristics>

              <dyn:characteristic>

                <dyn:name>ArtifactInstanceId</dyn:name>

                <dyn:value>14032664806161</dyn:value>

              </dyn:characteristic>

            </dyn:characteristics>

          </dyn:service>

        </dyn:services>

      </dyn:serviceRequest>

      <ngws:user>foo</ngws:user>

    </ngws:startDynamicOrderSync>

<ngws:startDynamicOrderAsync><dyn:serviceRequest/><ngws:user/></ngws:startDynamicOrder
Async>

</soapenv:Body></soapenv:Envelope>
```

7.2.3 Scale Up: Enlarge attributes amount of VM

Scale up process takes the policies nodes and increases the attributes amount as many instances of a VM as these policies determine.

The operation has an impact both in DB and on the Openstack side, changing the VM flavor, according to the attributes set in DB.

The following shows a typical Soap call for a scale up process:

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws">
  <soapenv:Header/>
 <soapenv:Body>
                <ngws:startServiceOrderAsync>
                  <ngws:type>NFVD</ngws:type>
                <ngws:name>VNF</ngws:name>
                        <ngws:action>SCALE_UP</ngws:action>
                        <ngws:inputParams>
                          <!--Zero or more repetitions:-->
                         <ngws:param>
                            <ngws:name>INPUT_INSTANCEARTIFACTID</ngws:name>
                            <ngws:value>14087039824031</ngws:value>
                         </ngws:param>
                         <!--Optional:-->
                         <ngws:param>
                            <ngws:name>INPUT_SCALEALLTREE</ngws:name>
                            <ngws:value>true</ngws:value>
                         </ngws:param>
                          <!--Optional:-->
                         <ngws:param>
                            <ngws:name>INPUT_FORCESTOP</ngws:name>
                            <ngws:value>false</ngws:value>
                         </ngws:param>
                        </ngws:inputParams>
                         <!--Optional:-->
                        <ngws:user>?</ngws:user>
                        <ngws:userId>?</ngws:userId>
                    </ngws:startServiceOrderAsync>
                </soapenv:Body>
</soapenv:Envelope>
```

The "INPUT_SCALEALLTREE" and "INPUT_FORCESTOP" parameters are optional. The predefined values are "true" and "false" respectively.

7.2.4 Scale Down: Reduce attributes amount of VM

Scale down process takes the policies nodes and decreases the attributes amount as many instances of a VM as these policies determine.

The operation has an impact both in DB and on the Openstack side, changing the VM flavor, according to the attributes set in DB.

The following shows a typical Soap call for a scale down process:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws">

  <soapenv:Header/>

 <soapenv:Body>

            <ngws:startServiceOrderAsync>

              <ngws:type>NFVD</ngws:type>

            <ngws:name>VNF</ngws:name>

                   <ngws:action>SCALE_DOWN</ngws:action>

                <ngws:inputParams>

                  <!--Zero or more repetitions:-->

                <ngws:param>

                    <ngws:name>INPUT_INSTANCEARTIFACTID</ngws:name>

                    <ngws:value>14087039824031</ngws:value>

                </ngws:param>

                  <!--Optional:-->

                <ngws:param>

                    <ngws:name>INPUT_SCALEALLTREE</ngws:name>

                    <ngws:value>true</ngws:value>

                </ngws:param>

                 <!--Optional:-->

                <ngws:param>

                    <ngws:name>INPUT_FORCESTOP</ngws:name>

                    <ngws:value>false</ngws:value>

                </ngws:param>

              </ngws:inputParams>

               <!--Optional:-->

              <ngws:user>?</ngws:user>

              <ngws:userId>?</ngws:userId>

            </ngws:startServiceOrderAsync>

          </soapenv:Body>

</soapenv:Envelope>
```

The "INPUT_SCALEALLTREE" and "INPUT_FORCESTOP" parameters are optional. The predefined values are "true" and "false" respectively.

7.2.5 Start Virtual Machine

Start Virtual Machines takes the machines selected and puts them in running state, and it could be checked on DDBB status and CS8 status.

Start Virtual Machine → Start the machines and change the status of them on DDBB and CS8.

VIRTUAL MACHINE

VIRTUAL MACHINE

DataBase status= STOPPED

CS8 status = SHUTOFF

DataBase status= ACTIVE

CS8 status = ACTIVE
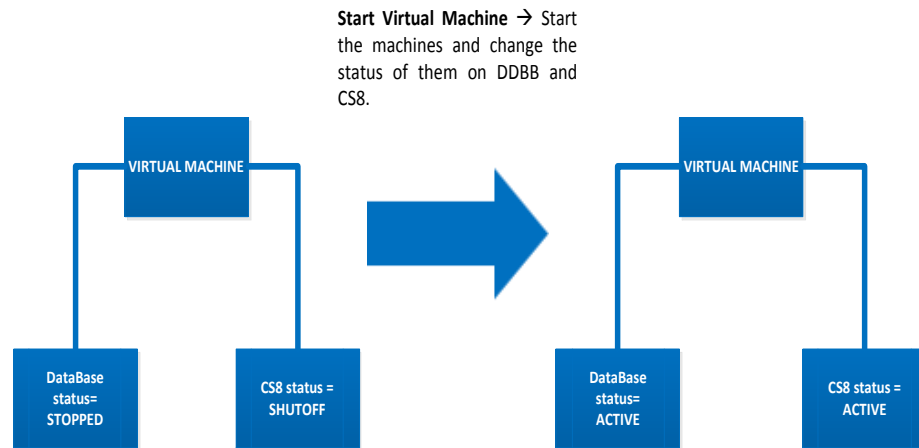
Figure 74: NBI: Start VM

The following shows a typical Soap call for a start virtual machine process:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws">

  <soapenv:Header/>

  <soapenv:Body>

    <ngws:startServiceOrderAsync>

      <ngws:type>NFVD</ngws:type>

      <ngws:name>VNF</ngws:name>

      <ngws:action>START_VM</ngws:action>

      <!--Optional:-->

      <ngws:inputParams>

        <!--Zero or more repetitions:-->

        <ngws:param>

          <ngws:name>INPUT_ARTIFACTTREEID</ngws:name>

          <ngws:value>14093117003051</ngws:value>

        </ngws:param>

      </ngws:inputParams>

      <ngws:user></ngws:user>

      <!--Optional:-->
```

```
<ngws:userId></ngws:userId>

    </ngws:startServiceOrderAsync>

  </soapenv:Body>

</soapenv:Envelope>
```

7.2.6 Stop Virtual Machine

Stop Virtual Machines takes the machines selected and makes them stop, and it could be checked on DDBB status and CS8 status.
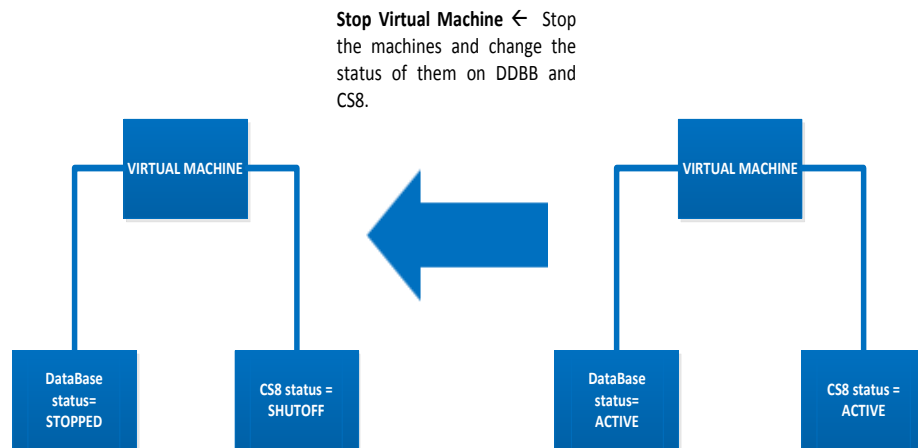
Figure 75: NBI: Stop VM

The following shows a typical Soap call for a stop virtual machine process:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws">

  <soapenv:Header/>

  <soapenv:Body>

    <ngws:startServiceOrderAsync>

      <ngws:type>NFVD</ngws:type>

      <ngws:name>VNF</ngws:name>

      <ngws:action>STOP_VM</ngws:action>

      <!--Optional:-->

      <ngws:inputParams>

        <!--Zero or more repetitions:-->

        <ngws:param>

          <ngws:name>INPUT_ARTIFACTTREEID</ngws:name>

          <ngws:value>14093117003051</ngws:value>

        </ngws:param>
```

```
</ngws:inputParams>
```

```
      <ngws:user></ngws:user>

      <!--Optional:-->

      <ngws:userId></ngws:userId>

   </ngws:startServiceOrderAsync>

  </soapenv:Body>

 </soapenv:Envelope>
```

Chapter 8

Creating VNF

Several processes are involved in creating VNF. After successfully loading a VNF template tree in the NFV Director, perform the following tasks to complete the VNF creation process.

1. Create instances.
2. Assign instances to physical resources.
3. Launch the activation operation.
4. Deploy VNF.

8.1 Creating instances

The VNF template tree should be well-formed with the right relationships between child elements and parents.

Use the following procedure to create instances:

1. Right-click the **VNF:GENERIC** template and select **Create Instance from Template** from the drop-down list.
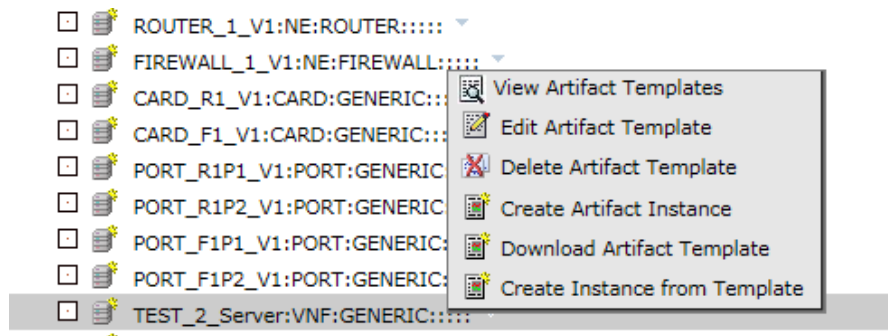
Figure 76: Creating VNF Instance from Template

A form appears on the right-hand side with three editable fields.

Create Instance from Template

Name	Value	Description
Template ID	TEST_2_Server	
Parent Instance ID (Optional)		
Parent relationship type (Optional)		
	OK Reset	

Figure 77: Creating Instance from Template

The first field is automatically filled by the ID of the VNF:GENERIC template. Do not update this box.

The other fields are useful if you want to establish a relationship between the future VNF:GENERIC instance with a parent. Enter the ID in the **Parent Instance ID** and the selected relationship in the **Parent relationship type**.

2. Click **OK**.

   After a short time, the instance tree is created.

When creating instances, the following two additional operations are involved:

- Assignment

- Validation

The artifact template tree can include three types of policies, each one with its special functionality:

- POLICY:ENTITY_ASSIGN

- POLICY:VALUE_VALIDATION

- POLICY:ENTIY_RANGE

The first additional operation sets the attributes of an instance indicated by the ENTITY_ASSIGN template. The second additional operation validates those attribute values against the policy VALUE_VALIDATION. If the validation is incorrect, the instances cannot be created.

One thing to keep in mind is that when a VNF is created from a template, it stores the template ID (in an internal field) that is used. If the creations were triggered from the end to end, the creation stores the assignment tree and the resource tree that were used as well (these IDs are stored in a special category). The assignment tree and the resource tree used are stored only for the VNF artifact.

WARNING: When an instance is being activated, only the VMs are created. These VMs are attached to a network that already exists on the VIM. The VMs that are in instantiated status are the ones that are activated. Deactivating process is similar, taking only VMs which statuses are not instantiated.

8.2 Generating a template

The VNF template should be well-formed with the policies for the additional operations to be successful. This section discusses the structure of the template and how to load the template in the NFV Director.

The following illustration explains the tree with the relationships between child components and parents.
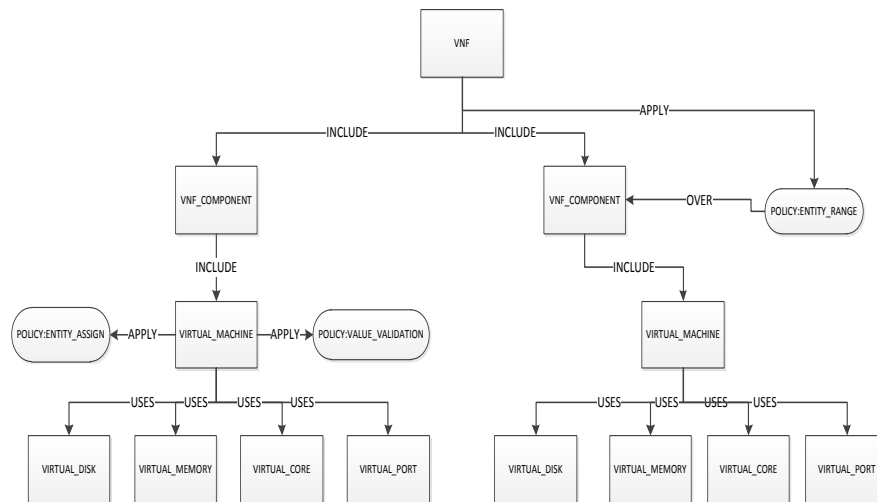
Figure 78: VNF Template example

When the VNF template tree is formed in the XML, you can load it in two ways.

1. Right-click the **Artifact Templates** and select the **Upload Artifact Templates** from the drop-down list.
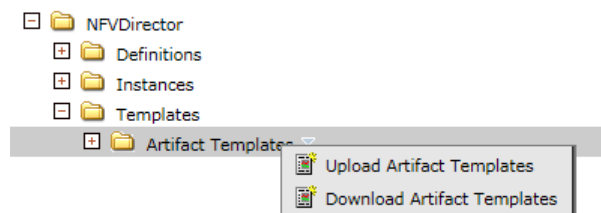
Figure 79: Upload Artifact Templates

The Upload Artifact Template window appears.
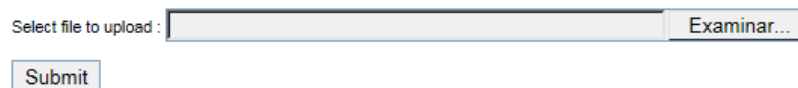
Figure 80: Upload Artifact Template window

2. Click the **Examinar…** button, select the XML file, and click the **Submit**.

    The other way to upload is through SOSA using the soapUI software.

8.3 Policies

This section discusses the policies and how to set the attributes for running the additional operations when creating the VNF.

Three kinds of policies are available.

- Assignment
- Validation
- Range

8.3.1 Assignment

Assignment is the additional operation linked with the POLICY:ENTITY_ASSIGN. In this task, you can set the attribute values before creating the instance.
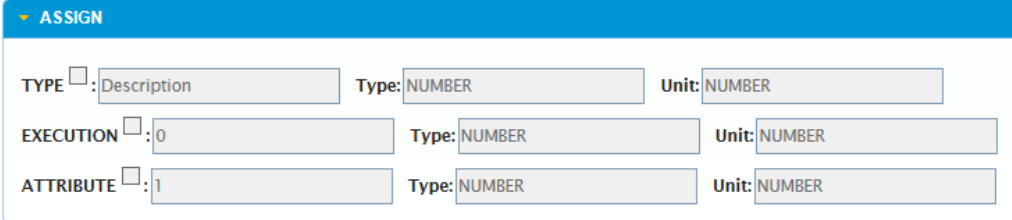
You can assign the policy in three ways:

- Java method
- Script method
- Workflow method

To select the assigning method, configure the POLICY:ENTITY_ASSIGN properly. This policy has a category named **ASSIGN** with three attributes:

- TYPE
- EXECUTION
- ATTRIBUTE

The assignment mode is decided on the basis of values you enter for the attributes.

Figure 81: Policy Assignment mode

Set the policy for each one of the three modes.

8.3.1.1 Java method assignment

TYPE = JAVA

EXECUTION = Complete path of the Java method.

Example:

If you want to assign the attributes **Speed** and **Amount** belonged to INFO category, using the method, assign to the following JavaMethodUtils class. The assignment must be self-programmed in Java:

```
com.hp.ov.activator.nfv.nodes.JavaMethodUtils.assign(INFO.Speed
,INFO.Amount)
```

Note

Write the attributes to be assigned between the brackets conforming to the Java method name. The format for the attributes is `Category.Attribute`.

8.3.1.2 Script assignment

TYPE = SCRIPT

EXECUTION = Complete path of the Script.

For example, `com.hp.ov.activator.nfv.name_script.sh`

ATTRIBUTE: `Category.Attribute`

For example, if the policy applies to a VIRTUAL_CORE and you want to assign the attribute Speed, and the attribute belongs to the INFO category, enter the following:

```
ATTRIBUTE: INFO.Speed.
```

The script must return an integer that can be assigned to the attribute. The script must also be self-programmed.

8.3.1.3 Workflow assignment

TYPE = WORKFLOW

EXECUTION = Name of the workflow to be launched.

ATTRIBUTE = Category.Attribute

The workflow must be self-built.

8.3.2 Validation

The other additional operation provides the possibility to validate the attribute values. Similar to the assignment task, you can validate in 4 ways:

- Range validation

- Java method

- Script method

- Regular Expression

The POLICY:VALUE_VALIDATION should be configured properly. The **destiny** value, which is mandatory variable, should be set in the attribute **ARTIFACT_CATEGORY_ATTRIBUTE_TARGET**. To set it, the format of the attribute should be `Category.Attribute`. For VIRTUAL_MACHINE, a sample format is `INFO.Speed`.

Figure 82: Policy Validation

After setting the attribute values, validate this attribute according to the policy mode specified in next type. To select the validation mode, set the appropriate fields in the next order.

Each policy validates and if any validation fails, creation fails and only a database rollback is performed.

8.3.2.1 Range validation

Range validation method takes the value of the artifact destiny. It checks if the value is greater than or equal to the policy validation MIN_VALUE and also whether the value is lower than or equal to the policy validation MAX_VALUE. You should define the values in integers (1,2,3, and so on).

For example:

MIN_VALUE=1

MAX_VALUE=3

In this example, in VIRTUAL_MACHINE, if the value of `INFO.Speed` is not greater than or equal to 1 and if it is not lower than or equal to 3, the validation fails.

8.3.2.2 Java method validation

This method takes the value of the artifact **destiny** to validate the content of the attribute Class of the policy. The content must be the fully qualified name of the class and the name of the method to execute using the corresponding parameters.

`FullyQualifiedClassName.methodName (Param1, Param2, …)`

For example:

`com.hp.ov.activator.nfv.nodes.JavaMethodUtils.imprime (INFO.Speed,INFO.Amount)`

The validation must be self-programmed in Java.

8.3.2.3 Script validation

This method uses an external script. The input contains the full path to the script and the attribute to be validated that is located in the instance. The script returns an integer with number 1 to check the correct execution. Modify the last script line with the following code:

```
VAR_SCRIPT_RESULT=1"
```

This code line sends a correct value to a case packet variable for checking the correct execution of the script.

For example, `com.hp.ov.activator.nfv.name_script.sh`

Validation must be self-programmed into the script.

8.3.2.4 Regular Expression

This method checks the content of the value located in the **destiny** variable with a pattern defined in the attribute REGULAR_EXPRESSION.

For example, if you want to check whether the attribute value contains a character, enter the character in the REGULAR_EXPRESSION field.

This mode of validation happens only when you complete another one of the modes of validation. The order to check the modes is the following:

1. Range validation
2. Java method
3. Script method
4. Regular Expression

8.3.3 Range

You can add another policy (POLICY:ENTITY_RANGE) to the **Create Instance from Template** operation. This policy allows the opportunity to create more than one instance in a single time of concrete artifacts.

8.3.3.1 Configuring the policy

The DEFAULT_SCALE_OUT indicates the number of instances to create. The MAX attribute controls the maximum number of instances that the NFV Director can have.

Figure 83: Policy Range

For example, if the policy is applied over a VNF_COMPONENT with two instances, the DEFALUT_SCALE_OUT = 2, and MAX = 3, the policy can create only one instance as the maximum is 3.

8.4 Virtual Machines and VNF lifecycle

This section explains the Virtual Machines lifecycle. The different status on Virtual machines can be checked on status field on NFVDirector interface and CS8 interface.

8.4.1.1 Start Virtual Machine

The initial status is:

VNF→ENABLE

VIRTUAL MACHINE → STOPPED/SHUTOFF

MONITOR → STOPPED

First step:

VNF→LOCKED

VIRTUAL MACHINE → STOPPED/SHUTOFF

MONITOR → STOPPED

Second step:

VNF→LOCKED

VIRTUAL MACHINE → ACTIVE/ACTIVE

MONITOR → STOPPED

Third step:

VNF→LOCKED

VIRTUAL MACHINE → ACTIVE/ACTIVE

MONITOR → STARTED

Final status is:

VNF→ENABLE

VIRTUAL MACHINE → ACTIVE/ACTIVE
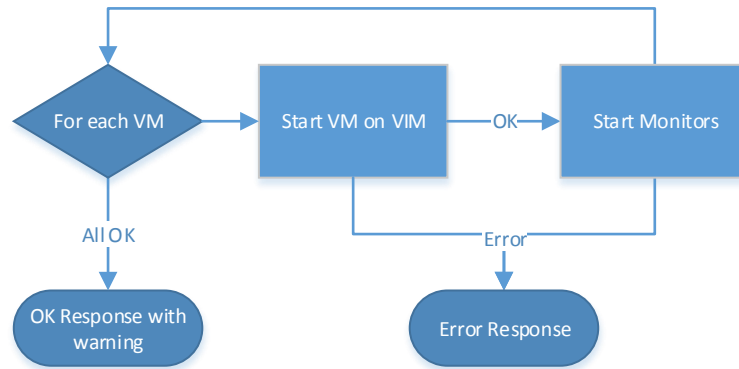
MONITOR → STARTED

Figure 84: Procedure for Start virtual machine

8.4.1.2 Stop Virtual Machine

The initial status is:

VNF→ENABLE

VIRTUAL MACHINE → ACTIVE/ACTIVE

MONITOR → STARTED

First step:

VNF→LOCKED

VIRTUAL MACHINE → ACTIVE/ACTIVE

MONITOR → STARTED

Second step:

VNF→LOCKED

VIRTUAL MACHINE → ACTIVE/ACTIVE

MONITOR → STOPPED

Third step:

VNF→LOCKED

VIRTUAL MACHINE → STOPPED/SHUTOFF

MONITOR → STOPPED

Final status is:

VNF→ENABLE

VIRTUAL MACHINE → STOPPED/SHUTOFF

MONITOR → STOPPED
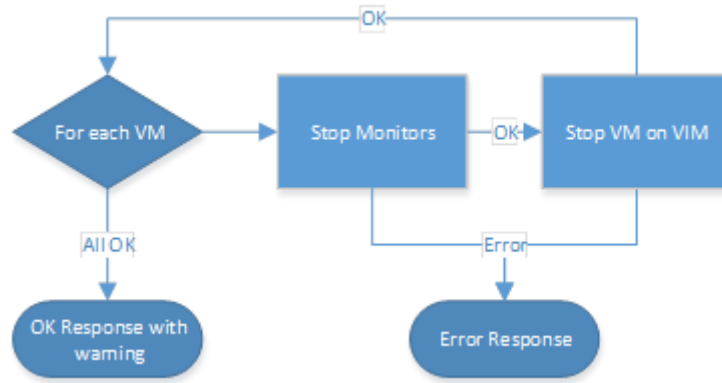
Figure 85: Procedure for Stop virtual machine

8.4.1.3 Scale Up/Down
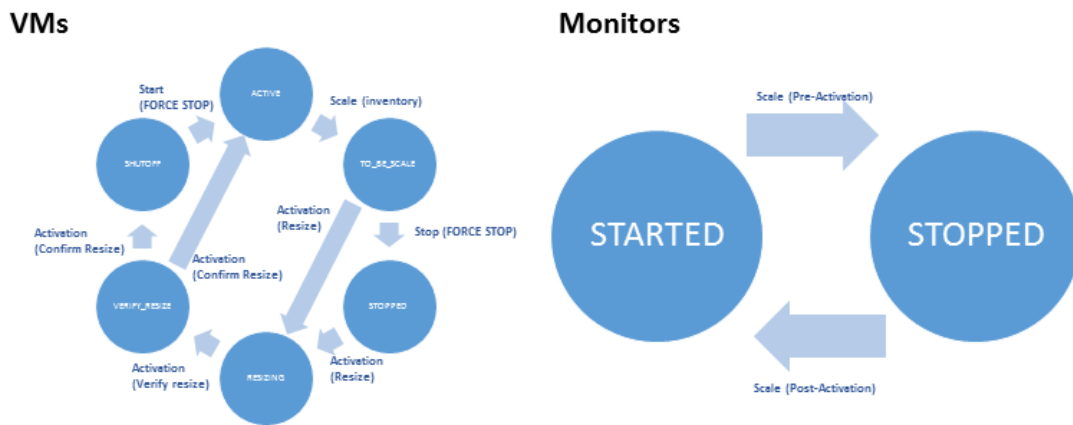
Figure 86: Virtual machine lifecycle
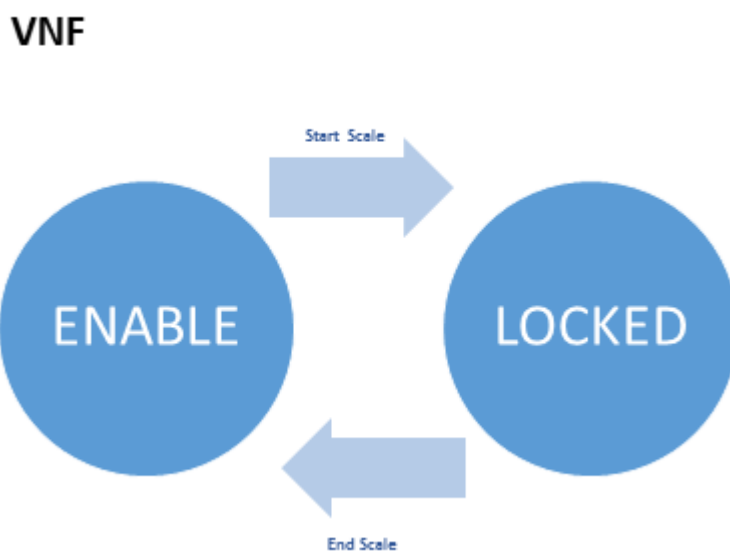
Figure 87: VNF lifecycle

8.5 Examples

This section explains two examples of the procedure to create VNF.

8.5.1 Example with Range Policy
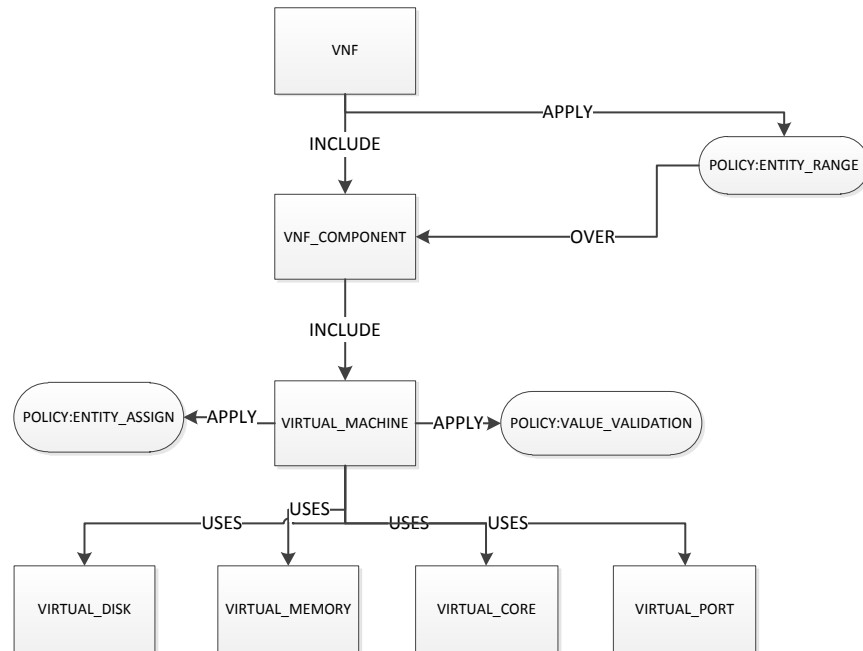
Figure 88: Template example with range policy

8.5.1.1 POLICY:ENTITY_RANGE parameters

Figure 89: POLICY:ENTITY_RANGE parameters

8.5.1.2 Instances
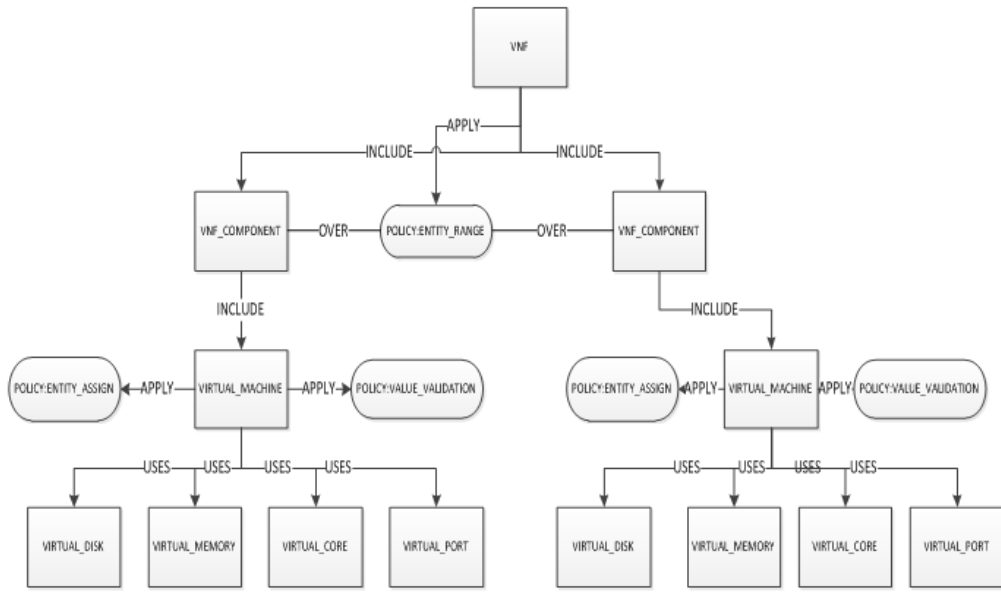
Figure 90: Instance result of template example with range policy

Doing the operation in the VNF and with the parameters set this way:

DEFAULT= 1, MAX= 5,

A VNF is generated with 2 VNF_COMPONENT as child components of the VNF.

If the VNF_COMPONENT does not have the POLICY:ENTITY_RANGE, the same instances tree is generated that is mentioned in the templates.

Note
You can create an instance only if the validation is correct.

8.5.2 Example with Assign Policy

8.5.2.1 POLICY:ENTITY_RANGE parameters for assign policy

Figure 91: Example Policy Assignment

8.5.2.2 Instances
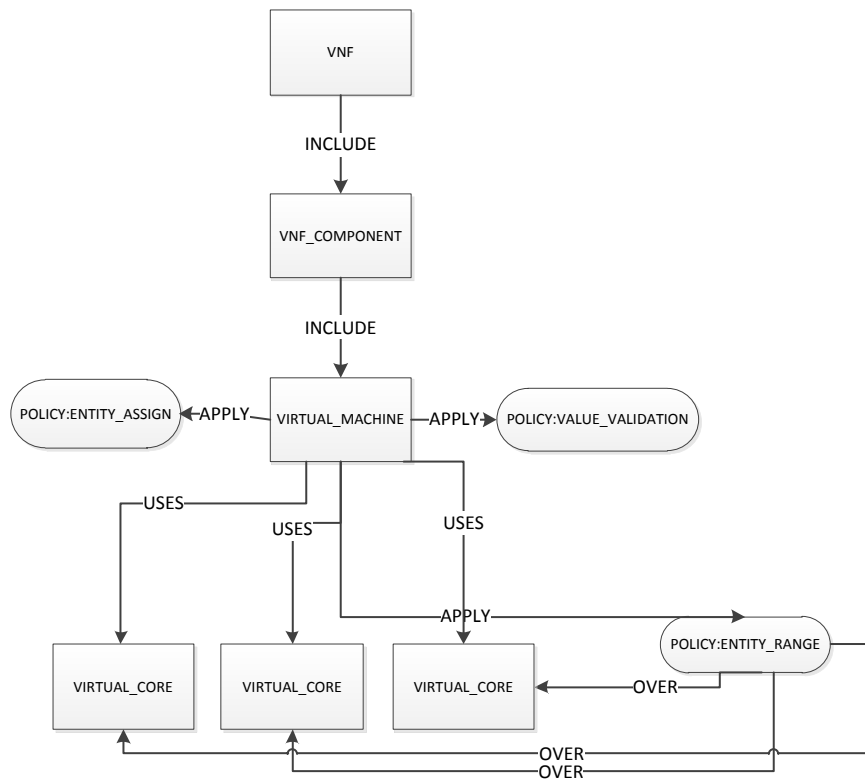
Figure 92: Instance result of Template example with assign policy

In this case, the policy creates two VIRTUAL_CORE instead of 4, because the MAX is 3.

VNF Resource assignment

9.1 Assignment Process

The Assignment process is the second major process involved in the global creation process. This process creates relationship between elements of instance tree created during instantiation and a pool of resources, which is another artifact tree. To create relationship between elements, you should define the elements the artifact instance consumes and the resources to which these instances can be allocated.

The instances tree and resources tree are artifact instances, which can be defined as:

- Artifacts instance tree—An instance tree created from artifact tree template.

- Resources tree—An instance tree used as resources to be consumed by the artifact instance tree.

9.2 Policies

The following sections describe the policies involved in resource assignment process.

9.2.1 Assignment relationship

Assignment relationship is defined as a hierarchical tree of relationship to assign instances to resources. The following illustration provides an overview.
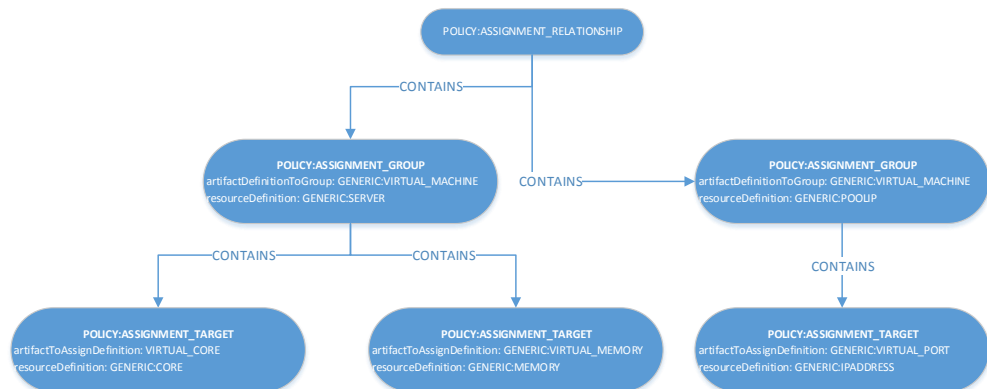
Figure 93: Policy Assignment Relationship hierarchical tree

9.2.1.1 ASSIGNMENT_RELATIONSHIP

This artifact is only a reference to grouping different kinds of relationships that you want to create. This artifact is the parent of elements that contain and define the relationship that you want.

Figure 94: Policy Assignment Relationship

It can have only one type of child: ASSIGNMENT_GROUP.

9.2.1.2 ASSIGNMENT_GROUP

This artifact represents a logical association between equivalent elements from instances and resources. For example, a virtual machine might be allocated inside a physical server. This representation does not involve a creation of a final relationship in the database and is meant only as a reference.

The representation is used to reduce and group final resource targets for low level instances. For example, when you define an assignment group between virtual machines and physical servers, you are defining that all elements of a virtual machine including vCore or memory (detailed and defined in ASSIGNMENT_TARGET policy) must have a direct and real relationship stored in the DB with resources under physical server artifact (core, memory, and so on).

This policy has an implicit validation. The resource candidate must have enough capacity to hold the instance. In other words, the final target resource candidate amount (cores, memory) must be greater than the artifact instance amount (vCores, memory) that you want to allocate on it.

Figure 95: Policy Assignment Group

Attributes under the Group category are the following:

- artifactDefinitionToGroup: Definition of the instances that must be related and in the NFVD Format (<Family>:<Category>:<Group>:<Type>:<SubType>:<Version>).

- resourceDefinition: Definition of the resources where the instances are related and in the NFVD Format (<Family>:<Category>:<Group>:<Type>:<SubType>:<Version>).

It can have only one type of child: ASSIGNMENT_TARGET.

9.2.1.3 ASSIGNMENT_TARGET

This policy identifies the relationship that the assignment process creates in the database. The policy defines the final elements that are related between, for example, vCores and physical cores or virtual memory and physical memory.

The assignment process gets all instances from the instances tree defined by the artifactDefinitionToGroup attribute in the Assignment group policy and a valid resource from the resources tree defined by the resourceDefinition attribute in the Assignment group policy.

Then it creates each relationship defined in the assignment targets.

Figure 96: Policy Assignment Target

Attributes under TARGET category are the following:

- artifactDefinitionToGroup: Definition of the instances that must be created for the relationship and in the NFVD Format
  (<Family>:<Category>:<Group>:<Type>:<SubType>:<Version>).

- resourceDefinition: Definition of the resources where the instances are related and in the NFVD Format
  (<Family>:<Category>:<Group>:<Type>:<SubType>:<Version>).

9.2.2 Over_subscription

All instances count as 1 as amount, unless it has an attribute name `INFO.Amount` that indicates a different amount.

Some resources around virtualization can have over-subscription to allocate more instances that the amount permits. To do so, you can define an over-subscription policy attached on that resource.

Figure 97: Policy Over Subscription

Attributes under OVER_SUBSCRIPTION category are the following:

- Rate—Ratio of over-subscription. The final amount is calculated as INFO.Amount per Rate. For example, a core with amount 8 and over-subscription rate as 2, has a final amount of 16.

- RELATIONSHIP_TYPE—Not used on this release. It indicates the relationship of over-subscription.

- MIN—Not used on this release.

- MAX—Not used on this release.

An over-subscription policy can be attached over any artifact that is defined. If this policy is included in a template, you should create a relationship, because a parent artifact template is instantiated.

9.2.3 Affinity

Sometimes you may have to allocate some instances over the same resource (two virtual machines in the same location) or these instances need to share another resource (some ports over same network). In these scenarios, you should define an Affinity policy that applies over some instances and share the same final resource.

Affinity policy works like an Assignment relationship policy, where you designate which particular elements to get together. For those instances, the same rules apply as that of the assignment policies.

Assignment policies are general rules to create relationship. For all artifacts defined for group policy, their target relationship policies are applied. Otherwise, you can define different affinity policies to group concrete artifacts and apply only on the subset of its elements.

For example, a general policy is available to assign virtual cores and memory over physical core and memory and virtual ports to IP address on a server. You can define an affinity policy because you need some of the virtual machines to be created over the same location (only applies to core and memory, but not the relationship between ports and IP address).

Figure 98: Policy Affinity

Attributes under AFFINITY category are the following:

- Type

    - MUST—If the resource is not a valid resource to allocate all artifacts, the process returns an error.

    - SHOULD—If the resource is not valid, the assignment ignores the affinity policy.

- AFFINITY_TARGET—Definition of the resources which the affinity instances are related and in the NFVD Format (<Family>:<Category>:<Group>:<Type>:<SubType>:<Version>).

- FINDMETHOD—Not used in this release. This attribute contains a search method to narrow the entire list of resources.

- LEVEL—Not used in this release.

9.2.3.1 Relationship

If this policy is included in a template, you should create a relationship of the APPLY type, because a parent artifact template should be instantiated. You should have defined it earlier.

- Must have at least one relationship of the type APPLY over another artifact (group of affinity). It must be defined earlier.

- Must have at least one relationship of type CONTAINS over a policy ASSIGNMENT_GROUP.
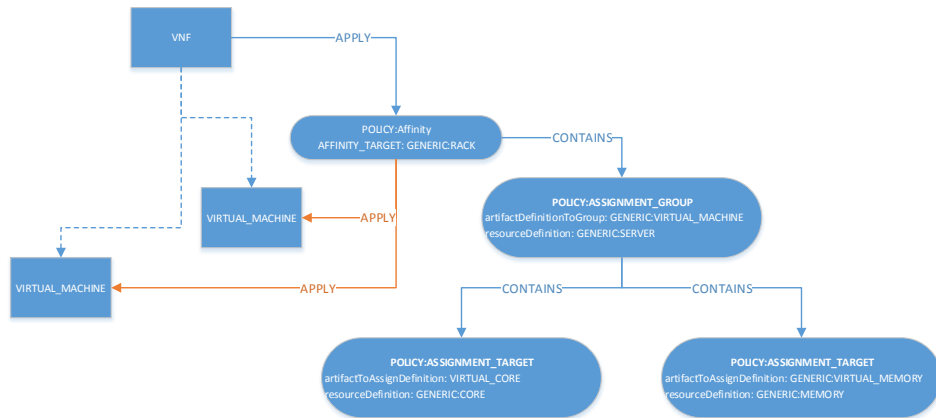
Figure 99: Policy: Apply Relationship

9.3 Process Description

This section provides an overview of the assignment process, how to apply the policies listed in the previous sections, and how final relationships are created.

This process needs the following tree input parameters:

- artifactTreeID—Parent artifact ID of instances tree that needs to be allocated.

- resourceTreeID—Parent artifact ID of resources tree which can allocate instances.

- assignmentRelationshipID—Assignment relationship ID.

Use the following procedure to assign the affinity policies:

1. Assign the artifacts that have affinity policies associated.

   The process queries all affinity policies and their groups and target of assignment.
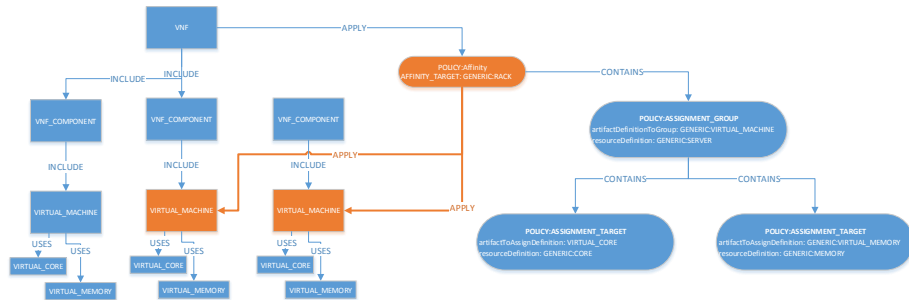
Figure 100: Policy Assignment process

For each affinity policy found:

- It queries all affine artifacts and calculates total amount of all policy targets defined in the affinity policy tree.

- It queries the resource tree for all resource candidates identified by the AFFINITY_TARGET attribute on the Affinity current policy. For each resource candidate, the free amount of current candidate is compared with total amount of all affine artifacts.

  If this resource can contain all artifacts, it proceeds to assign these affine artifacts to this resource (following the group and target policies). If not, it tries with the next resource candidate.

102
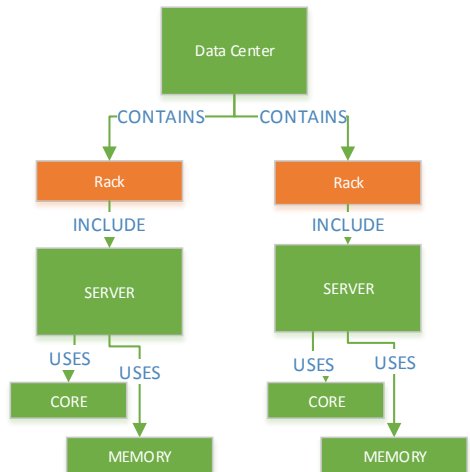
Figure 101: Policy Assignment Flow

Each assignment group of affinity policy is queried for all resources (children of affinity resource candidate) defined in the resourceDefinition attribute and all artifacts instances identified by the artifactDefinitionToGroup attribute.
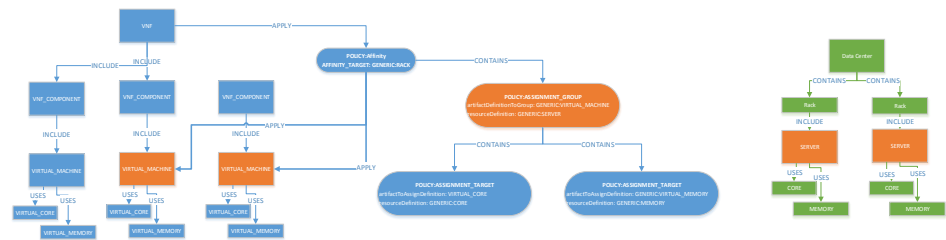
Figure 102: Policy Assignment Group

2. For each artifact and resource queried by the group policy, all target policies are queried.
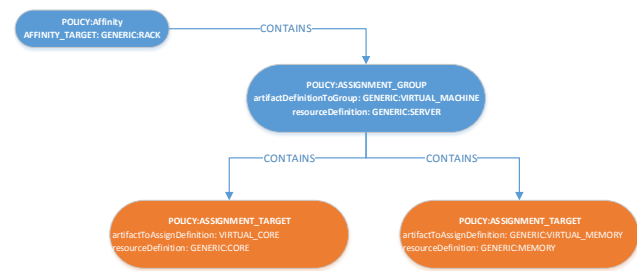
Figure 103: Query Target Policies

3. Query again for artifacts, including all child elements defined for each assignment target policy.
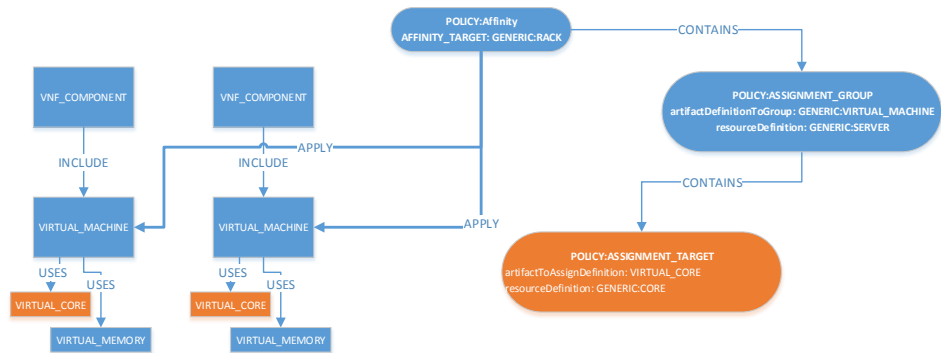
Figure 104: Query artifact for assignment target policy

4. Perform the same steps with resources.
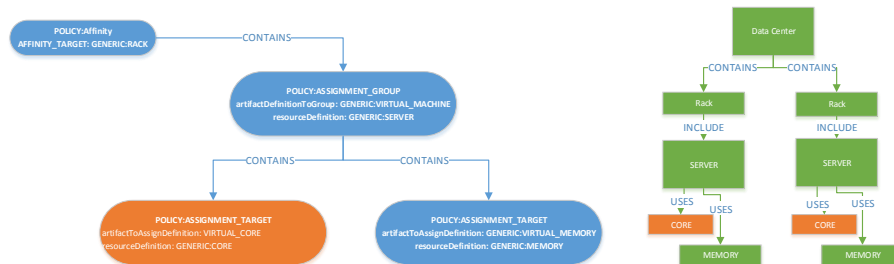
Figure 105: Query Resources for assignment target policy

5. Final instances and resources of each artifact are placed in the descending order and resources are placed in ascending order.

6. Assign if the resource allows an artifact.
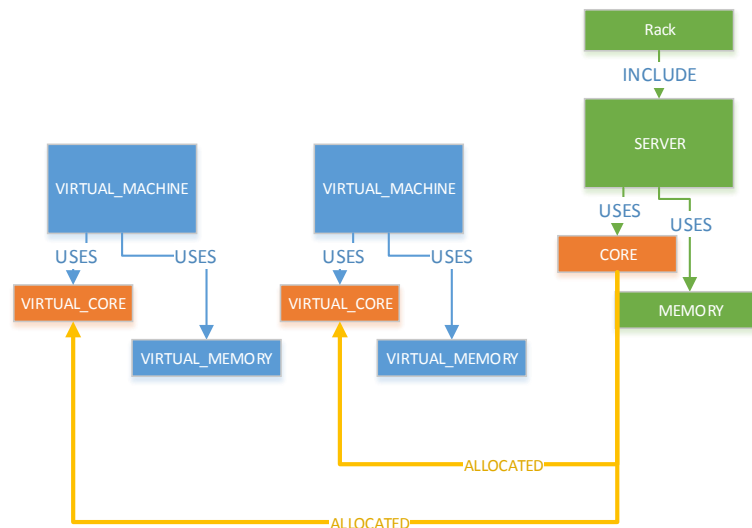
Figure 106: Assign Resource Artifact

Note

If the relationship already exists, it is assumed that it was created in the previous affinity of assignment process, but not reported as an error.

7. When all affinity policies are processed, a generic assignment process is launched, following similar steps as that of affinity, but on the resource tree.

For each assignment group policy, all resources defined in the resourceDefinition attribute and all artifacts instances identified by the artifactDefinitionToGroup attribute are queried.
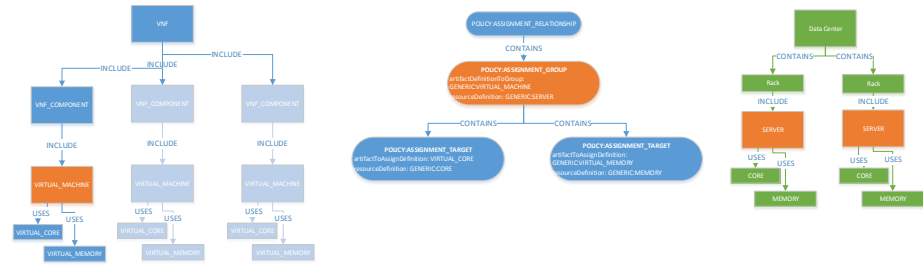
Figure 107: Query resources in Assignment Group policy

8. Each artifact and resource is queried for target policies and group policies.
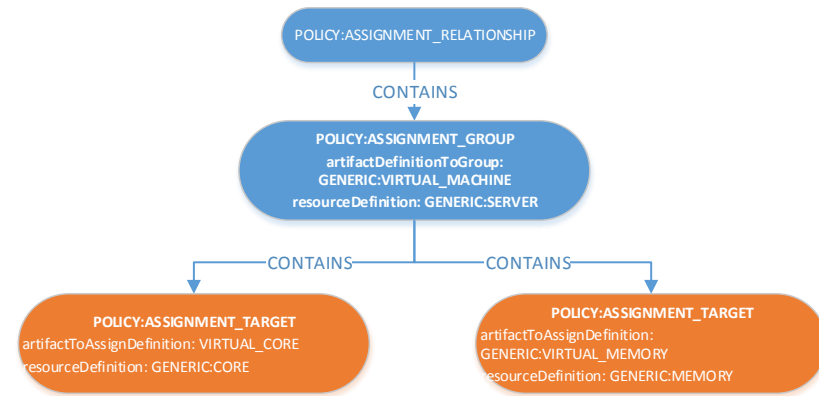
Figure 108: Query Target policy and Group policy for artifact and resource

9. Query again over artifact all children defined for each assignment target policy.
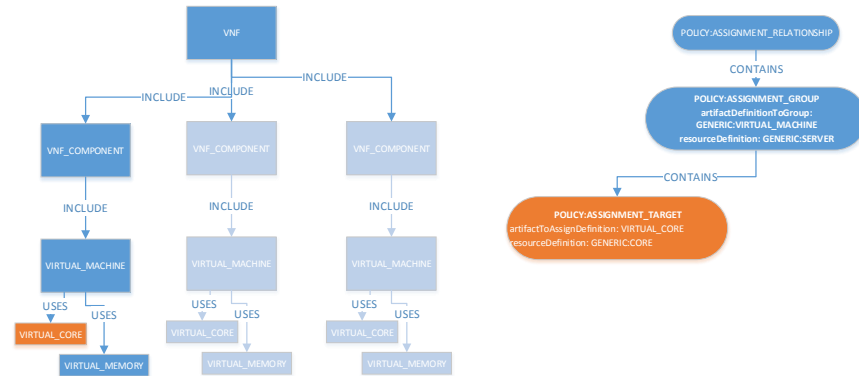
Figure 109: Query Assignment Target Policy for artifact children

10. Perform the same steps with resources.
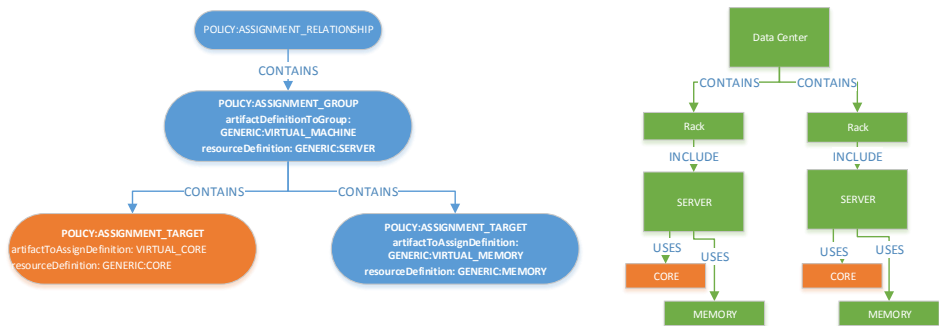
Figure 110: Query Assignment Target Policy for Resource Children

11. Final instances and resources of each artifact it will be ordered from greater to lower and resources will be ordered from lower to greater, and try to assign if the resource can allow artifact.
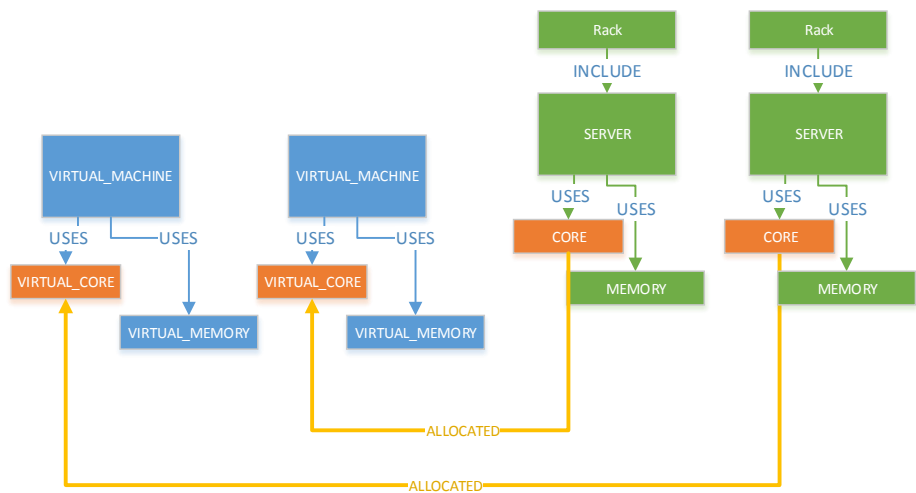
Figure 111: Assign Artifact to Resource

Note

If the relationship already exists, it is assumed that it was created in the previous affinity of assignment process, but not reported as an error.

VNF Scaling

10.1 Scale-in

The scale in operation is built to decrease, assign, and activate resources on the instance tree. For example: one virtual machine with 2 children (Virtual Cores).

Actually this operation can be called only from a VNF artifact.
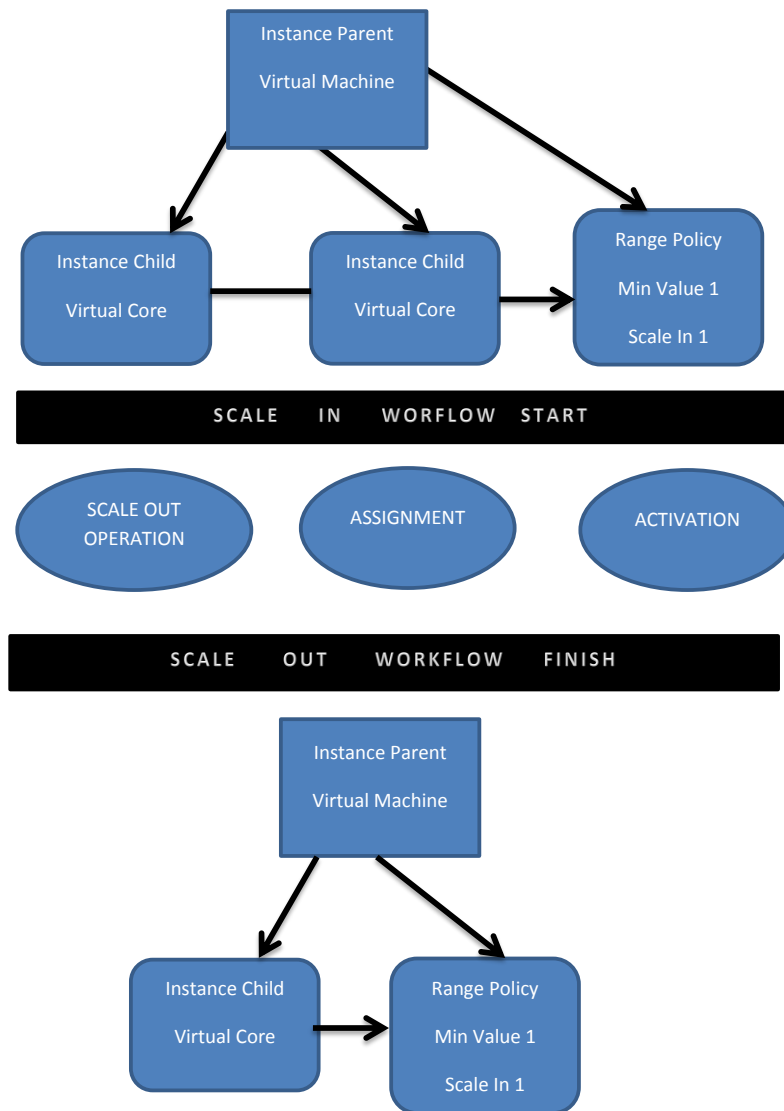
Figure 112: VNF Scale In

You can decrease the number of Virtual Cores according to the policy range using the scale-in operation. The Scale-In operation is also responsible for un-assigning and activating the resources.

For more information about Policy Range, refer to the *8.3 Policies* section.

Particular Case:

When scale in workflow is called, it tries to scale as many elements as possible. If any element tries to scale below the minimum, it does not scale. However, the workflow continues scaling other components and displays a warning.

For example, consider a scenario where a 2 Virtual machine VNF starting as:

VM1 = 5 instance, VM2 = 5 instance, where

VM1 default=5, scale in=5, min= 2,

VM2 default=5, scale in=1, min= 1

If it tries to scale, the result is VM1 = 5 instance, VM2 = 4 instance 1, because VM1 cannot scale below the minimum but VM2 can.

10.1.1 Launching the Workflow

1. Select **Instance** > **Artifact Instances**.

2. Right-click the instance template and select the **Scale In** option.
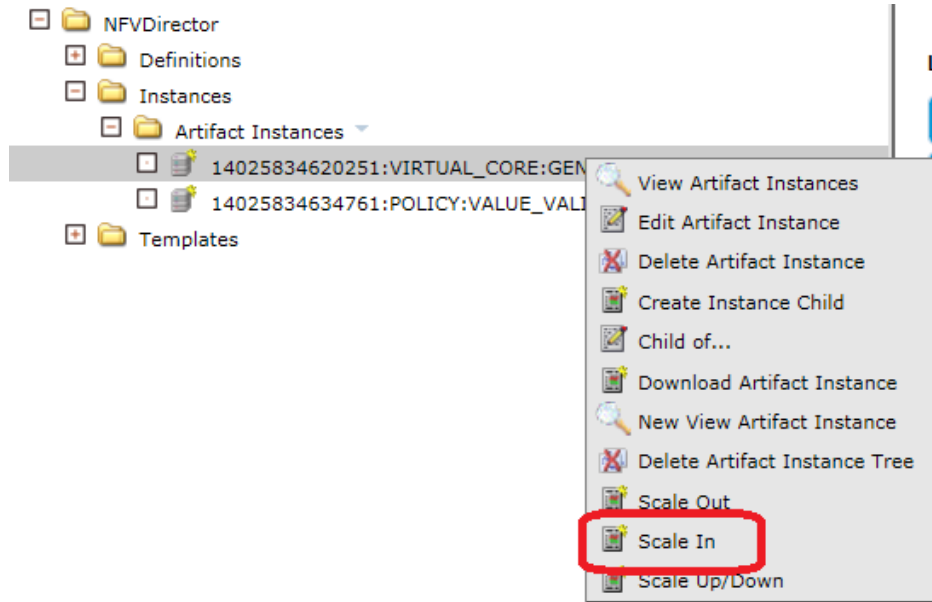
Figure 113: Scale In: Workflow launch

3. Check the Instance ID.
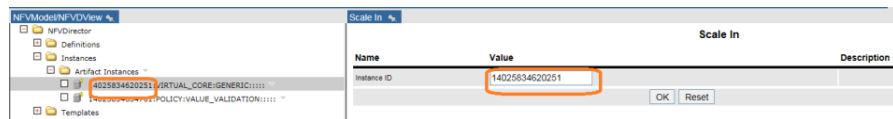
Figure 114: Scale in: check Instance ID

4. Click the **OK** button to scale.
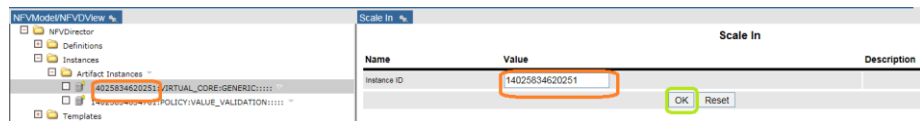
Figure 115: Scale In: confirm operation

The workflow is launched and the scale is done.

10.1.2 Scale in operation

Scale in operation is in charge of decreasing the amount of resources.

It starts checking the Instance ID and getting the artifact associated to this ID. After getting the artifact the operation continues getting the template associated with the instance.
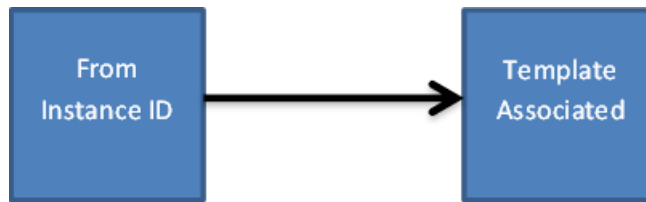
Figure 116: Scale In : get associated template

With the instance located the next step is to get all template children and check one by one whether it is a policy or not.

Figure 117: Scale In: get all template children

If it is not a policy, the template Id is gotten and the scale in operation is called to apply it recursively.
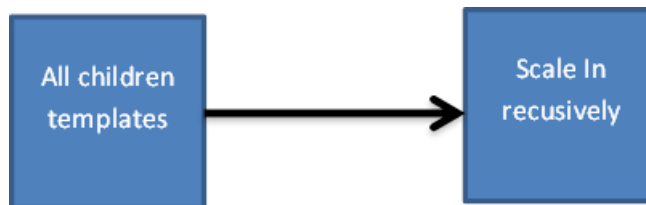
Figure 118: Scale In: Perform operation recursively

If it is a policy, the template id is gotten and all the children to get all the policies are called.

After policies evaluated and confirmed the process continues getting all the relationships between templates parents and children and calling create instance from template to create the new instances.
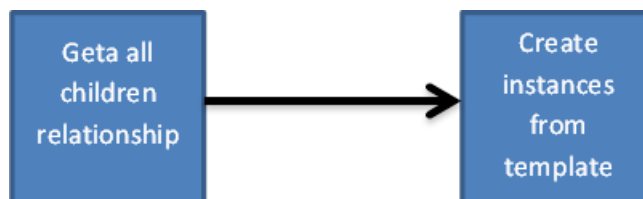
Figure 119: Scale In: Create new instance from template

Later, the operation gets all the relationships and checks the minimum range value of the scale in.

After that the scale in operation is finished.

Figure 120: Scale In: Check policy

10.1.3 End-to-End Example

A Policy Node is available with DEFAULT=4, MIN=2 and DEFAULT_SCALE_IN=2. It also has OVER Parent Relationship to VIRTUAL_CORE.
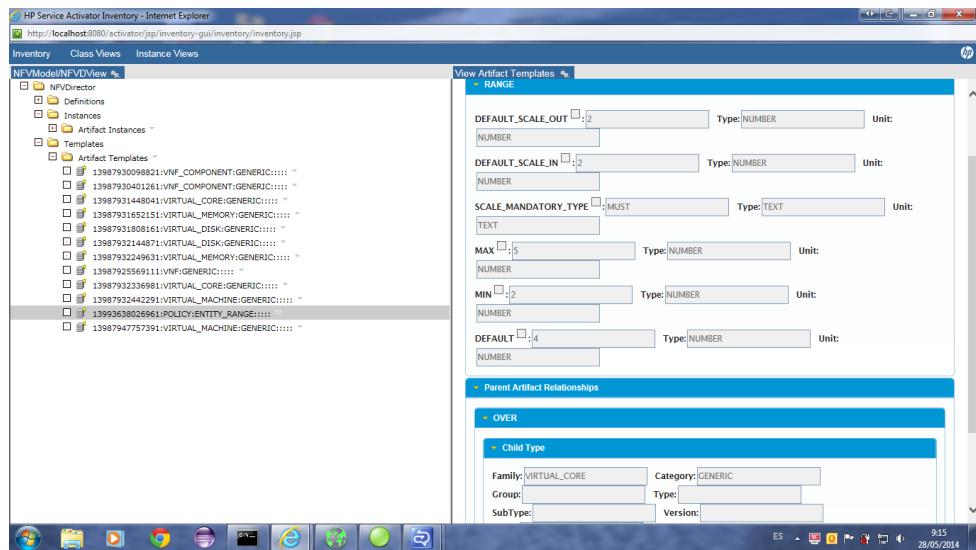
Figure 121: Scale In Example: define policy

The Artifact Template Tree is available with these relationships.
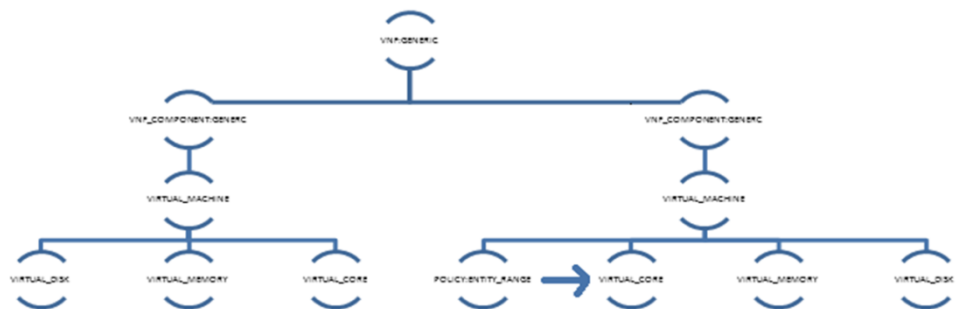
Figure 122:Scale In Example: Artifact Template Tree

1. Create instance from the desired template.
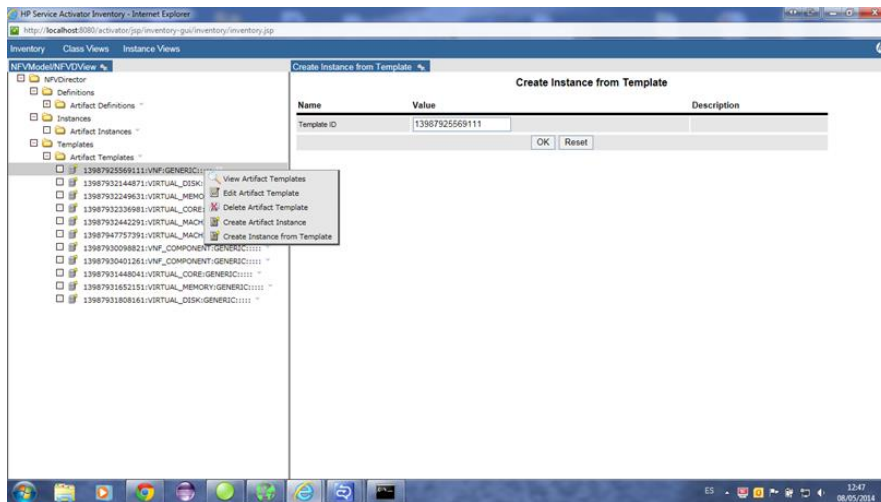
Figure 123: Scale In Example: Create Instance from Template

2. Create the same Artifact Instance Tree with the same relationships.

3. Create 4 VIRTUAL_CORE Instances as policy index.
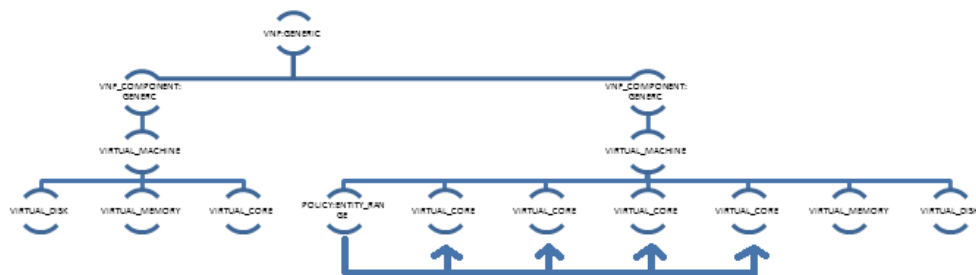
Figure 124: Scale In Example: Create Virtual Core Instance

The DEFAULT_SCALE_IN parameter is the number of instances you want to delete (scale).

The MIN parameter is the minimum of instances you must have.

You can get the Scale In operation if: MIN>=Instance amount in BBDD - DEFAULT_SCALE_IN.
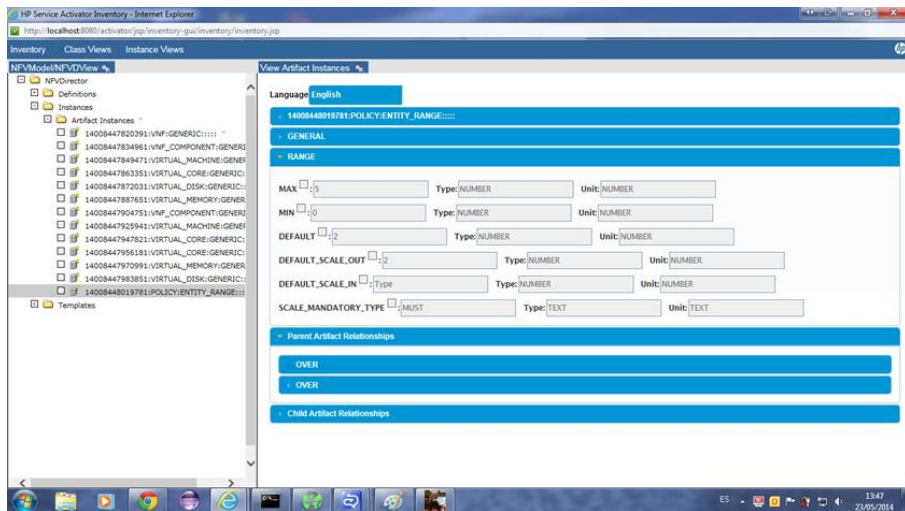
Figure 125: Scale In Example: Condition for scale in

Consider the parameters on the instance and not on the template.

In this case, 4 VIRTUAL_CORE instances depending on the POLICY are available.

MIN=2, DEFAULT_SCALE_IN=2   2<=4+2   √

The Scale In deletes 2 instances (DEFAULT_SCALE_IN parameter).

If the parameters do not satisfy the equation, the Scale In operation fails and does not delete any instance.

Apply the Scale In operation as a basic test.
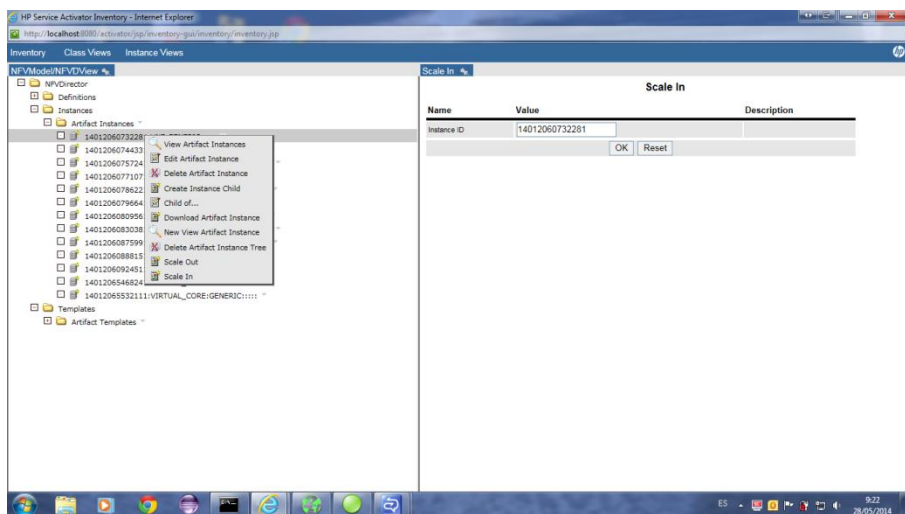
Figure 126: Scale In Example: Test

Delete 2 instance children depending on the POLICY.
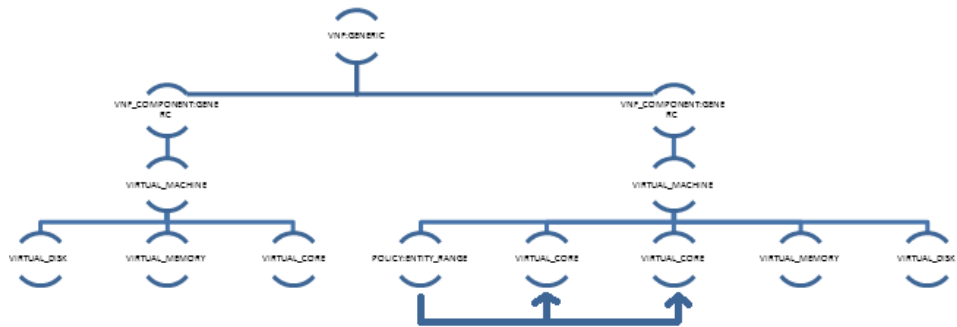
Figure 127: Scale In Example: Test Verification

10.1.4 Recap of End Messages

10.1.4.1 Errors

- 5001: ArtifactInstanceId is a mandatory input parameter.
- 5002: Artifact Instance with instanceId = %INPUT_INSTANCEARTIFACTID% does not exist in the system.
- 5003: Error delete instance from template.
- 5004: Recursive call failed.

10.1.4.2 Successful ends

- 0: Workflow ends ok.
- 0: OK. SCALE OUT Operation was not possible to do in all artifacts.

10.2 Scale-out

The scale out operation is built to generate, assign, and activate new resources on the instance tree. For example, if you have one virtual Machine with 1 child (Virtual Core), running the Scale Out operation increases the number of Virtual Cores according to the policy range.

Actually this operation can be called only from a VNF artifact.

The Scale Out operation manages the resource assignment and the activation. For more information about Policy Range, refer to the *8.3 Policies* section.
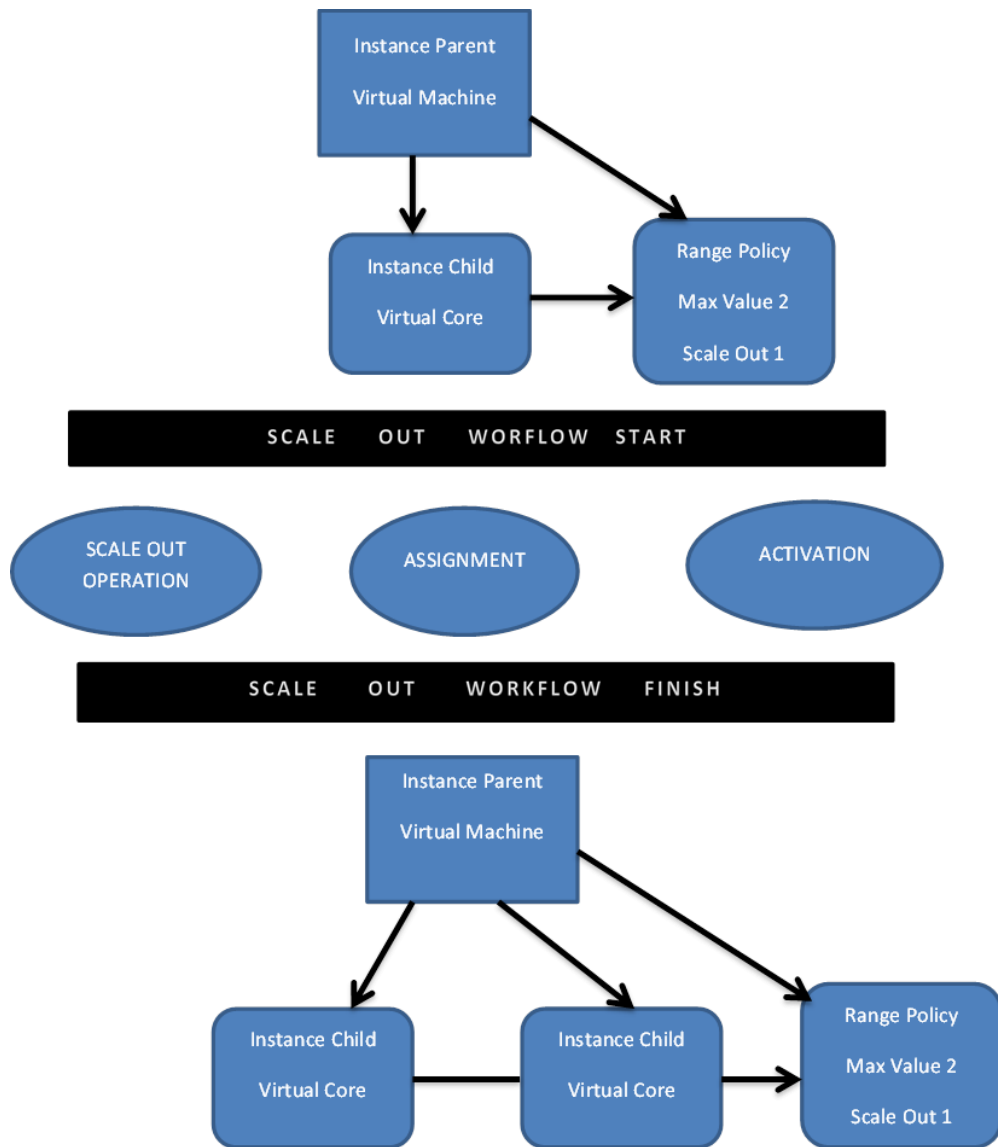
Figure 128: Scale Out

Particular case:

When the scale out workflow is called, it tries to scale as many elements as possible. If any element tries to scale above the maximum limit, it does not scale. However, the workflow continues scaling other components and displays a warning.

For example, consider a scenario where 2 virtual machine VNF starting as:

VM1 = 1 instance, VM2 = 1 instance, where

VM1 default=1, scale out=5, max= 2,

VM2 default=1, scale out=5, max= 10

If it tries to scale, the result is VM1 = 1, instance VM2 = 6 instance 1, because VM1 cannot scale above the maximum but VM2 can.

10.2.1 Launching the Workflow

1. Select **Instance** > **Artifact Instances**.
2. Right-click the instance template and select the **Scale Out** option.
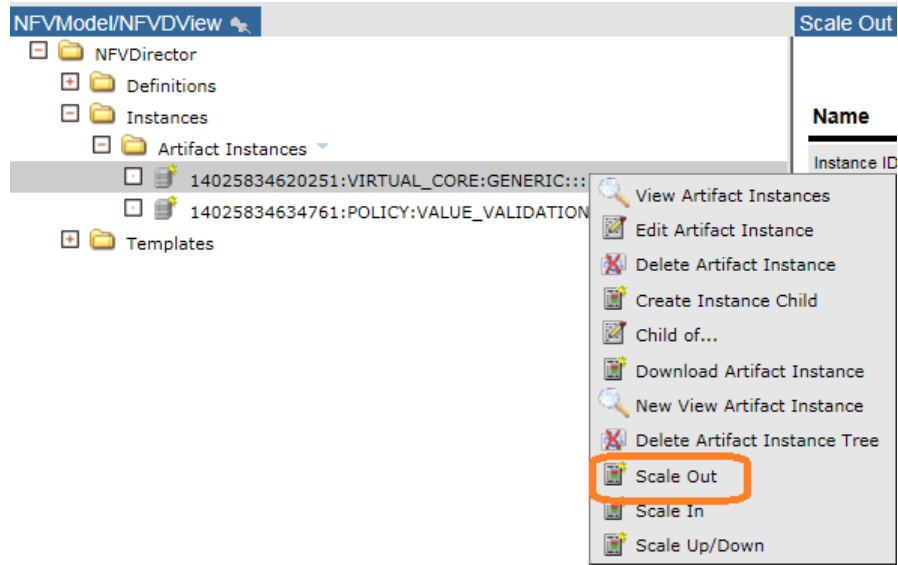
Figure 129: Scale Out: Launch Workflow

3. Check the Instance ID.
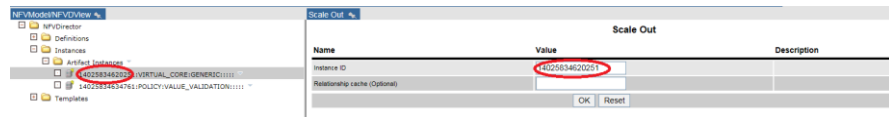
Figure 130: Scale Out: Verify Instance ID

4. Click the **OK** button to do the Scale.

Figure 131: Scale Out: Confirm operation

The workflow is launched and the scale is done.

10.2.2 Scale-out operation

Scale out operation takes care of increasing the amount of resources.

It starts checking the Instance ID and getting the artifact associated to this ID. After getting the artifact the operation continues getting the template associated with the instance.
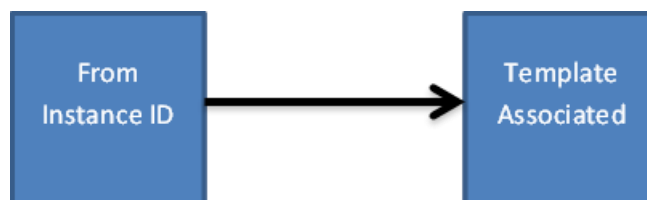
Figure 132: Scale out: query associated template

With the instance located the next step is to get all template children and check one by one whether it is a policy or not.
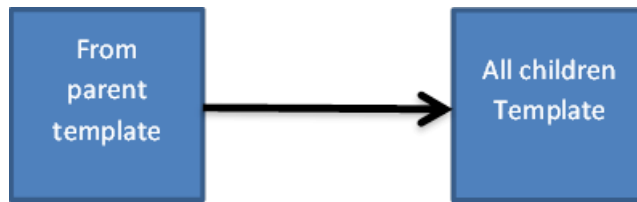
Figure 133: Scale out: query template children

If it is not a policy, the template Id is gotten and the scale out operation is called to apply it recursively.
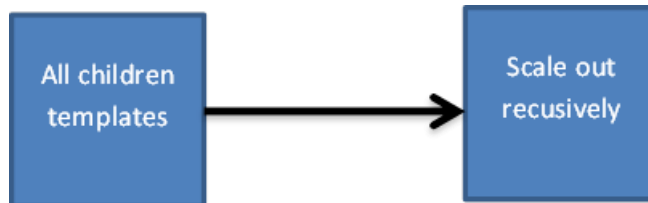
Figure 134: Scale out: apply scale out recursively

If it is a policy, the template id is gotten and all the children to get all the policies are called.

After policies evaluated and confirmed the process continues getting all the relationships between templates parents and children and calling create instance from template to create the new instances.
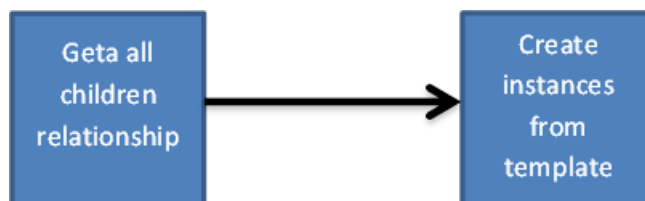
Figure 135: Scale out: create instances from template

Later, the operation gets all the relationships and checks the maximum range of the scale out.

After that the scale out operation is finished.
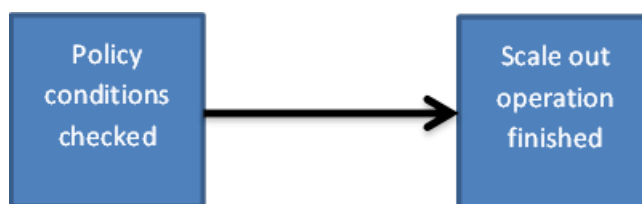
Figure 136: Scale out: operation completed

10.2.3 End-to-End Example

A policy node is available with DEFAULT=2, MAX=5, and DEFAULT_SCALE_OUT=2. Also, the node has OVER Parent Relationship to VIRTUAL_CORE.
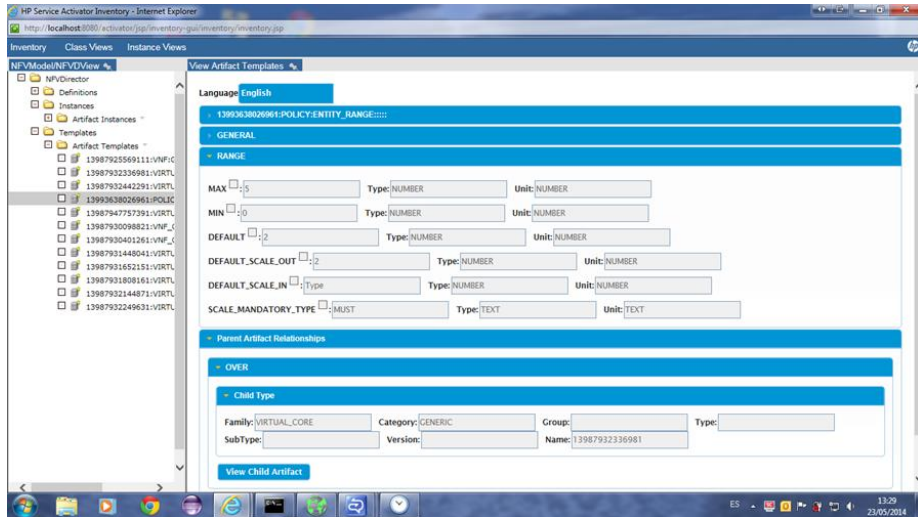
Figure 137: Scale out Example: Set policy

This scenario has an Artifact Template Tree with the following relationships.
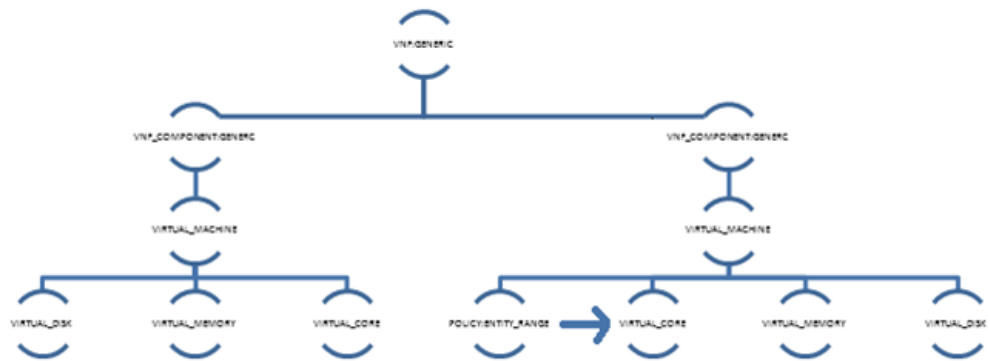
Figure 138: Scale out Example: Artifact template tree

Create Instance from the desired Template.
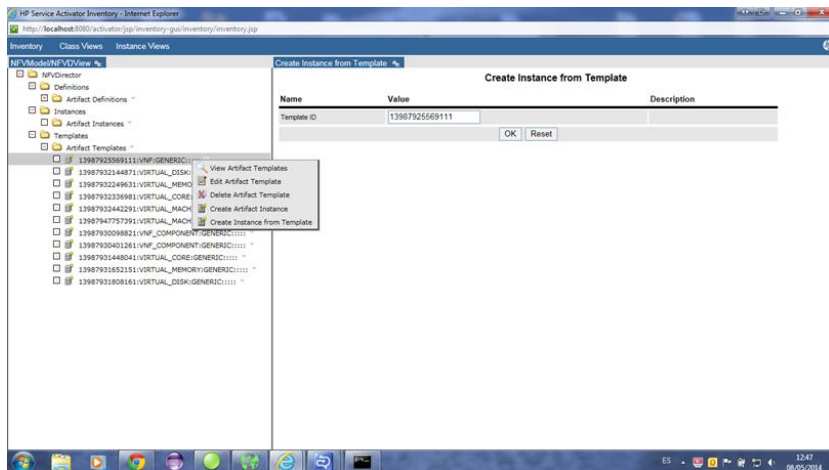
Figure 139: Scale out Example: Create instance from template

Create the same Artifact Instance Tree with the same relationships.

Even create 2 VIRTUAL_CORE Instances as policy index.
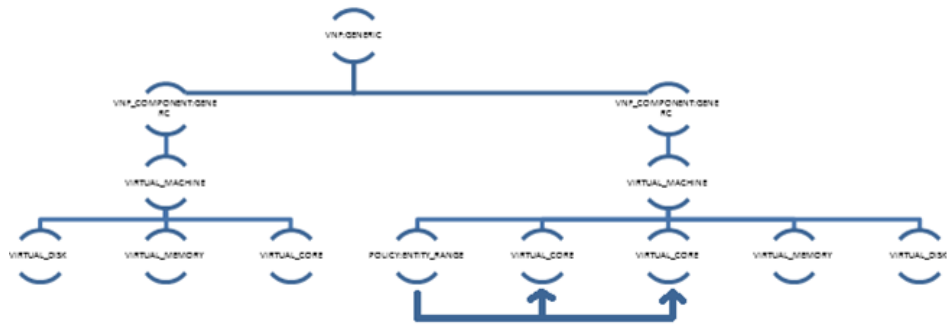
Figure 140: Scale out Example: Create virtual core

The DEFAULT_SCALE_OUT parameter is the number of instances you want to create (scale).

The MAX parameter is the maximum instances that you can have.

You can scale out only if MAX>=Instance amount in BBDD + DEFAULT_SCALE_OUT.
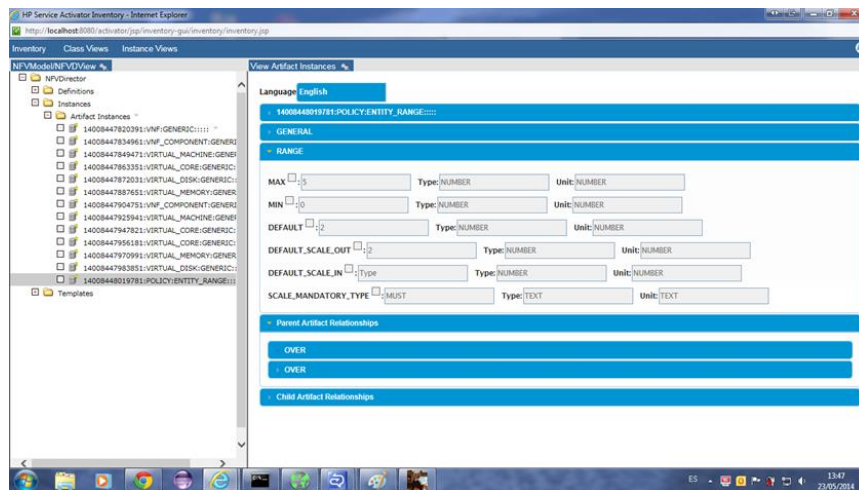
Figure 141: Scale out Example: Apply policy

Consider the parameters on the instance and not on the template. This scenario has 2 VIRTUAL_CORE instances depending on the policy.

MAX=5, DEFAULT_SCALE_OUT=2   5>=2+2   √

The Scale Out creates 2 instances (DEFAULT_SCALE_OUT parameter). If the parameters do not satisfy the equation, the Scale Out operation fails and does not create any instance.

Apply the Scale Out operation as the basic test.
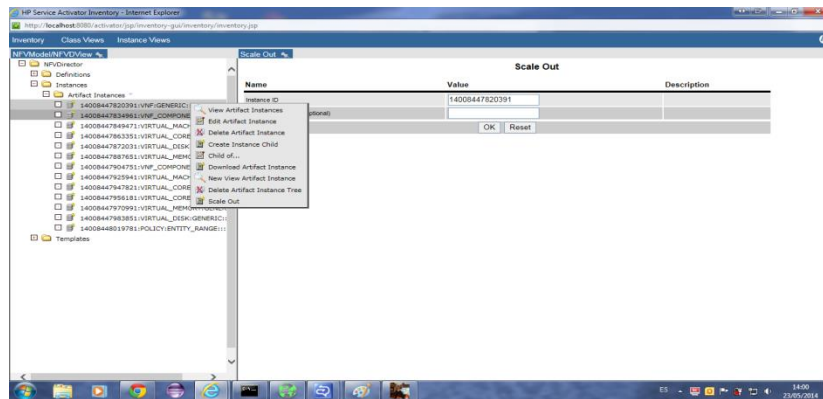
Figure 142: Scale out Example: Test

Create the 2 new instance child elements depending on the policy.
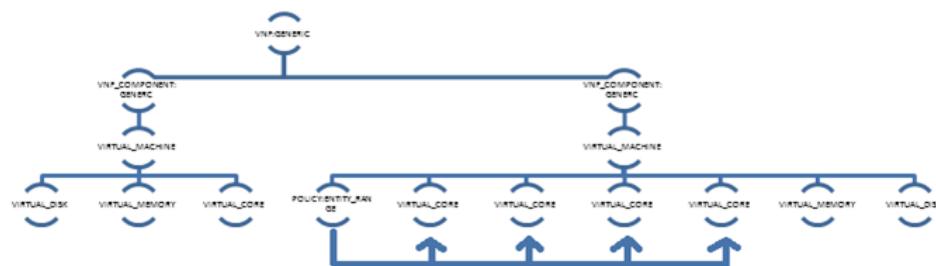
Figure 143: Scale out Example: Test verification

10.2.4 Recap of End Messages

10.2.4.1 Errors

- 4001: ArtifactInstanceId is a mandatory input parameter.

- 4002: Artifact Instance with instanceId = %INPUT_INSTANCEARTIFACTID% does not exist in the system.

- 4003: Scale out operation is only supported for instances created from template. InstanceId = %INPUT_INSTANCEARTIFACTID%.

- 4004: Recursive call failed.

- 4005: Malformed Template: The number of children of tempalteId = %VAR_TEMPLATE_PR.Id% is not 1.

- 4006: Malformed template: parent (%VAR_ARTIFACT_TEMPLATEID%) of the policy (%VAR_TEMPLATE_PR%), must be parent of the OVER Child (%VAR_TEMPLATE_PR_OVER_CHILD%) too.

- 4007: ERROR Create Instance From Template WF.

- 4008: ERROR Create New Child Relationship.

10.2.4.2 Successful Ends

- 0: Workflow ends ok.

- 0: OK. SCALE OUT Operation was not possible to do in all artifacts.

10.3 Scale Up

This operation enlarges attributes for an instance based on the entity-scale policy (see *8.3 Policies*). It is necessary to have defined relationships that are required.

The operation has an impact both in DB and on the Openstack side, changing the VM flavor, according to the attributes set in DB.

For example, if you have one virtual Machine with 3 children (Virtual Core, Virtual Disk and Virtual Memory), and the Virtual Memory is parent of the scale policy, running the Scale Up operation increases the content of Virtual Core Amount attribute according to the policy range.

On the Openstack side, when the increase is successful, it tries to find a flavor that matches the DB amounts. If it finds it, it automatically changes the flavor of the VM.

10.3.1 Launching the Workflow

1. Select **Instance** > **Artifact Instances**.
2. Right-click on the VNF or VNF_COMPONENT instance and select the **Scale Up** option.
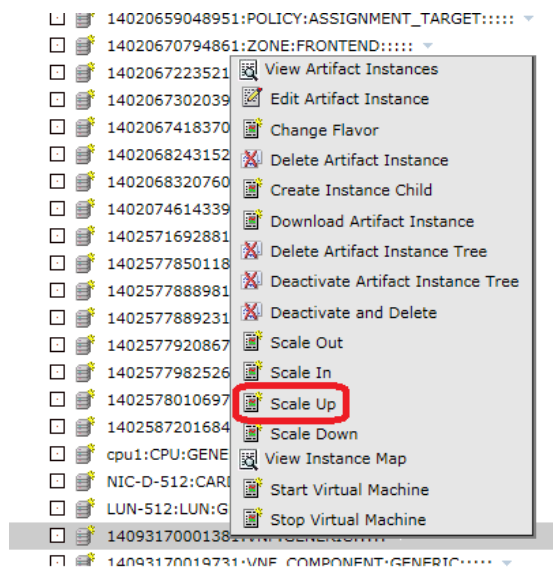
Figure 144: Scale up: launch

3. It is possible to set the value of "Scale All Tree" and "Force Stop of Virtual Machines" parameters. If these fields are empty, the predefined values are "true" and "false" respectively.

   If "Scale All Tree" is "false", the scale has effect only over the artifact where the operation has been done.

   If "Force Stop of Virtual Machines" is "true", the VMs will be stopped before resizing the flavor.

Scale Up

Name	Value	Description
Service Name	VNF	
Service Type	NFVD	
Service Operation	SCALE_UP	
Instance ID	14093170001381	
Scale All Tree?		
Force Stop of Virtual Machines?		

OK   Reset

Figure 145: Scale up operations

4.   Click **OK** and wait till the operation ends.

10.3.2 Scale Up operation

The purpose of Scale Up operation increases the quantity of the attribute INFO.Amount allocated in the VIRTUAL_CORE, VIRTUAL_MEMORY and VIRTUAL_DISK.

The increment is defined by the scale policy in its attribute SCALE:INCREASEAMOUNT. This policy has to be child of the artifact ready to scale.

Once the scale is done in DB, immediately it looks for a flavor in the VIM (Icehouse, CS8, …) according to the amount of core, memory and disk in DB and if it finds it, it changes the VM flavor.

As we have seen, this operation has an implicit restart over VM. This process will check the state of monitors and VMs in case it needs to be re-launched.
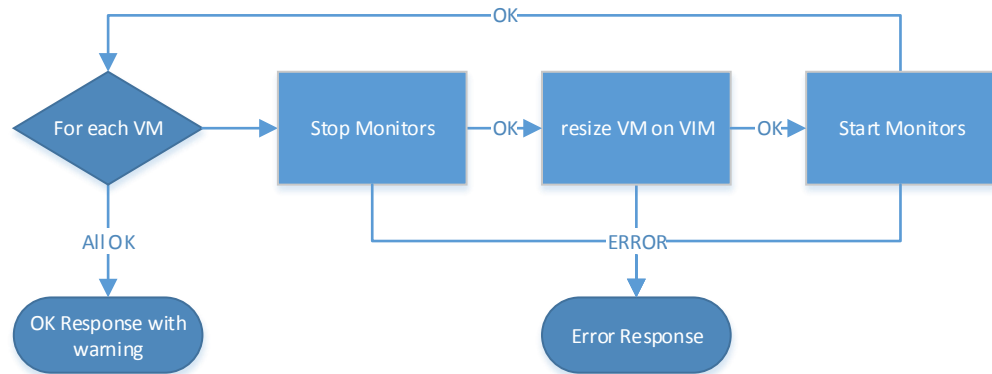
Figure 146: Scale up flow

Next, the details of the completion will be listed.

On the DB side, the operation starts looking for the VNF and sets its status as "LOCKED".

Then it looks for scale policies. When it finds one, it gets the MAX and INCREASEAMOUNT attribute values in order to check if the scale is possible, validating through this equation:

CURRENT_VALUE (INFO.AMOUNT of artifact to scale)+INCREASEAMOUNT<=MAX

Next, it checks if there are free physical resources to allocate and validates the new value with WF_NFVD_INSTANCE_VALIDATION operation.

Finally, it sets the VM status to "TO_BE_SCALE".

When the scale in DB ends OK, it starts the activation side.

On this side, first it queries the VMs to change the flavor.

Then, it looks for the VIRTUAL_DATACENTER or TENANT and gets GENERAL.Name (tenant name) and stops the monitor if it is possible.

For each VM, it updates LAST_OPERATION attribute to "resize", and gets the VIM artifact to extract the url, user and pass of the VIM.

Immediately, it launches the wf that resizes the flavor and sets the result of this operation in the VM attributes called LAST_OPERATION.Result_code and LAST_OPERATION.Result_description.

Finally, it starts the monitors and sets the VNF status to "ENABLE".

For more detailed information about VM and VNF status and its lifecycle, refer to section *8.4.1.3 Scale Up/Down*.

10.3.3 End-to-End Example

A scale-policy node is available with MAX=2048, INCREASEAMOUNT=512, and DESTINY=INFO.Amount. Also, the node has APPLY Child Relationship to VIRTUAL_MEMORY.

Figure 147: Scale up example

This scenario has a VNF Tree with the following relationships.

Figure 148: VNF tree for scale up

The VIRTUAL_MEMORY, belonged to the VM, in DB has 512mb of memory (INFO.Amount).

Figure 149: Memory amount

The VIM, Icehouse in this case, has the VM activated with a flavor of 512mb memory.

	Instance Name	Image Name	IP Address	Size
☐	test	cirros_new	16.17.101.19	1vcpu_512RAM_1_Disk | 512MB RAM | 1 VCPU | 1.0GB Disk

Figure 150: Memory amount instance

Scale Up operation from the inventory tree.
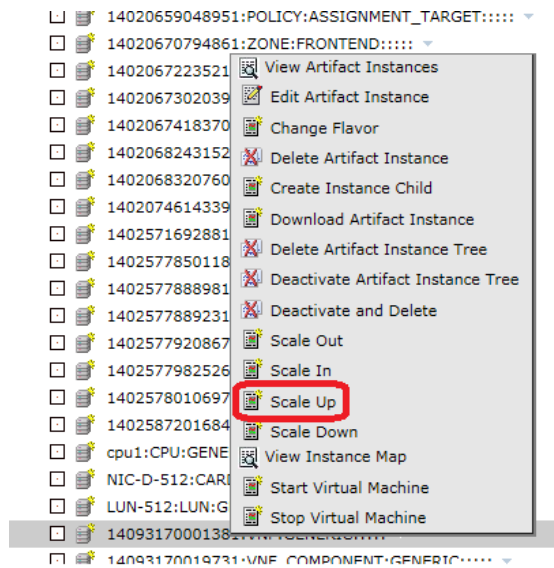
Figure 151: Scale up: launch

All the parameters by default.

Figure 152: Scale up: operations

The results are the increase of 512mb in DB.

Figure 153: Scale up: results

Even the resized flavor in the VIM (Icehouse).

	Instance Name	Image Name	IP Address	Size
☐	test	cirros_new	16.17.101.19	1vcpu_1024RAM_1_Disk | 1GB RAM | 1 VCPU | 1.0GB Disk

10.3.4 Recap of End Messages

10.3.4.1 Errors

WF_NFVD_SCALE_UPDOWN_ACTIVATION

- 16001: Mandatory input parameter ArtifactInstanceId is not present

- 16002: Exist more than 1 tenants

- 16003: Exist more than 1 Virtual Datacenters16004: Recursive call failed.

- 16005: Dont exist VirtualDatacenter or Tenant

- 16006: Dont exist VNF or exists more than 116007: ERROR Create Instance From Template WF.

- 16009: Stop monitor was not ok.

- 16010: Start monitor was not ok

- 16011: Change Flavor was not ok

WF_NFVD_SCALE_UPDOWN_INVENTORY

- 17001: Mandatory input parameter ArtifactInstanceId is not present

- 17002: Instance Artifact Id dont exist in system

- 17003: Mandatory input parameter INPUT_OPERATION is not present or is not valid

- 17004: Dont exist VNF or exists more than 1

- 17005:Min limit exceeded

- 17006: Max limit exceeded

- 17007: It cant be allocated

- 17008:Exist more than VM Parent

WF_NFVD_SCALE_UPDOWN_INVENTORY

- 15001: Mandatory input parameter ArtifactInstanceId is nor present

- 15002:No Flavor Found

- 15003: ERROR Stopping the server

- 15004: ERROR Starting VM

- 15005: ERROR INCONSISTENT

- 15006: ERROR Resizing the server

- 15007: VM has not a valid state

- 15008: ERROR Confirming the resize of the server

- 15009: ERROR Starting VM

- 15010: Query flavor was not ok

- 15011:Query flavors was not ok

- 15012: Stop server was not ok

- 15013:Start server was not ok

- 15014:Revert resize was not ok

- 15015: Starting server was not ok in the rollback

10.3.4.2 Successful Ends

- 0: End OK.

- 0: End OK. No scale policies to execute

10.4 Scale Down

This operation decreases attributes for an instance based on the entity-scale policy (see *8.3 Policies*). It is necessary to have defined relationships that are required.

It is the opposite operation of Scale Up.

The operation has an impact both in DB and on the Openstack side, changing the VM flavor, according to the attributes set in DB.

On the Openstack side, when the decrease is successful, it tries to find a flavor that matches the DB amounts. If it finds it, it automatically changes the flavor of the VM.

10.4.1 Launching the Workflow

Follow the same steps to launch the Scale Up operation, but now click on **Scale Down**. (See the *10.3.1 Launching the Workflow* section)
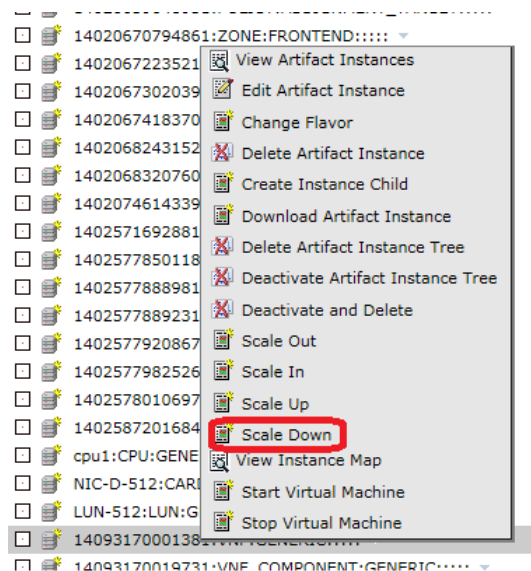
Figure 155: Scale down: launch

10.4.2 Scale Down operation

The description of the Scale Up is exactly the same as the Scale Down, but the result is a decrease of the amounts. To achieve the decrease, it is necessary to accomplish this equation:

CURRENT_VALUE (INFO.AMOUNT of artifact to scale)+DECREASEAMOUNT>=MIN

See the *10.3.2 Scale Up operation* section.

10.4.3 End-to-End Example

A scale policy node is available with MIN=512, DECREASEAMOUNT=512, and DESTINY=INFO.Amount. Also, the node has APPLY Child Relationship to VIRTUAL_MEMORY.

Figure 156: Scale down example

This scenario has a VNF Tree with the following relationships.
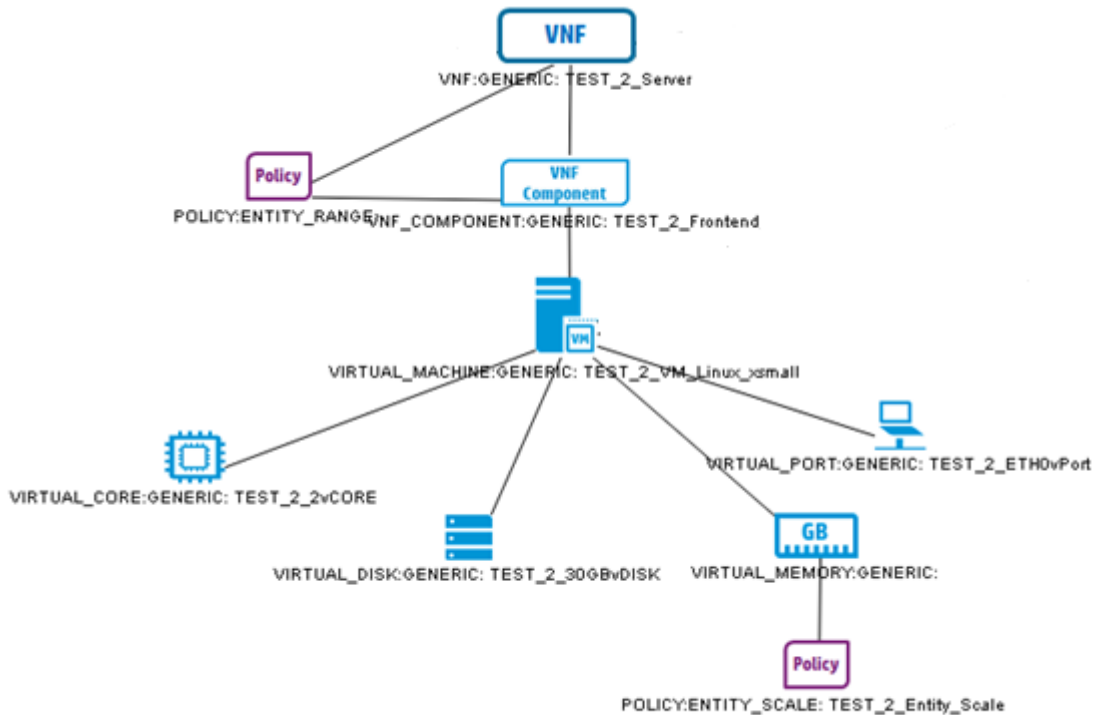
Figure 157: VNF tree for scale down

The VIRTUAL_MEMORY, belonged to the VM, in DB has 1024mb of memory (INFO.Amount).

Figure 158: Memory amount

The VIM, Icehouse in this case, has the VM activated with a flavor of 1024mb memory.

	Instance Name	Image Name	IP Address	Size
☐	test	cirros_new	16.17.101.19	1vcpu_1024RAM_1_Disk | 1GB RAM | 1 VCPU | 1.0GB Disk

Figure 159: Memory amount instance

Scale Down operation from the inventory tree.
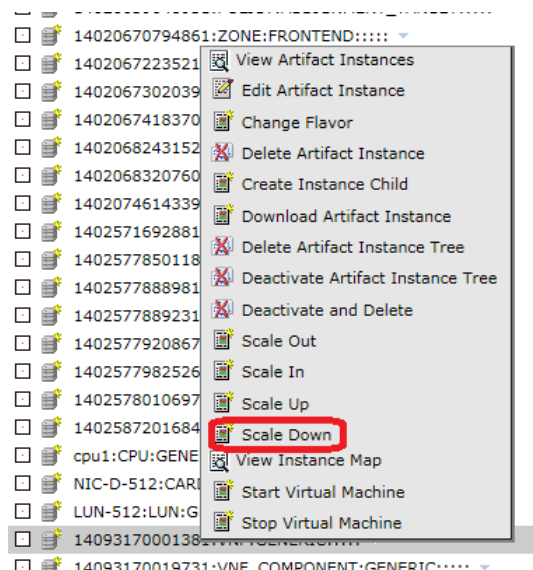
Figure 160: Scale down: launch

All the parameters by default.

Scale Down

Name	Value	Description
Service Name	VNF	
Service Type	NFVD	
Service Operation	SCALE_DOWN	
Instance ID	14093170001381	
Scale All Tree?		
Force Stop of Virtual Machines?		
	OK Reset	

Figure 161: Scale down: operations

The results are the decrease of 512mb in DB.

Language **English**

14093170303631:VIRTUAL_MEMORY:GENERIC:::::

▼ **INFO**

ID ☐ : TEST_2_40GBvRAM_2   Type: TEXT   Unit: TEXT

Name ☐ : Name   Type: TEXT   Unit: TEXT

Type ☐ : Type   Type: TEXT   Unit: TEXT

Amount ☐ : 512   Type: Number   Unit: GB

▶ **Child Artifact Relationships**

Figure 162: Scale down: operations

Even the resized flavor in the VIM (Icehouse).

	Instance Name	Image Name	IP Address	Size
☐				
☐	test	cirros_new	16.17.101.19	1vcpu_512RAM_1_Disk | 512MB RAM | 1 VCPU | 1.0GB Disk

Figure 163: Scale down: resize flavor

10.4.4   Recap of End Messages

10.4.4.1 Errors

The same that the Scale Up operation. (See 10.3.4.1)

10.4.4.2 Successful Ends

The same that the Scale Up operation. (See 10.3.4.2)

Chapter 11

Activation

The third step on global creation process is activation, and the process will be responsible to check the relationship between the Virtual Data Center and the machine, after that the resources will be detected in a flavor. Later the activation will be effective.
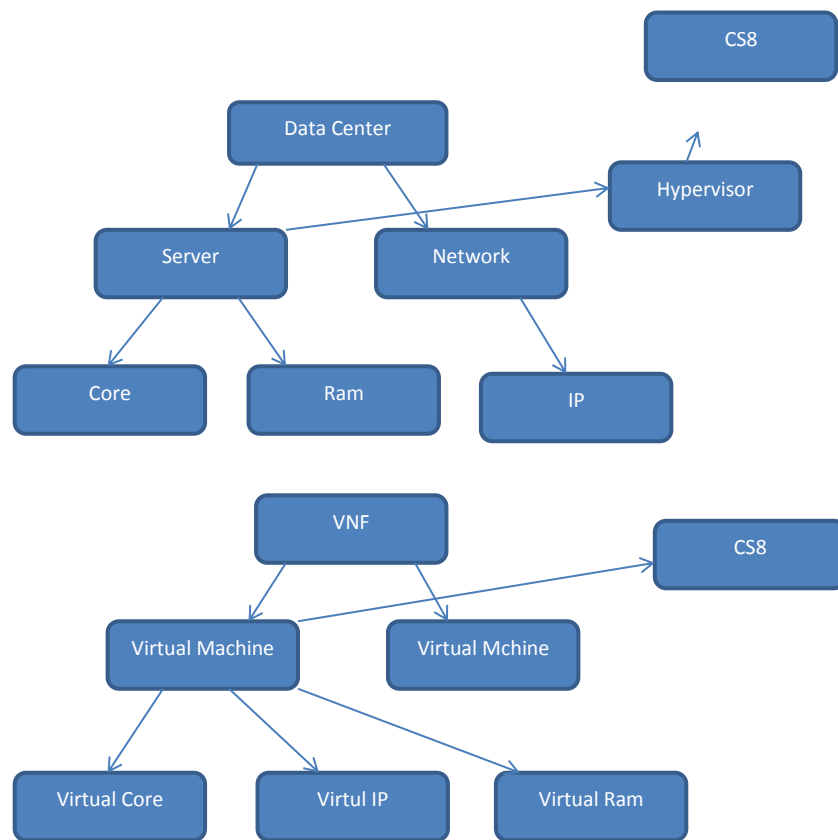
Figure 164: Activation flow

The basic steps in the process are:

- Check the image and take the image ID.
- Check a flavor with ram and disk.
- Check network Id.
- Create Server with image ID, RAM, disk, and network ID.

11.1 Checking and getting the Virtual Data Centers

First of all the workflow receives VIM in the input and it checks the information is correct and also it checks the Virtual Data Center associated.
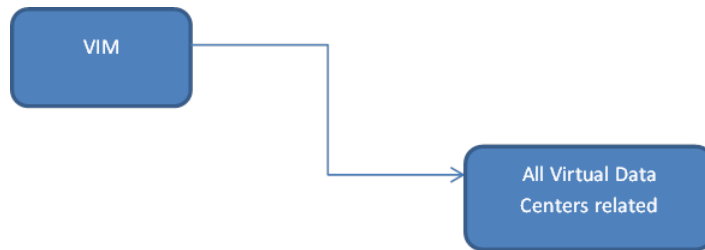
Figure 165: Activation: get vDC

The action manages the extracting of all data that is required later.

11.2 Checking a flavor with RAM and disk

After you get the Virtual Data Center, you should compose the flavor. It is mandatory to get the Virtual RAM and then the virtual disk.
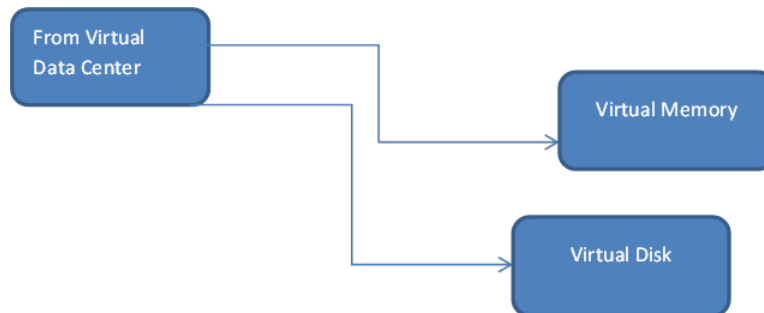
Figure 166: Activation: get Virtual Memory and Virtual Disk

When the virtual memory and the virtual disk are obtained, check the correct value of both in CS8 Flavor. The flavor is used later. The flavor ID is extracted to use in the operation.

11.3 Getting Image ID and network ID

After getting the flavor, get the machine image.

1. For getting the image, the OpenStack workflow is called from the activation.
2. Other OpenStack workflow is called to assign the resources to an output object.
3. The network connected to the object is located.

   An OpenStack workflow is called to get the network.
4. When the resource is added, you should assign the output-object to the network.
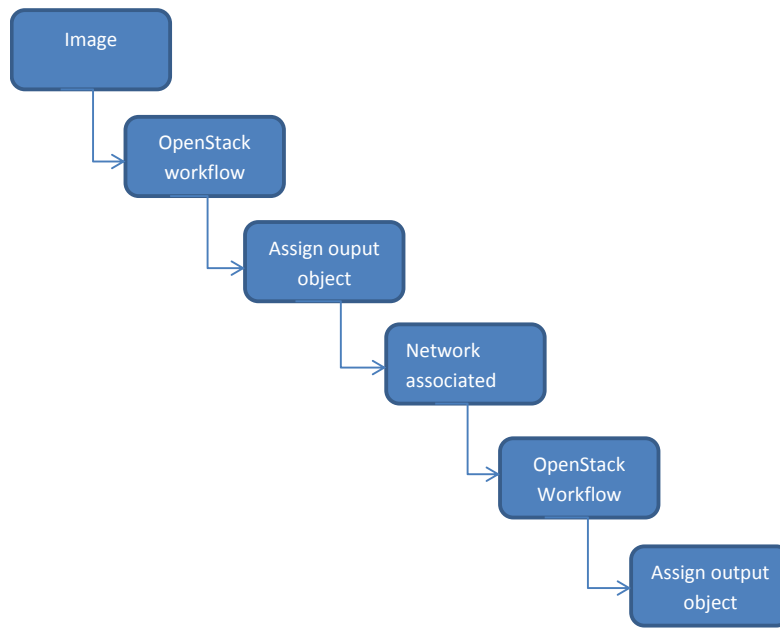
Figure 167: Activation: get Image ID and network ID

11.4 Creating Server

In this moment another different open stack Workflow is called. This workflow is called create server cs8 and it will work in the same way as the other cs8 workflows do. The operation is called and when the data is gotten and output object is assigned.

11.5 Updating Status

When the server is created, the check operation is coming—the server previously created is called and associated to an output object, then the activation checks the server has been created properly and the machine is active, and it changes the status to activated, then it takes the ID returned by CS8 and sets it into VIM ID—the artifact is updated.
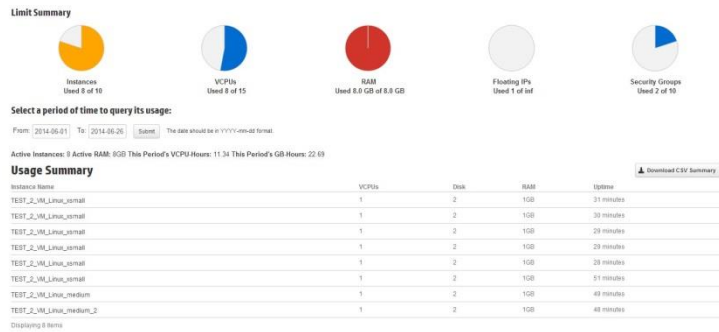
Figure 168: Activation: update Status

11.6 Activating workflow parent

The cs8 activation is called from a workflow parent. It works getting all the VNF components and getting the VIM components. Depending on the VIM status the full operation of activation is called or not.

11.6.1 Testing

The activation is the last part of other operations:

- Create instance from template

- Scale in

- Scale Out

- Scale Up/down

Test it properly to launch the other operations. However, you can check it using the Soap UI and testing this code.

```
ACTIVATE:

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ngws="http://www.hp.com/sosa/protocoladapter/ngws">
  <soapenv:Header/>
  <soapenv:Body>
    <ngws:startServiceOrderAsync>
      <ngws:type>NFVD</ngws:type>
      <ngws:name>VNF</ngws:name>
      <ngws:action>ACTIVATE</ngws:action>
      <!--Optional:-->
      <ngws:inputParams>
        <!--Zero or more repetitions:-->

        <ngws:param>
          <ngws:name>INPUT_ARTIFACTTREEID</ngws:name>
          <ngws:value>14036935344741</ngws:value>
        </ngws:param>

      </ngws:inputParams>
      <ngws:user>?</ngws:user>
      <!--Optional:-->
      <ngws:userId>?</ngws:userId>
    </ngws:startServiceOrderAsync>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 169: Activation: Test

11.7 Deactivating workflow

The deactivate_cs8 workflow is called to deactivate virtual machines. The process of deactivating involves using an artifact ID to get VIM, virtual datacenter, and through it, the server name.

When the server name is obtained, the OpenStack delete operation is called and after a test, the operation is completed.
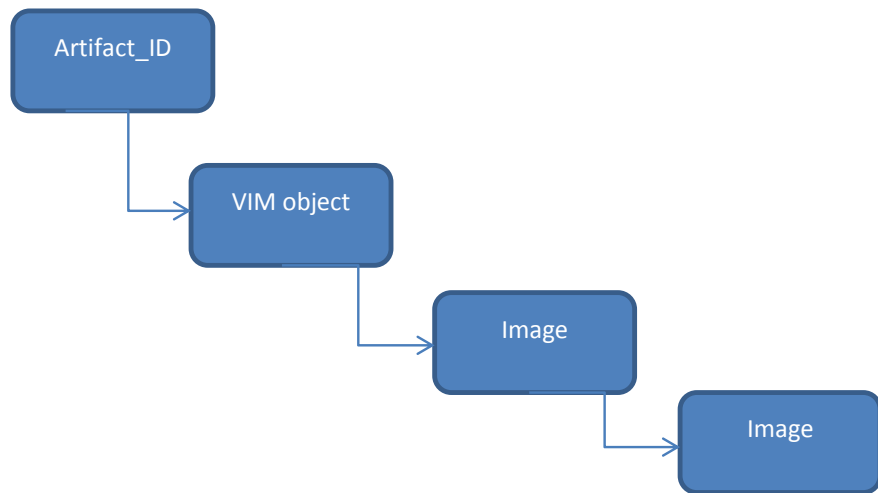
Figure 170: Deactivate flow

11.8 Error Recap

- 6XXX—Error related to active flow parent.
- 1XXXX—Error related to activate cs8.
- 14XXX—Error related to deactivate flow.

Chapter 12

Administering NFV Director Components

This section describes the procedure to manage or administer various components of NFV Director.

12.1 Stopping/Starting NFV Director

The `nfv-director.sh` script starts and stops the NFV-Director. The script is available in the `<Base RPM Install Path>/opt/HP/nfvd/bin` directory.

Script usage:

```
nfv-director.sh:
```

Usage:

```
nfv-director.sh [OPTIONS...]
 -a start | stop | restart | status
 [-c] [ activator | sosa | ecpool | lockmgr | ppasdb |
openmediation | SiteScope | uca-ebc | uca-atm | nfvd-agw ]
```

To get help on `nfvd-director.sh`:

```
#nfv-director.sh -h
```

```
#./nfv-director.sh -h

Usage: nfv-director.sh [OPTIONS...]
 -a start | stop | restart | status
 [-c] [ activator | sosa | ecpool | lockmgr | ppasdb | openmediation | sitescope | uca-ebc | uca-atm | nfvd-agw ]
#
```

Note

The default stop script works for components installed in the default location. Otherwise, the user should run the start/stop commands provided by the individual components in their respective installation directories.

Alternate commands are provided wherever applicable if the product is installed in non-default locations.

The following is the list of components start/stop in order when the script is run.

1. HP Service Activator(HPSA)
2. HP Service Order Smart Adapter (SOSA)
3. HP Equipment connection pool (ECP)
4. HP Lock Manager
5. Postgres Plus Advanced Server 9.2
6. HP Open Mediation
7. HP SiteScope

8. HP UCA-EBC

9. HP UCA Automation Console

10. HP NFVD Assurance Gateway

12.1.1 Starting all components

To start all components, run the following command:

```
#nfv-director.sh -a start
```

The following is a sample of the output:

Figure 171: nfv-director.sh: start all components output

12.1.2 Stopping all NFV-Director components

To stop all NFV-Director components, run the following command:

```
#nfv-director.sh -a stop
```

Following is a sample of the output:

Figure 172: nfv-director.sh: stop all components

Following is a sample of the output for the stop command:

Figure 173: nfv-director.sh: stop all components output

12.1.3 Getting status of NFV-D components

To check status of NFV-D components, run the following command:

```
#nfv-director.sh -a status
```

Following is a sample of the output:

```
================================================================================
                    NFV Director Solution - Process Status Modifier
================================================================================
HP Service Activator(HPSA)  status...
HP Service Activator application server is running

HP Service Activator(HPSA) status action completed ...

HP Service Order Smart Adapter (SOSA) does not exists, kindly install or execute nfv-director.sh script from localhost...

HP Equipment connection pool (ECP) does not exists, kindly install or execute nfv-director.sh script from localhost...

HP Lock Manager does not exists, kindly install or execute nfv-director.sh script from localhost...

Postgres Plus Advanced Server 9.2  status...
pg_ctl: server is running (PID: 18220)
/opt/PostgresPlus/9.2AS/bin/edb-postgres "-D" "/opt/PostgresPlus/9.2AS/data"

Postgres Plus Advanced Server 9.2 status action completed ...

HP Open Mediation  status...
List of the containers:
0        STARTED          Hub

HP Open Mediation status action completed ...

HP SiteScope  status...
Sitescope is running

HP SiteScope status action completed ...

HP UCA-EBC  status...
Fatal error : /var/opt/UCA-EBC/instances/default/instances/default does not exist. Change the -i option value or create instance default

HP UCA-EBC status action completed ...

HP UCA Automation Console  status...
*** INFO: UCA Automation Console is not running.

HP UCA Automation Console status action completed ...

HP NFVD Assurance Gateway  status...
HP Assurance Gateway application server is running

HP NFVD Assurance Gateway status action completed ...
```

Figure 174: nfv-director.sh: get status

Following is a sample of the status command output:

```
HP Equipment connection pool (ECP) stop action completed ...

HP Lock Manager  stop...
Connecting to rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:32:48,748 [main] INFO RmiLockManagerClient - Invoking service rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:32:49,171 [main] INFO RmiLockManagerClient - Return: LockManager shutdown successfull
HP Lock Manager stop action completed ...

Postgres Plus Advanced Server 9.2 does not exists, kindly install or execute nfv-director.sh script from localhost...

HP Open Mediation does not exists, kindly install or execute nfv-director.sh script from localhost...

HP SiteScope does not exists, kindly install or execute nfv-director.sh script from localhost...

HP UCA-EBC does not exists, kindly install or execute nfv-director.sh script from localhost...

HP UCA Automation Console does not exists, kindly install or execute nfv-director.sh script from localhost...

HP NFVD Assurance Gateway does not exists, kindly install or execute nfv-director.sh script from localhost...
```

Figure 175: nfv-director.sh: get status output1

Following is another sample of the output:

```
HP Service Activator(HPSA)  status...
HP Service Activator application server is running
HP Service Activator(HPSA) status action completed ...

HP Service Order Smart Adapter (SOSA)  status...
Testing sosa
Sosa is running
HP Service Order Smart Adapter (SOSA) status action completed ...

HP Equipment connection pool (ECP)  status...
Connecting to rmi://127.0.0.1:1200/
20140622-142634.558: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Finding rmi://127.0.0.1:1200/RmiEcpService
20140622-142634.678: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Service rmi://127.0.0.1:1200/RmiEcpService found: R
miEcpService_Stub[UnicastRef [liveRef: [endpoint:[127.0.0.1:1102](remote),objID:[53c1a67d:146bd1453b5:-7ffe, -459705695538777
17]]]]
20140622-142634.678: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Executing remote task with: {}
20140622-142634.681: com.hp.spain.connection.pool.server.RmiEcpClient: 1: {result=The RMI Registry is up.}
20140622-142634.681: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Invoking service RmiEcpService
20140622-142634.683: com.hp.spain.connection.pool.server.RmiEcpClient: 1: [EMAPool, HSSPool, MNPPool]
20140622-142634.706: com.hp.spain.connection.pool.server.RmiEcpClient: 1: POOL_MANAGER_RUNNING = true
20140622-142634.707: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Return: [{NOMBRE=EMAPool, ESTADO_POOL=AVAILABLE, PO
OL_BUSY=false, REQUEST_MANAGER_RUNNING=true, RESOURCE_MANAGER_RUNNING=true, SUBPOOLS=[{NOMBRE_SUBPOOL=0, ESTADO_SUBPOOL=AVAIL
ABLE, SUBPOOLINFO=telnet://hp@127.0.0.1:3337<BR>driver TemplateDriver<BR>busy 0; free 0; max 5; min 1, SESIONES=[{ES_SPY=fals
e, NOMBRE_SESION=1, ESTADO_SESION=DISCONNECTED, INACTIVE, CADENA_ACCION=EMAPool;0;1;}]}], PETICIONES=0, POOL_RUNNING=true}, {
NOMBRE=MNPPool, ESTADO_POOL=AVAILABLE, POOL_BUSY=false, REQUEST_MANAGER_RUNNING=true, RESOURCE_MANAGER_RUNNING=true, SUBPOOLS
=[{NOMBRE_SUBPOOL=0, ESTADO_SUBPOOL=AVAILABLE, SUBPOOLINFO=tcp://hp@127.0.0.1:8000<BR>driver TemplateDriver<BR>busy 0; free 0
; max 5; min 1, SESIONES=[{ES_SPY=false, NOMBRE_SESION=3, ESTADO_SESION=DISCONNECTED, INACTIVE, CADENA_ACCION=MNPPool;0;3;}]}
], PETICIONES=0, POOL_RUNNING=true}, {NOMBRE=HSSPool, ESTADO_POOL=AVAILABLE, POOL_BUSY=false, REQUEST_MANAGER_RUNNING=true, R
ESOURCE_MANAGER_RUNNING=true, SUBPOOLS=[{NOMBRE_SUBPOOL=0, ESTADO_SUBPOOL=AVAILABLE, SUBPOOLINFO=telnet://hp@127.0.0.1:23<BR>
driver TemplateDriver<BR>busy 0; free 0; max 5; min 1, SESIONES=[{ES_SPY=false, NOMBRE_SESION=2, ESTADO_SESION=DISCONNECTED,
INACTIVE, CADENA_ACCION=HSSPool;0;2;}]}], PETICIONES=0, POOL_RUNNING=true}]
HP Equipment connection pool (ECP) status action completed ...

HP Lock Manager  status...
Connecting to rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:26:34,832 [main] INFO RmiLockManagerClient - Invoking service rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:26:34,914 [main] INFO RmiLockManagerClient - Return: Service running. Lock manager notificator running.
HP Lock Manager status action completed ...
```

Figure 176: nfv-director.sh: get status output2

12.1.4 Restarting NFV-Director components

To restart NFV-Director components, run the following command:

```
#nfv-director.sh -a restart
```

12.2 Fulfillment components

12.2.1 Starting the Activator

To start the activator, run the following command:

```
#nfv-director.sh -a start -c activator
```

Alternate command:

```
/etc/init.d/activator start
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a start -c activator
Start HP Service Activator daemon
Starting HP Service Activator application server
HP Service Activator(HPSA) start action completed...
nfvdff /tmp $
```

12.2.2 Getting status of the Activator

To check activator status, run the following command:

```
#nfv-director.sh -a status -c activator
```

Alternate command:

```
/etc/init.d/activator check
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a status -c activator
HP Service Activator application server is running
HP Service Activator(HPSA) status action completed...
nfvdff /tmp $
```

12.2.3 Stopping the activator

To stop the activator, run the following:

```
#nfv-director.sh -a stop -c activator
```

Alternate command:

```
/etc/init.d/activator stop
```

```
nfvdff /tmp $ ./nfv-director.sh -a stop -c activator
Stop HP Service Activator daemon
Stopping HP Service Activator (PID:26813)
HP Service Activator(HPSA) stop action completed...
nfvdff /tmp $
nfvdff /tmp $ ./nfv-director.sh -a status -c activator
HP Service Activator application server is not running
HP Service Activator(HPSA) status action completed...
nfvdff /tmp $
```

12.2.4 Starting HP SOSA

To start the HP SOSA, run the following command:

```
#nfv-director.sh -a start -c sosa
```

Alternate Command:

```
/opt/OV/ServiceActivator/EP/SOSA/bin/sosa.sh start
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a start -c sosa
Starting sosa
/usr/java/jdk1.6.0_37/bin/java -Dsosa -Djava.security.policy=conf/sosa.policy -Dlog4j.configuration=properties/sosa-log4j.pr
operties -Xms256m -Xmx1024m   -classpath .::properties:conf:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/sosa.jar:/opt/H
P/jboss/standalone/deployments/hpsa.ear/lib/mwfm.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/activator_utils.jar:/o
pt/HP/jboss/standalone/deployments/hpsa.ear/lib/resmgr.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/inventoryruntime
.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/ep-utils.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/commons
-codec-1.5.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/sosa-hibernate.jar:lib/activation-1.1.1.jar:lib/antlr-2.7.6.
jar:lib/asm-1.5.3.jar:lib/axis-1.4.jar:lib/axis-ant-1.4.jar:lib/c3p0-0.9.1.2.jar:lib/cglib-2.1.3.jar:lib/commons-collections-
3.2.1.jar:lib/commons-dbcp-1.3.jar:lib/commons-discovery-0.2.jar:lib/commons-el-1.0.jar:lib/commons-lang-2.2.jar:lib/commons-
logging-1.0.4.jar:lib/commons-pool-1.5.6.jar:lib/displaytag-1.1.jar:lib/dom4j-1.6.jar:lib/dtdparser-121.jar:lib/edb_jdbc.jar:
lib/gmbal-api-only-3.1.0.jar:lib/gson-2.2.2.jar:lib/ha-api-3.1.8.jar:lib/hibernate-3.2.5-modif.jar:lib/inventory_NFVModel.jar
:lib/jasper-compiler-1.0.jar:lib/jasper-runtime-1.0.jar:lib/javassist-3.8.0.GA.jar:lib/javax.servlet-5.1.11RC0.jar:lib/jaxb-a
pi-2.2.4.jar:lib/jaxb-impl-2.2.4-1.jar:lib/jaxrpc-1.1.jar:lib/jaxrs-api-3.0.3.Final.jar:lib/jaxws-api-2.2.jar:lib/jaxws-rt-2.
2.5.jar:lib/jersey-client-1.4.jar:lib/jersey-core-1.4.jar:lib/jettison-1.3.4.jar:lib/jox-improved-116.jar:lib/jsr173_api.jar:
lib/jsr181-api.jar:lib/jsr250-api-3.2.0.jar:lib/jta-1.0.1b.jar:lib/log4j-1.2.15.jar:lib/mail-1.4.5.jar:lib/management-api-3.1
.0.jar:lib/MSA-SOSA-1.2.2.jar:lib/NFVModel-1.0.0.jar:lib/nfvModel_config.jar:lib/oracle_jdbc.jar:lib/org.mortbay.jetty-5.1.11
RC0.jar:lib/org.mortbay.jmx-5.1.11RC0.jar:lib/oscache-2.4.1.jar:lib/policy-2.2.2.jar:lib/quartz-1.6.0.jar:lib/resolver.jar:li
b/resteasy-jaxb-provider-3.0.3.Final.jar:lib/resteasy-jaxrs-3.0.3.Final.jar:lib/resteasy-jettison-provider-3.0.3.Final.jar:li
b/RESTPA-1.0.0.jar:lib/saaj-api.jar:lib/saaj-impl.jar:lib/scannotation-1.0.3.jar:lib/servlet-api-5.1.0.jar:lib/stax-ex.jar:li
b/streambuffer-1.1.jar:lib/woodstox.jar:lib/wsdl4j-1.5.1.jar:lib/xercesImpl-2.6.2.jar:lib/xml-apis-2.0.2.jar:lib/xmlParserAPI
s-2.2.1.jar:lib/xsltc-1.1.jar com.hp.sosa.Main start conf/sosa.xml conf/sosa.dtd >log/sosa.stdout 2> log/sosa.stderr
HP Service Order Smart Adapter (SOSA) start action completed...
nfvdff /tmp $
```

12.2.5 Getting status of HP SOSA

To check the HP SOSA status, run the following command:

```
#nfv-director.sh -a status -c sosa
```

Alternate command:

```
/opt/OV/ServiceActivator/EP/SOSA/bin/sosa.sh test
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a status -c sosa
Testing sosa
Sosa is running
HP Service Order Smart Adapter (SOSA) status action completed...
nfvdff /tmp $
```

12.2.6 Stopping HP SOSA

To stop HP SOSA, run the following command:

```
#nfv-director.sh -a stop -c sosa
```

Alternate command:

```
/opt/OV/ServiceActivator/EP/SOSA/bin/sosa.sh stop
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a stop -c sosa
Stoping sosa
/usr/java/jdk1.6.0_37//bin/java -Dsosa -Dlog4j.configuration=properties/sosa-log4j.properties -classpath .::properties:conf:/
opt/HP/jboss/standalone/deployments/hpsa.ear/lib/sosa.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/mwfm.jar:/opt/HP/
jboss/standalone/deployments/hpsa.ear/lib/activator_utils.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/resmgr.jar:/o
pt/HP/jboss/standalone/deployments/hpsa.ear/lib/inventoryruntime.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/ep-uti
ls.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/lib/commons-codec-1.5.jar:/opt/HP/jboss/standalone/deployments/hpsa.ear/
lib/sosa-hibernate.jar:lib/activation-1.1.1.jar:lib/antlr-2.7.6.jar:lib/asm-1.5.3.jar:lib/axis-1.4.jar:lib/axis-ant-1.4.jar:l
ib/c3p0-0.9.1.2.jar:lib/cglib-2.1.3.jar:lib/commons-collections-3.2.1.jar:lib/commons-dbcp-1.3.jar:lib/commons-discovery-0.2.
jar:lib/commons-el-1.0.jar:lib/commons-lang-2.2.jar:lib/commons-logging-1.0.4.jar:lib/commons-pool-1.5.6.jar:lib/displaytag-1
.1.jar:lib/dom4j-1.6.jar:lib/dtdparser-121.jar:lib/edb_jdbc.jar:lib/gmbal-api-only-3.1.0.jar:lib/gson-2.2.2.jar:lib/ha-api-3.
1.8.jar:lib/hibernate-3.2.5-modif.jar:lib/inventory_NFVModel.jar:lib/jasper-compiler-1.0.jar:lib/jasper-runtime-1.0.jar:lib/j
avassist-3.8.0.GA.jar:lib/javax.servlet-5.1.11RC0.jar:lib/jaxb-api-2.2.4.jar:lib/jaxb-impl-2.2.4-1.jar:lib/jaxrpc-1.1.jar:lib
/jaxrs-api-3.0.3.Final.jar:lib/jaxws-api-2.2.jar:lib/jaxws-rt-2.2.5.jar:lib/jersey-client-1.4.jar:lib/jersey-core-1.4.jar:lib
/jettison-1.3.4.jar:lib/jox-improved-116.jar:lib/jsr173_api.jar:lib/jsr181-api.jar:lib/jsr250-api-3.2.0.jar:lib/jta-1.0.1b.ja
r:lib/log4j-1.2.15.jar:lib/mail-1.4.5.jar:lib/management-api-3.1.0.jar:lib/MSA-SOSA-1.2.2.jar:lib/NFVModel-1.0.0.jar:lib/nfvM
odel_config.jar:lib/oracle_jdbc.jar:lib/org.mortbay.jetty-5.1.11RC0.jar:lib/org.mortbay.jmx-5.1.11RC0.jar:lib/oscache-2.4.1.j
ar:lib/policy-2.2.2.jar:lib/quartz-1.6.0.jar:lib/resolver.jar:lib/resteasy-jaxb-provider-3.0.3.Final.jar:lib/resteasy-jaxrs-3
.0.3.Final.jar:lib/resteasy-jettison-provider-3.0.3.Final.jar:lib/RESTPA-1.0.0.jar:lib/saaj-api.jar:lib/saaj-impl.jar:lib/sca
nnotation-1.0.3.jar:lib/servlet-api-5.1.0.jar:lib/stax-ex.jar:lib/streambuffer-1.1.jar:lib/woodstox.jar:lib/wsdl4j-1.5.1.jar:
lib/xercesImpl-2.6.2.jar:lib/xml-apis-2.0.2.jar:lib/xmlParserAPIs-2.2.1.jar:lib/xsltc-1.1.jar com.hp.sosa.Main stop
HP Service Order Smart Adapter (SOSA) stop action completed...
nfvdff /tmp $

nfvdff /tmp $ ./nfv-director.sh -a status -c sosa
Testing sosa
Sosa is not running
HP Service Order Smart Adapter (SOSA) status action completed...
nfvdff /tmp $
```

12.2.7 HP Equipment connection pool

To start the HP Equipment connection pool (ECP), run the following command:

```
#nfv-director.sh -a start -c ecpool
```

Alternate Command:

```
/opt/OV/ServiceActivator/EP/ECP/bin/StartServer.sh
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a start -c ecpool
Starting RMI service com.hp.spain.connection.pool.server.RmiEcpService on rmi://127.0.0.1:1200/RmiEcpService
Saving pid in /opt/OV/ServiceActivator/EP/ECP/log/ecp.pid
Done. Check /opt/OV/ServiceActivator/EP/ECP/log for details.
HP Equipment connection pool (ECP) start action completed...
nfvdff /tmp $
```

12.2.8 Stopping HP ECP

To stop HP Equipment connection pool (ECP), run the following command:

```
#nfv-director.sh -a stop -c ecpool
```

Alternate Command:

```
/opt/OV/ServiceActivator/EP/ECP/bin/StopServer.sh
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a stop -c ecpool
Connecting to rmi://127.0.0.1:1200/
20140622-145338.366: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Finding rmi://127.0.0.1:1200/RmiEcpService
20140622-145338.461: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Service rmi://127.0.0.1:1200/RmiEcpService found: R
miEcpService_Stub[UnicastRef [liveRef: [endpoint:[127.0.0.1:1101](remote),objID:[-50ffd47e:146c39bc24b:-7ffe, 245780643084261
8081]]]]
20140622-145338.461: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Executing remote task with: {operation=shutdown}
20140622-145338.463: com.hp.spain.connection.pool.server.RmiEcpClient: 1: {result=The RMI Registry is up.}
20140622-145338.463: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Invoking service RmiEcpService
20140622-145338.782: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Return: Shutdown completed
HP Equipment connection pool (ECP) stop action completed...
nfvdff /tmp $
```

12.2.9 Getting status of HP ECP

To check HP Equipment connection pool (ECP) status, run the following command:

```
#nfv-director.sh -a status -c ecpool
```

Alternate Command:

```
/opt/OV/ServiceActivator/EP/ECP/bin/showStatus.sh
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a status -c ecpool
Connecting to rmi://127.0.0.1:1200/
20140622-144546.038: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Finding rmi://127.0.0.1:1200/RmiEcpService
20140622-144546.150: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Service rmi://127.0.0.1:1200/RmiEcpService found: R
miEcpService_Stub[UnicastRef [liveRef: [endpoint:[127.0.0.1:1101](remote),objID:[-50ffd47e:146c39bc24b:-7ffe, 245780643084261
8081]]]]
20140622-144546.150: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Executing remote task with: {}
20140622-144546.154: com.hp.spain.connection.pool.server.RmiEcpClient: 1: {result=The RMI Registry is up.}
20140622-144546.154: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Invoking service RmiEcpService
20140622-144546.156: com.hp.spain.connection.pool.server.RmiEcpClient: 1: [EMAPool, HSSPool, MNPPool]
20140622-144546.179: com.hp.spain.connection.pool.server.RmiEcpClient: 1: POOL_MANAGER_RUNNING = true
20140622-144546.179: com.hp.spain.connection.pool.server.RmiEcpClient: 1: Return: [{NOMBRE=EMAPool, ESTADO_POOL=AVAILABLE, PO
OL_BUSY=false, REQUEST_MANAGER_RUNNING=true, RESOURCE_MANAGER_RUNNING=true, SUBPOOLS=[{NOMBRE_SUBPOOL=0, ESTADO_SUBPOOL=AVAIL
ABLE, SUBPOOLINFO=telnet://hp@127.0.0.1:3337<BR>driver TemplateDriver<BR>busy 0; free 0; max 5; min 1, SESIONES=[{ES_SPY=fals
e, NOMBRE_SESION=1, ESTADO_SESION=DISCONNECTED, INACTIVE, CADENA_ACCION=EMAPool;0;1;}]}], PETICIONES=0, POOL_RUNNING=true}, {
NOMBRE=MNPPool, ESTADO_POOL=AVAILABLE, POOL_BUSY=false, REQUEST_MANAGER_RUNNING=true, RESOURCE_MANAGER_RUNNING=true, SUBPOOLS
=[{NOMBRE_SUBPOOL=0, ESTADO_SUBPOOL=AVAILABLE, SUBPOOLINFO=tcp://hp@127.0.0.1:8000<BR>driver TemplateDriver<BR>busy 0; free 0
; max 5; min 1, SESIONES=[{ES_SPY=false, NOMBRE_SESION=3, ESTADO_SESION=DISCONNECTED, INACTIVE, CADENA_ACCION=MNPPool;0;3;}]}
], PETICIONES=0, POOL_RUNNING=true}, {NOMBRE=HSSPool, ESTADO_POOL=AVAILABLE, POOL_BUSY=false, REQUEST_MANAGER_RUNNING=true, R
ESOURCE_MANAGER_RUNNING=true, SUBPOOLS=[{NOMBRE_SUBPOOL=0, ESTADO_SUBPOOL=AVAILABLE, SUBPOOLINFO=telnet://hp@127.0.0.1:23<BR>
driver TemplateDriver<BR>busy 0; free 0; max 5; min 1, SESIONES=[{ES_SPY=false, NOMBRE_SESION=2, ESTADO_SESION=DISCONNECTED,
INACTIVE, CADENA_ACCION=HSSPool;0;2;}]}], PETICIONES=0, POOL_RUNNING=true}]
HP Equipment connection pool (ECP) status action completed...
nfvdff /tmp $
```

12.2.10 Starting HP Lock Manager

To start HP Lock Manager, run the following command:

```
#nfv-director.sh -a start -c lockmgr
```

Alternate command:

```
/opt/OV/ServiceActivator/EP/LockManager/bin/StartServer.sh
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a start -c lockmgr
Starting RMI service com.hp.spain.lock.manager.RmiLockManagerService on rmi://localhost:1220/RmiLockManagerService
Saving pid in /opt/OV/ServiceActivator/EP/LockManager/tmp/lckmgr.pid
Done. Check /opt/OV/ServiceActivator/EP/LockManager/log for details.
HP Lock Manager start action completed...
nfvdff /tmp $
```

12.2.11 Getting status of HP Lock Manager

To Check HP Lock Manager Status, run the following command:

```
#nfv-director.sh -a status -c lockmgr
```

Alternate command:

```
/opt/OV/ServiceActivator/EP/LockManager/bin/showStatus.sh
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a status -c lockmgr
Connecting to rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:55:26,707 [main] INFO RmiLockManagerClient - Invoking service rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:55:26,834 [main] INFO RmiLockManagerClient - Return: Service running. Lock manager notificator running.
HP Lock Manager status action completed...
nfvdff /tmp $
```

12.2.12 Stopping HP Lock Manager

To stop HP Lock Manager, run the following command:

```
#nfv-director.sh -a stop -c lockmgr
```

Alternate command:

```
/opt/OV/ServiceActivator/EP/LockManager/bin/StopServer.sh
```

Following is a sample of the output:

```
nfvdff /tmp $ ./nfv-director.sh -a stop -c lockmgr
Connecting to rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:56:20,767 [main] INFO RmiLockManagerClient - Invoking service rmi://localhost:1220/RmiLockManagerService
2014-06-22 14:56:21,157 [main] INFO RmiLockManagerClient - Return: LockManager shutdown successfull
HP Lock Manager stop action completed...
nfvdff /tmp $
```

12.3 Assurance components

12.3.1 Starting Assurance Gateway

To start NFVD Assurance Gateway, run the following command:

```
#nfv-director.sh –a start –c nfvd-agw
```

Following is a sample of the output:

```
#./nfv-director.sh -c nfvd-agw -a start
Start HP Assurance Gateway daemon
Starting HP Assurance Gateway

HP NFVD Assurance Gateway start action completed...
#
```

12.3.2 Stopping Assurance Gateway

To stop NFVD Assurance Gateway, run the following command:

```
#nfv-director.sh –a stop –c nfvd-agw
```

Following is a sample of the output:

```
#./nfv-director.sh -c nfvd-agw -a stop
Stop HP Assurance Gateway daemon
{"outcome" => "success"}
HP Assurance Gateway Jboss Stopped........

HP NFVD Assurance Gateway stop action completed...
#
```

12.3.3 Getting status of Assurance Gateway

To check status of NFVD Assurance Gateway, run the following command:

```
#nfv-director.sh –a status –c nfvd-agw
```

Following is a sample of the output:

```
#./nfv-director.sh -c nfvd-agw -a status
HP Assurance Gateway application server is running

HP NFVD Assurance Gateway status action completed...
#
```

12.3.4 Restarting NFVD Assurance Gateway

To restart NFVD Assurance Gateway, run the following command:

```
#nfv-director.sh –a restart –c nfvd-agw
```

Following is a sample of the output:

```
#./nfv-director.sh -c nfvd-agw -a restart
Stop HP Assurance Gateway daemon
{"outcome" => "success"}
HP Assurance Gateway Jboss Stopped........

HP NFVD Assurance Gateway stop action completed...

Restarting HP NFVD Assurance Gateway...
Start HP Assurance Gateway daemon
Starting HP Assurance Gateway

HP NFVD Assurance Gateway start action completed...
```

12.3.5 Starting Open Mediation

To start Open Mediation, run the following command:

```
#nfv-director.sh -a start -c openmediation
```

Alternate command:

```
/opt/openmediation-V62/bin/nom_admin --start-container –
all
```

Following is a sample of the output:

```
#./nfv-director.sh -a start -c openmediation
Container instance number 0 has been STARTED.
HP Open Mediation start action completed...
```

12.3.6 Stopping Open Mediation

To stop Open Mediation, run the following command:

```
#nfv-director.sh -a stop -c openmediation
```

Alternate command:

```
/opt/openmediation-V62/bin/nom_admin --shutdown-container
–all
```

Following is a sample of the output:

```
#./nfv-director.sh -a stop -c openmediation
Container instance number 0 has been SHUTDOWN.
HP Open Mediation stop action completed...
```

12.3.7 Getting status of Open Mediation

To check status of Open Mediation, run the following command:

```
#nfv-director.sh -a status -c openmediation
```

Alternate command:

```
/opt/openmediation-V62/bin/nom_admin --list-container
```

Following is a sample of the output:

```
#./nfv-director.sh -a status -c openmediation
List of the containers:
0       STARTED      Hub
HP Open Mediation status action completed...
#
```

12.4 Monitoring components

12.4.1 Starting HP SiteScope

To start HP SiteScope, run the following command:

```
#nfv-director.sh –a start –c SiteScope
```

Alternate command:

```
/opt/HP/SiteScope/start
```

Following is a sample of the output:

```
#./nfv-director.sh -a start -c sitescope
SiteScope started as a background process
HP SiteScope start action completed...
#./nfv-director.sh -a status -c sitescope
Sitescope is running
HP SiteScope status action completed...
#
```

12.4.2 Stopping HP SiteScope

To Stop HP SiteScope, run the following command:

```
#nfv-director.sh –a stop –c SiteScope
```

Alternate command:

```
/opt/HP/SiteScope/stop
```

Following is a sample of the output:

```
#./nfv-director.sh -a stop -c sitescope
Stopped SiteScope process (18675)
Stopped SiteScope monitoring process (18691)
HP SiteScope stop action completed...
#
```

12.4.3 Getting status of HP SiteScope

To check status of HP SiteScope, run the following command:

```
#nfv-director.sh –a status –c SiteScope
```

Following is a sample of the output:

```
#./nfv-director.sh -a status -c sitescope
Sitescope is running
HP SiteScope status action completed...
#
```

12.4.4 Restarting HP SiteScope

To restart HP SiteScope, run the following command:

```
#nfv-director.sh –a restart –c SiteScope
```

Following is a sample of the output:

```
#./nfv-director.sh -a restart -c sitescope
Stopped SiteScope process (11365)
/opt/HP/SiteScope/start: line 55: 11365 Killed                  ./start-service -x $@ > /dev/null 2>&1
Stopped SiteScope monitoring process (11381)
HP SiteScope stop action completed...

Restarting HP SiteScope...
SiteScope started as a background process
HP SiteScope start action completed...
#
```

12.5 Automation components

12.5.1 Starting Postgres database

To start Postgres database, run the following command:

```
#nfv-director.sh -a start -c ppasdb
```

Alternate command:

```
/etc/init.d/ppas-9.2 start
```

Following is a sample of the output:

```
#./nfv-director.sh -a start -c ppasdb
Starting Postgres Plus Advanced Server 9.2:
waiting for server to start.... done
server started
Postgres Plus Advanced Server 9.2 started successfully
Postgres Plus Advanced Server 9.2 start action completed...
#./nfv-director.sh -a start -c ppasdb
Postgres Plus Advanced Server 9.2 start action completed...
#
```

12.5.2 Getting status of Postgres database

To check status of Postgres database, run the following command:

```
#nfv-director.sh -a status -c ppasdb
```

Alternate command:

```
/etc/init.d/ppas-9.2 status
```

Following is a sample of the output:

```
#./nfv-director.sh -a status -c ppasdb
pg_ctl: server is running (PID: 25745)
/opt/PostgresPlus/9.2AS/bin/edb-postgres "-D" "/opt/PostgresPlus/9.2AS/data"
Postgres Plus Advanced Server 9.2 status action completed...
```

12.5.3 Stopping Postgres database

To stop Postgres database, run the following command:

```
#nfv-director.sh -a stop -c ppasdb
```

Alternate command:

```
/etc/init.d/ppas-9.2 stop
```

Following is a sample of the output:

```
#./nfv-director.sh -a stop -c ppasdb
Stopping Postgres Plus Advanced Server 9.2:
waiting for server to shut down.... done
server stopped
Postgres Plus Advanced Server 9.2 stop action completed...
#
```

12.5.4 Startting HP UCA-EBC

To start HP UCA-EBC, run the following command:

```
#nfv-director.sh -a start -c uca-ebc
```

Alternate command:

```
su - uca /opt/UCA-EBC/bin/uca-ebc start
```

Following is a sample of the output:

```
#./nfv-director.sh -a start -c uca-ebc
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
Using UCA for EBC Home directory specified by the UCA_EBC_HOME environment variable: /opt/UCA-EBC
Using UCA for EBC Data directory specified by the UCA_EBC_DATA environment variable: /var/opt/UCA-EBC
Using UCA for EBC Instance directory : /var/opt/UCA-EBC/instances/default
*** INFO: Starting UCA for Event Based Correlation version 3.0
HP UCA-EBC start action completed...
*** INFO: UCA for Event Based Correlation version 3.0 started (pid=26042)
```

12.5.5 Getting status of HP UCA-EBC

To check the status of HP UCA-EBC, run the following command:

```
#nfv-director.sh -a status -c uca-ebc
```

Alternate command:

```
su - uca /opt/UCA-EBC/bin/uca-ebc show
```

Following is a sample of the output:

```
#./nfv-director.sh -a status -c uca-ebc
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO  - Server is running.
HP UCA-EBC status action completed...
```

12.5.6 Stopping HP UCA-EBC

To stop HP UCA-EBC, run the following command:

```
#nfv-director.sh -a stop -c uca-ebc
```

Alternate command:

```
su - uca /opt/UCA-EBC/bin/uca-ebc stop
```

Following is a sample of the output:

```
#./nfv-director.sh -a stop -c uca-ebc
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO  - Requesting Server Stop
*** INFO: UCA for Event Based Correlation version 3.0 stopped (0).
*** INFO: Stop completed
HP UCA-EBC stop action completed...
#
```

12.5.7 Starting HP UCA-automation

To start HP UCA-Automation, run the following command:

```
#nfv-director.sh -a start -c uca-atm
```

Alternate command:

```
/opt/UCA-ATM/bin/ucautomation-ui start
```

Following is a sample of the output:

```
#./nfv-director.sh -a start -c uca-atm
*** INFO: Starting UCA Automation Console
*** INFO: Unable to start UCA Automation Console server. Please check the logs.
HP UCA Automation Console start action completed...
```

12.5.8 Getting status of HP UCA-automation

To check status of HP UCA-Automation, run the following command:

```
#nfv-director.sh -a status -c uca-atm
```

Alternate command:

```
su - uca /opt/UCA-ATM/bin/ucautomation-ui show
```

Following is a sample of the output:

```
#./nfv-director.sh -a status -c uca-atm
*** INFO: UCA Automation Console is not running.
HP UCA Automation Console status action completed...
#
```

12.5.9 Stopping HP UCA-automation

To stop HP UCA-Automation, run the following command:

```
#nfv-director.sh -a stop -c uca-atm
```

Alternate command:

```
/opt/UCA-ATM/bin/ucautomation-ui stop
```

Following is a sample of the output:

```
#./nfv-director.sh -a stop -c uca-atm
*** INFO: UCA Automation Console is not running
HP UCA Automation Console stop action completed...
#./nfv-director.sh -a status -c uca-atm
```

Chapter 13

VNF monitoring

This section describes how to use the NFV Director to monitor resources.

NFV is a complex system that needs constant monitoring of the physical and the logical entities. The provisioning and monitoring functions can be brought together through rules that define manual or autonomous scaling and placement actions, based on measured key performance indicators (KPIs).

To resolve this, HP NFV Director includes the agent-less monitoring component. Built using HP SiteScope, it can monitor a wide variety of monitoring points, issuing events, or executing commands when predefined thresholds are crossed. Since different virtual network functions have different monitoring needs, the monitoring points and thresholds are automatically configured by HP NFV Director as the VNF is provisioned or modified. As an agent-less solution, HP NFV Director does not require the installation of monitoring agents on the target systems.

The following sections use SiteScope v11.23 for illustrating the product capabilities. Refer to its documentation for more details.

13.1 Accessing SiteScope

Use the following steps to access SiteScope.

1. To access SiteScope from a browser, enter the SiteScope address in a Web browser.

    The default address is: `http://<server name>:<port>/SiteScope`.

2. (Optional) To access SiteScope from the Start menu (Windows platforms only), select **Start** > **Programs** > **HP SiteScope** > **Open HP SiteScope**.

3. Enter the login credentials and click the **Login** button.

13.2 Overview of SiteScope dashboard

When you connect to SiteScope, default dashboard can be seen on successful login. For the first time SiteScope is deployed, there is a delay for initialization of the interface elements. Dashboard displays current performance data for the infrastructure elements being monitored by SiteScope and provides access to functions you use to define filters. Dashboard displays a table of groups and monitors for the elements highlighted in the monitor tree or listed in the path. You can double-click each group or monitor node to navigate to child nodes and monitors.

SiteScope dashboard window contains the following key elements:

*   Common toolbar—Provides access to page options, documentation, and additional resources. This toolbar is located on the upper part of the window.

*   Context toolbar—Contains buttons for frequently-used commands in the selected SiteScope context.

- Context tree—Enables you to create and manage SiteScope objects in a tree structure.

- Context buttons—Provide access to the SiteScope Monitors, Remote Servers, Templates, Preferences, Server Statistics, and Diagnostic Tools.
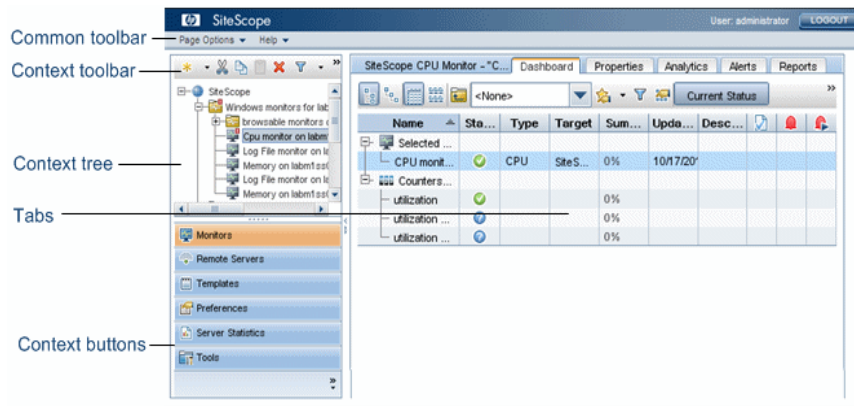
Figure 177: SiteScope dashboard

SiteScope monitoring provides a real-time picture of system availability and performance. You configure SiteScope monitors to collect metrics from a range of infrastructure components, including Web, application, database, and firewall servers. The status and metrics are then aggregated for presentation in SiteScope Dashboard.

Dashboard is linked to the SiteScope monitor tree hierarchy. The data displayed in Dashboard represents the selected context in the monitor tree. The highest level is the SiteScope node and any applicable monitor groups. The lowest-level element for display in a Dashboard view is an individual SiteScope monitor and its measurements.

Dashboard includes functions that you can use to customize the display of monitor information. This includes defining named filter settings to limit the display of data to those matching defined criteria. You can also select various data display options.

Dashboard also includes hyperlinks and menus that you can use to navigate through the hierarchy of monitor elements, manually run a monitor, disable monitors, and access alert definitions.

SiteScope Dashboard has the following context buttons that are available from the left pane:
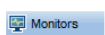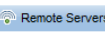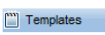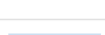
UI Element	Description
Monitors	Enables you to create and manage SiteScope groups and monitors in a hierarchy represented by a monitor tree.
Remote Servers	Enables you to set up the connection properties so that SiteScope can monitor systems and services running in remote Windows and UNIX environments.
Templates	Enables you to use templates to deploy a standardized pattern of monitoring to multiple elements in your infrastructure. You can use preconfigured SiteScope solution template or create and manage your own templates.
Preferences	Enables you to configure specific properties and settings related to most of the administrative tasks within SiteScope.
Server Statistics	Enables you to view key SiteScope server performance metrics.
Tools	Displays diagnostic tools that can help you troubleshoot problems in SiteScope and facilitate monitor configuration.

Figure 178: SiteScope Dashboard context buttons
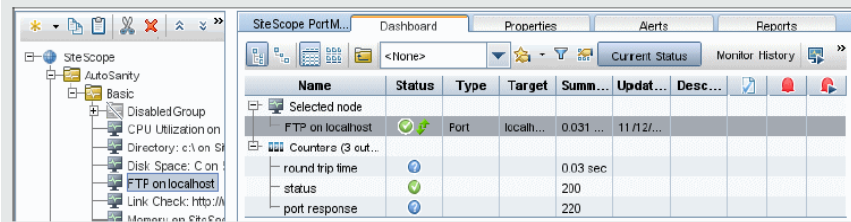
13.3 Analyzing data in SiteScope dashboard

This task describes the steps to analyze data in SiteScope Dashboard.

1. View monitor and measurement status and availability.

   When viewing SiteScope data in the Current Status view of Dashboard, you can explore the monitor tree to view monitor and measurement status and availability.
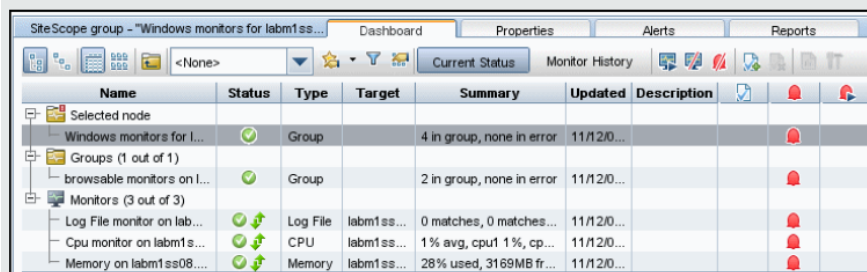
Figure 179: SiteScope Dashboard: view monitor status

2. View configured and triggered alerts.

   You can view data about alerts in the configured alerts and triggered alerts columns. If alerts are configured for a monitor, you can double-click the Configured Alert icon to see the list of configured alerts, and select an alert to view or edit the alert properties.

Figure 180: SiteScope Dashboard: view configured and triggered alerts

3. Disable monitors or monitors in group.

   Depending on the diagnosis, you can disable the monitor or monitors in group, or disable alerts associated with the monitor or group and continue to use the monitor.

4. Acknowledge monitors.

   To acknowledge monitor status, select a monitor or group and click the **Add Acknowledgment** icon or select **Add Acknowledgment** from the context menu, and enter the details in the **Acknowledge Monitors In Group** dialog box.
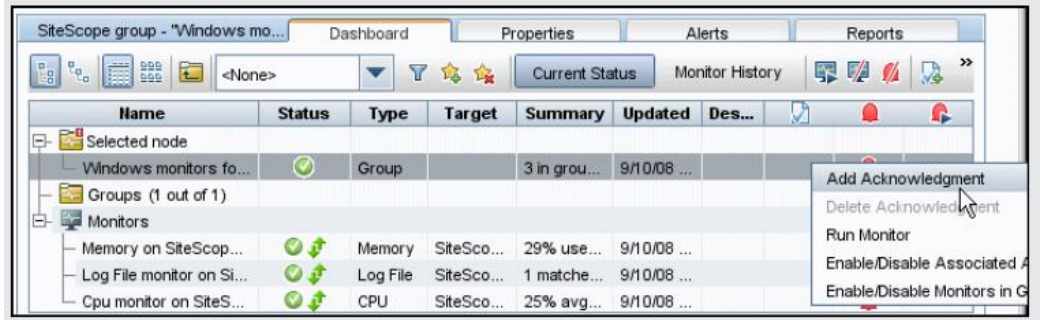
Figure 181:SiteScope Dashboard: Acknowledge monitors

5. Monitor your Microsoft Windows/UNIX server's resources.

   You can create a Microsoft Windows or UNIX Resources monitor to monitor your Windows or UNIX Server.

6. View monitor history.

   You enable and configure monitor history in the General Preferences. To view monitor history, click the Monitor **History** button in SiteScope Dashboard.
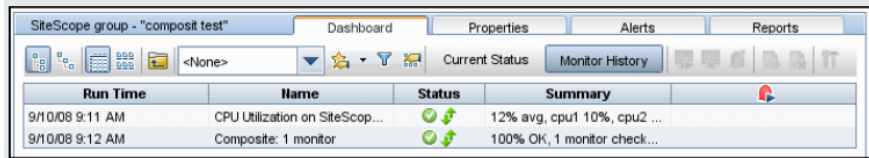
Figure 182: SiteScope Dashboard: view monitor history

13.4 Dashboard - status and availability levels

The following table provides details on different status and availability levels:

Icon	Description
✓	**Good Status**. All performance measurements are within the Good threshold level.
⚠	**Warning Status**. At least one performance measurement is within the Warning range, but no measurements are within the Error or Poor range.
✗	**Error/Poor Status**. At least one performance measurement is within the Error or Poor range. This indicates either of the following: • The performance measurement has a value, but at poor quality level. • There is no measurement value due to some error.
⊘	**Status Not Defined (No Data)**. There is no data for the group or monitor. This can be caused by any of the following reasons: • A new monitor has not yet run. • Monitor counters have not yet been collected. • The monitors on which the group or monitor depend are not reporting a Good condition.
?	**No Thresholds Breached Status**. No thresholds were defined for the monitor counter, so no status is assigned.

Figure 183: SiteScope Dashboard: status and availability levels

13.5 SiteScope templates and monitoring

SiteScope Templates provide an enterprise solution for standardizing the monitoring of different IT elements in your enterprise, including servers, applications, databases, network environments, and so on. You can use templates to rapidly deploy sets of monitors that check systems in the infrastructure that shares similar characteristics. You can create and customize your own templates to meet the requirements of your organization.

SiteScope templates are used to standardize a set of monitor types and configurations into a single structure. This structure can then be repeatedly deployed as a group of monitors targeting multiple elements in the network environments.

Templates speed up the deployment process of monitors across the enterprise through a single-operation deployment of groups, monitors, alerts, remote servers, and configuration settings.

Note

• Make sure that the monitor-run frequency is always greater than the time taken to scale-in/scale-out a VNF. Otherwise, multiple scale-in/scale-out requests might be sent for a single scale-in/out condition.

• In Fulfillment artifact templates, each Monitor artifact should be associated with a separate Monitor Handler artifact (*even* if the handler/hypervisor is the same). One-to-one mapping should be present between a Monitor artifact and a Monitor Handler artifact.

The following table describes the objects used in templates:

Icon	Object Type	Description
	Template Container	A template container enables you to manage your template monitoring solutions. You can add a template to a template container only.
	Template	The template contains the SiteScope group, monitors, remote servers, variable definitions, and alerts that make up the template monitoring solution.
	Template Variable	A variable is used to prompt for user input during template deployment. Template variables are either user-defined or predefined system variables.
	Template Remote Server	A template remote server is used to define Windows or UNIX remote server preferences that are created when the template is deployed.
	Template Group	A template group contains the template monitors and associated alerts. You use template groups to manage the deployment of monitors and associated alerts in your infrastructure.
	Template Monitor	Template monitors are used to define monitors that are created when the template is deployed.
	Template Alert	Template alerts are used to define alerts on groups and monitors that are created when the template is deployed. If an alert has been set up for the template monitor or group, the alert symbol is displayed next to the monitor or group icon.

Table 5: SiteScope template objects

SiteScope provides template examples for monitoring in Windows and UNIX environments. These templates are available from the Template Examples folder in the template tree. You can use the template examples to help you use SiteScope templates.

The following example shows the Windows basic template. The template contains a template group, Windows monitors for %%host%%, two template monitors (CPU and Memory), four user-defined variables (host, user, password, and frequency), and a template remote server.
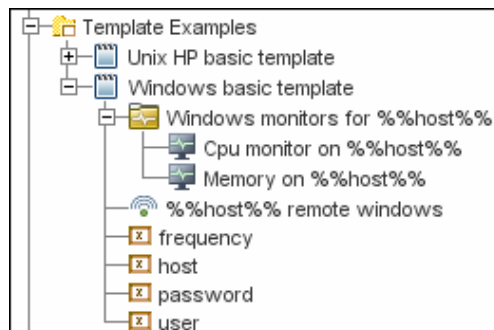
Figure 184: SiteScope: sample template

For deploying a monitor template path is a very essential input which decides which KPI has to be monitored. In the following example, the Template Path for CPU is NFVDirector/VM_Monitors/KVM_VM/CPU. After triggering the respective template for

deployment with the associated variables, it moves to Deployed state and this monitor can be accessed from Monitors Context.
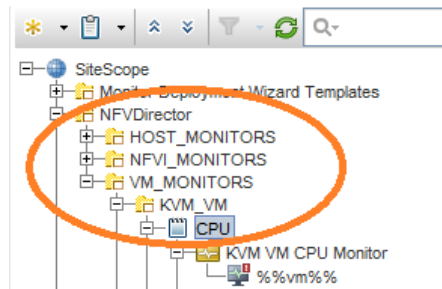
Figure 185: SiteScope: monitor context

Note

When configuring variables for Frequency and Error frequency in the Monitor Run Settings, the variable values can only be in time units of seconds.

When a monitor is copied or moved from one template to another, any user-defined variables in the monitor are also copied or moved.

The following table shows KPI's supported matrix for various hypervisors which comes by default as part of NFVD. Boxes marked in pale yellow indicate those KPI's are not supported for respective hypervisors.

	VMware ESXi		VMware VCenter		KVM		Openstack
	Host	*VM*	*Host*	*VM*	*Host*	*VM*	*VM*
CPU	✓	✓	✓	✓	✓	✓	✓
DiskRead	✓	✓	✓	✓	--	✓	✓
DiskWrite	✓	✓	✓	✓	--	✓	✓
Memory	✓	✓	✓	✓	✓	✓	--
NetworkRx	✓	✓	✓	✓	--	✓	✓
NetworkTx	✓	✓	✓	✓	--	✓	✓

Table 6: KPI's supported matrix for various hypervisors

The following table is the KPIs and counters supported matrix for various hypervisors with units which comes by default as part of NFVD.

Monitor	Counters	Hypervisor					Unit	Details
		VMWare		KVM		Openstack		
		Host/ VCenter	VM	Host	VM	VM		
CPU	cpu_usage_average	✓	✓	✓	✓	✓	%	Percentage of CPU used
Memory	memory_usage_average	✓	✓	✓	✓	✗	%	Percentage of Memory used
DiskRead	disk_read_rate_average	✓	✓	✗	✗	✗	kB/s	Average number of kilobytes read from disk
	disk_read_requests	✗	✗	✗	✓	✓	request	Disk read requests
DiskWrite	disk_write_rate_average	✓	✓	✗	✗	✗	kB/s	Average number of kilobytes written to disk
	disk_write_requests	✗	✗	✗	✓	✓	request	Disk write requests
NetworkRx	network_bytes_received	✗	✗	✗	✓	✗	bytes	Network bytes received
	network_packets_received	✗	✗	✗	✗	✓	packet	Number of incoming packets
	network_data_rx_rate_average	✓	✓	✗	✗	✗	kB/s	Average rate at which data was received
NetworkTx	network_bytes_transmitted	✗	✗	✗	✓	✗	bytes	Network bytes transmitted
	network_packets_transmitted	✗	✗	✗	✗	✓	packet	Number of outgoing packets
	network_data_tx_rate_average	✓	✓	✗	✗	✗	kB/s	Average rate at which data was transmitted

Figure 186: KPIs and counters supported matrix for various hypervisors

The following is a list of out-of-the-box monitors supported by NFVD. These monitors support infrastructure monitoring for KVM and VMWare. You can use these monitors

for checking the overall performance of the respective infrastructure. You can find these templates under `NFVDirector/NFVI_MONITORS` in Sitescope Templates context. For details on the usage of these monitors, refer to the Creating custom templates section.

Monitors	Hypervisor
Infrastructure Performance	KVM
Infrastructure Performance	VMWare

Table 7: Monitors supported by NFVD

The following is a Condition Expression support matrix, which can be associated with a monitor artifact. If a monitor artifact is associated with only a single condition artifact, the other conditions are associated with that monitor artifact using the default values for v1 release.

Counter names used in the expression should match the ones from the KPI's and counters supported matrix. A sample of usage for a CPU error condition is `cpu_usage_average > 90`.

Condition Type	Operator supported
ERROR	>
WARNING	>
GOOD	<

Table 8: Counter conditions

13.5.1 Creating custom templates

Custom templates broaden the capabilities of the regular SiteScope monitors other than the NFVD supported monitors. They help in tracking availability and performance of monitored environments. The custom templates enable you to create your own monitor by using any existing monitors provided by Sitescope.

1.   Select the templates context.

2.   Right-click the SiteScope root node from the tree and select **New** > **Template Container**.
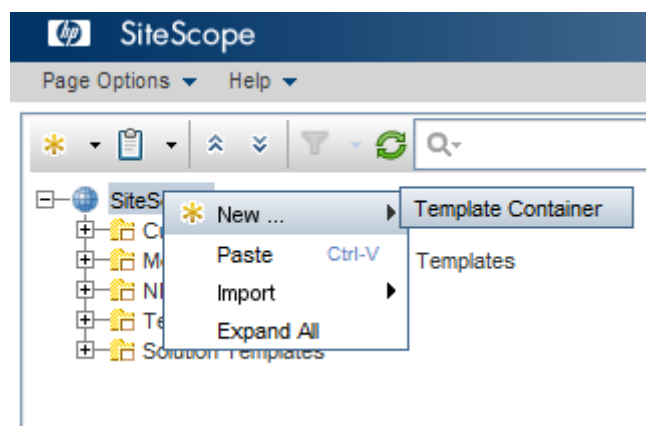
Figure 187: SiteScope: Create custom templates

3.   Enter the name of the template container and click the **OK** button.

4.   Right-click this new template container node and select **New** > **Template**.
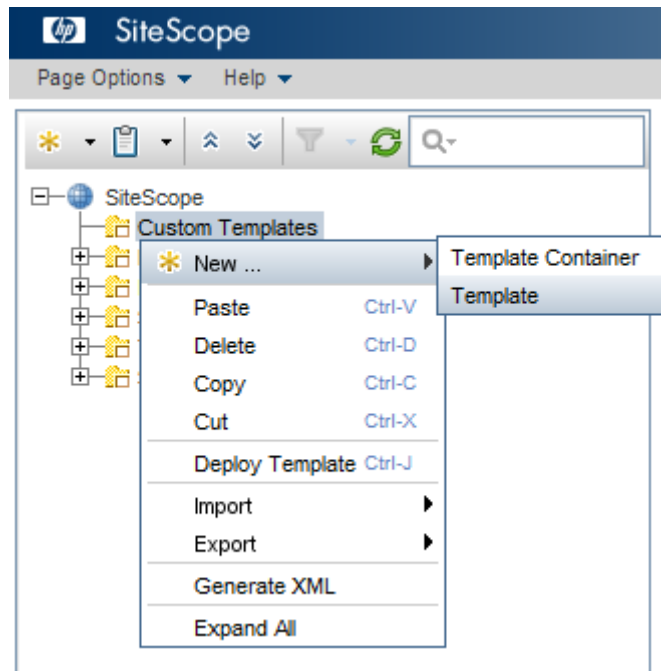
Figure 188: SiteScope: new template

5. Enter the name of the template and click the **OK** button.

   In the example provided in this section, a template container is created with the name Custom Templates.

6. Right-click this new template node and select **New** > **Group**.
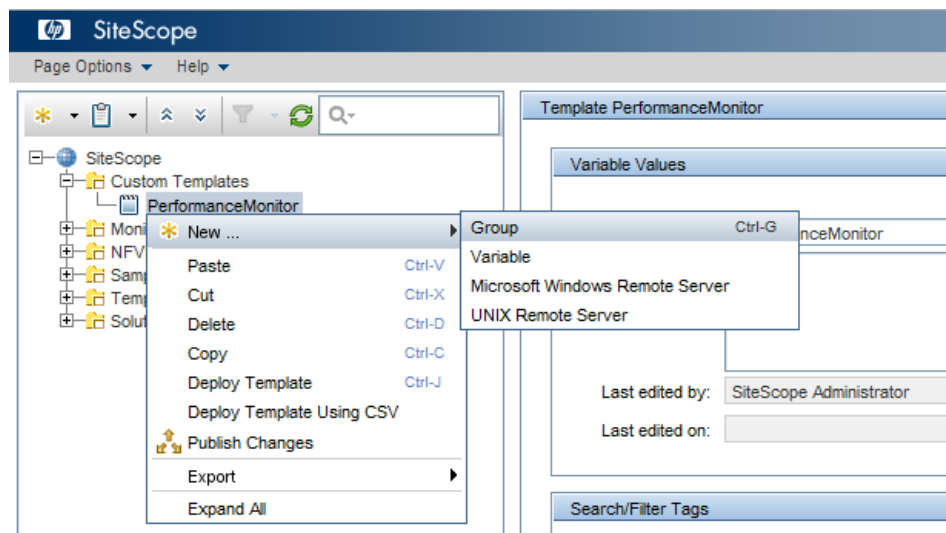
Figure 189: SiteScope: new group

7. Enter the name of the group and click the **OK** button.

8. In the example provided in this section, a template is created with name PerformanceMonitor.

9. Right-click this group node and select **New** > **Monitor**.
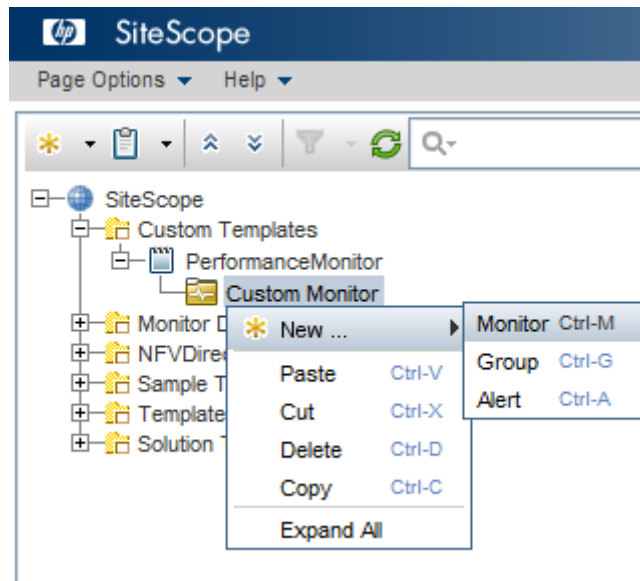
Figure 190: SiteScope: new Monitor

10. Select any one of the monitors of your choice from the list.

11. In the example provided in this section, a group is created with the name Custom Monitor.

12. Right-click the PerformanceMonitor template node and select **New** > **Variable**.
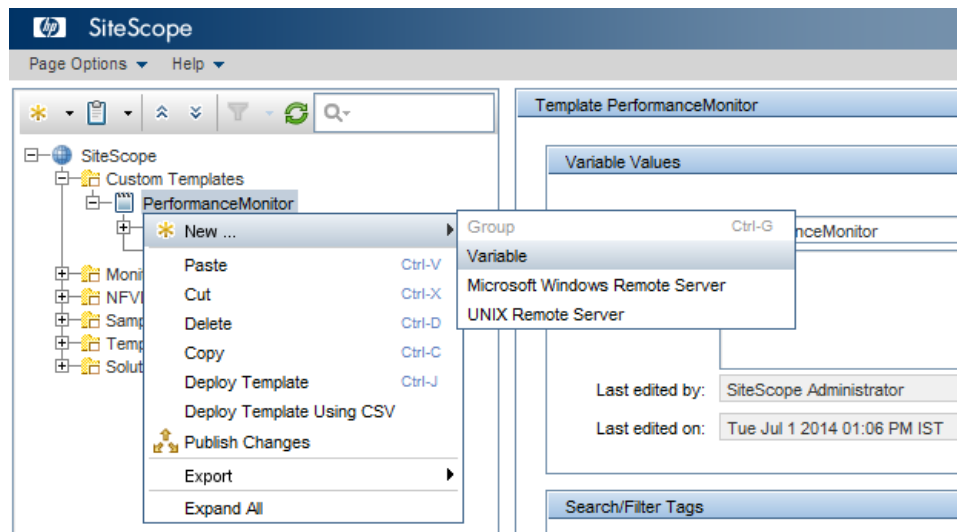
Figure 191: SiteScope: new Variable

The following window opens.
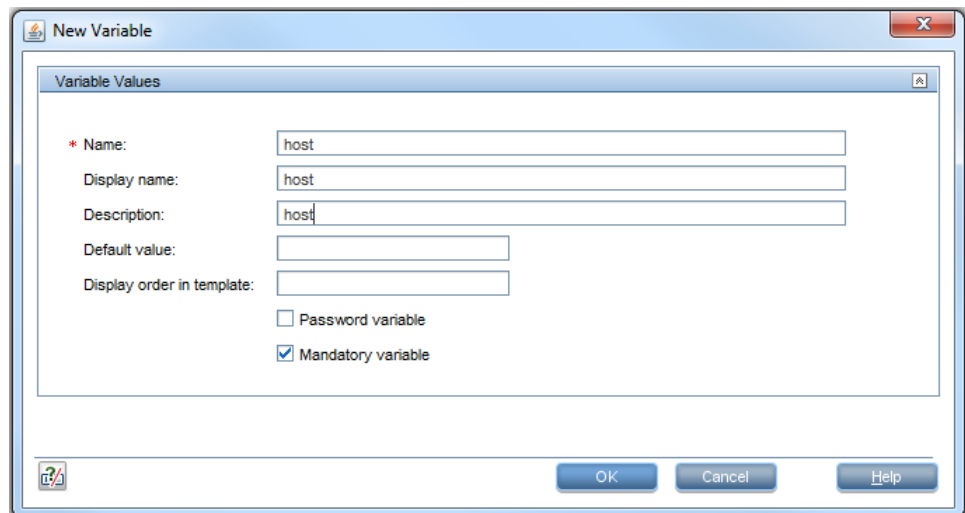
Figure 192: SiteScope: new Variable details

13. Enter the details for configuring a variable to associate it with the template.

In the following example, the host variable is configured.

After the variable is configured, it appears in the tree under the Template node.

You can configure any number of variables. The following example shows how to use these variables in the monitor depicted.
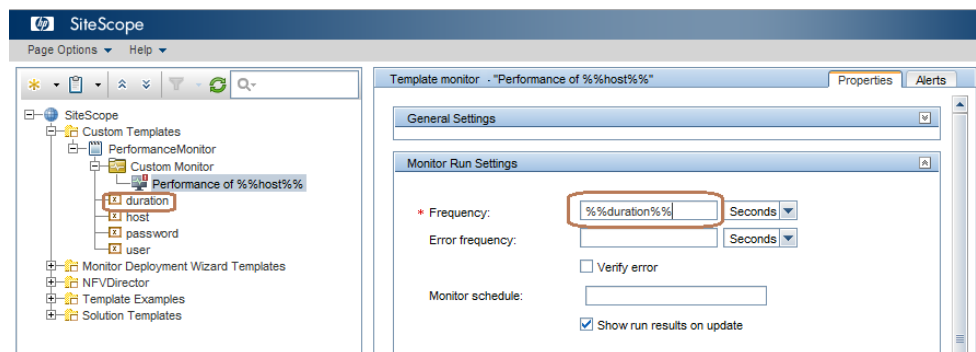
Figure 193: SiteScope: enter variable to associate with template

Following is the complete hierarchy of the Custom Template created.
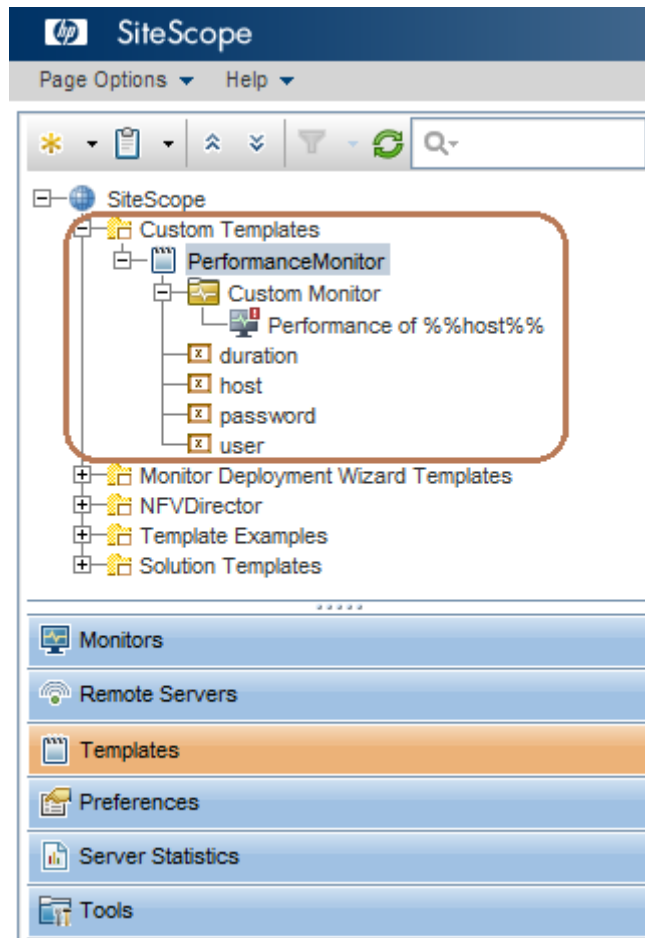
Figure 194: SiteScope: hierarchy of custom template

13.5.1.1 Associating Monitor Artifact to Monitor Handler Artifact

After you create Custom templates, associate the artifacts for deploying successfully.

1. Enter the full monitor name.

2. Enter the Template path up to the group level.

   An example of Monitor artifact:

```
<category>
        <label>GENERAL</label>
        <version>1</version>
        <order>1</order>
        <attributes>
                <attribute>
                        <label>Name</label>
                        <type>
                                <label>TEXT</label>
                        </type>
                        <unit>TEXT</unit>
                        <value> PerformanceMonitor </value>
                        <mandatory>true</mandatory>
                </attribute>

                <attribute>
                        <label>TemplatePath</label>
                        <type>
                                <label>TEXT</label>
                        </type>
                        <unit>TEXT</unit>
                         <value> Custom Templates/PerformanceMonitor/Custom
Monitor<value/>
```

```
                    <mandatory>false</mandatory>
                </attribute>
            </attributes>

</category>
```

3. Associate the Monitor artifact to the Monitor Handler artifact.

4. Create Monitor Argument artifacts with the exact variable name and variable value.

5. Sample of the MonitorArgument artifact.

```
<category>
        <label>GENERAL</label>
        <version>1</version>
        <order>1</order>
            <attributes>
                <attribute>
                    <label>Name</label>
                    <type>
                        <label>TEXT</label>
                    </type>
                    <unit>TEXT</unit>
                    <value>host</value>
                    <mandatory>true</mandatory>
                     <order>1</order>
                </attribute>
                <attribute>
                    <label> Value</label>
                    <type>
                        <label>TEXT</label>
                    </type>
                    <unit>TEXT</unit>
                    <value> hostname</value>
                    <mandatory>true</mandatory>
                    <order>2</order>
                </attribute>
            </attributes>
</category>
```

6. Associate the Monitor Argument artifacts to the Monitor Handler artifact.

13.6 Alerts section

SiteScope alerts are notification actions that are triggered when the conditions for the alert definition are detected. You use an alert to send some notification of an event or change of status in some element or system in your infrastructure. For example, an alert can be triggered when a SiteScope monitor detects a change from Good to Error indicating that the monitored system has stopped responding.

- Alerts can be of three types namely Error, Warning, and Good.

- Error alert will be triggered on breach of error threshold condition and will be sent to the configured SNMP target. An error alert will be sent to the destination target only if the threshold breach has occurred at least 4 times.

- Warning alert will be triggered on breach of warning threshold condition and will be sent to the configured SNMP target. A warning alert will be sent to the destination target only if the threshold breach has occurred at least 4 times.

- Good alert will be triggered on meeting normal/safe threshold condition and will be sent to the configured SNMP target. A good alert will be sent to the destination target only if the monitored entity was previously in error condition.

- If an alert is defined for a monitor, then it is activated on that monitor only. If an alert is defined for a template, then it is activated for all the monitors in the template.

- SNMPTarget has to be configured in Preferences section. Destination address and port have to be configured to map to the endpoint to where alerts have to be sent.

13.7 Configuring an alert

This task describes the steps involved in configuring an alert definition.

13.7.1 Prerequisites

Only a SiteScope administrator user or a user granted the appropriate alerts permissions can view, create, or edit alerts.

13.7.2 Creating/copying an alert

You can create a new alert or copy an existing alert into any group or monitor container in the SiteScope tree.

13.7.2.1 Creating a new alert

1. Right-click the container to which you want to associate the alert and select **New** > **Alert**.
2. Enter a name for the alert.
3. Select the targets to trigger the alert.
4. Configure an alert action.

   In the **Alert Actions** panel, click the **New Alert Action** to start the **Alert Action** wizard.

13.7.2.2 Copying an Alert Definition

1. In the **Alerts** tab, select the alert you want to copy.
2. Copy and paste it into the desired group or monitor container.

   The alert target automatically changes to the group or monitor into which the alert is copied.

Note

If you copy an alert definition from one group container to another, the Alert targets for the pasted alert are automatically reset to include all of the children of the container into which the alert is pasted. After pasting an alert, edit the alert definition properties to be sure that the assigned Alert targets are appropriate to the new alert context and your overall alerting plan.

13.7.2.3 Testing the alert

1. Select the alert in the Alerts tab of the monitor tree.
2. Click **Test**.
3. Select the monitor instance you want to test and click **OK**.

   A dialog box opens with information about the alert test.

The monitor you select does not have to be reporting the same status category that is selected to trigger the alert to test the alert. For example, the monitor does not have to currently be reporting an error to test an alert that is triggered by error conditions.

13.7.2.4 Disabling an alert - optional

You can disable alerts from the Alerts tab.

1. Select the alerts that you want to disable.
2. Click the **Disable** button.

   Alerts disabled from the Alerts tab cannot be triggered; this overrides the

   associated alerts status set for a monitor in the monitor Properties tab or

   Dashboard.

13.8 SNMP preferences

You use SNMP Preferences to configure the settings SiteScope needs to communicate with an external SNMP host or management console. These are the default SNMP parameters for use with SNMP Trap alerts.

To access, select **Preferences context** > **SNMP Preferences**.

You must be an administrator in SiteScope, or a user granted View SNMP lists permissions to be able to view SNMP Preferences.

SNMP Preferences enable you to define settings that are used by SiteScope SNMP Trap alerts when sending data to management consoles. It also enables you to define SNMP Trap receivers, and listen to multiple local addresses and ports at the same time. SiteScope uses the SiteScope SNMP Trap Alert type to integrate with SNMP-based network management systems.

User interface elements are described below:

UI Element	Description
General Settings	
Name	Name string assigned to the setting profile when creating a new SNMP recipient.
Description	Description for the setting profile, which appears only when editing or viewing its properties. You can include HTML tags such as the , <HR>, and <B> tags to control display format and style.
	Note: HTML code entered in this box is checked for validity and security, and corrective action is taken to fix the code (for example, code is truncated if it spans more than one line). If malicious HTML code or JavaScript is detected, the entire field is rejected. The following is prohibited HTML content:
	• Tags: **script**, **object**, **param**, **frame**, **iframe**.
	• Any tag that contains an attribute starting with **on** is declined. For example, onhover.
	• Any attribute with **javascript** as its value.
Preferences Settings: Main Settings Area	
Send to host	Domain name or IP address of the machine that receives all SNMP trap messages. This machine must be running an SNMP console to receive the trap message.
	Examples: snmp.mydomain.com or 206.168.191.20.
SNMP port	SNMP port to which the trap is sent.
	Default value: 162

Table 9: SNMP User Interface Elements

Preferences Settings: SNMP Connection Settings Area	
UI Element	**Description**
Timeout (seconds)	Amount of time, in milliseconds, to wait for the SNMP trap requests (including retries) to complete. **Default value:** 5
Number of retries	Number of times each SNMP trap GET request should be retried before SiteScope considers the request to have failed. **Default value:** 1
Community	Default SNMP community name used for sending traps. The community string must match the community string used by the SNMP management console. **Default value:** public
SNMP version	Default SNMP protocol version number to use. SNMP V1 and V2c are currently supported. **Default value:** V1
Authentication algorithm	Authentication algorithm used for SNMP V3. You can select MD5, SHA, or None. **Note:** This field is available only if SNMP V3 is selected.
User name	User name to be used for authentication if you are using SNMP version 3. **Note:** This field is available only if SNMP V3 is selected.
Password	Password to be used for authentication if you are using SNMP version 3. **Note:** This field is available only if SNMP V3 is selected.

Table 10: SNMP Preference Settings

Preferences Settings: Advanced Settings Area	
UI Element	**Description**
SNMP trap ID	Select the type of trap to send. There are several predefined ID types for common conditions: • **Generic SNMP trap ID.** Select a generic SNMP type from the drop-down list. • **Enterprise-Specific SNMP trap ID.** To use an enterprise specific SNMP ID type, enter the number of the specific trap type in the box. **Note:** When integrating SiteScope with NNMi, you must select **Enterprise-Specific SNMP trap ID**, and enter **1**. SiteScope sends a different notification ID for each SNMP version: • SNMP V1: .1.3.6.1.4.1.11.15.1.4 • SNMP V2: .1.3.6.1.4.1.11.15.1.4.1
SNMP object ID	Identifies to the console the object that sent the message. • **Preconfigured SNMP object IDs.** Select one of the predefined objects from the drop-down list. • **Other SNMP object ID.** To use another object ID, enter the other object ID in the box. **Note:** • In SiteScope version 11.20 and later, all logged traps have an object ID that starts with a dot ("."). For example, oid= .1.3.6.1.2.1.0.1.3.6.1.4.1.11.2.17.1. • When integrating SiteScope with NNMi, select **Preconfigured SNMP object IDs** and choose **HP SiteScope Event** from the list.
Add System OID as a prefix to SNMP Trap	Adds the default system OID (1.3.6.1.2.1) as a prefix to all SNMP Trap OIDs. Clear the check box if you do not want to use this prefix. **Default value:** Selected
SNMP source	The SNMP trap source: SiteScope Server or the monitor target server. **Default value:** Monitored Host

Table 11: SNMP Preferences Advanced Settings

13.9 Sending SiteScope Alerts

SiteScope triggers the alert as soon as any monitor it is associated with matches the alert trigger condition.

The following examples illustrate how different alert configurations send alerts after the error condition has persisted for more than one monitor run. If a monitor runs every 15 seconds and the alert is set to be sent after the third error reading, the alert is sent 30 seconds after the error was detected. If the monitor run interval is once every hour with the same alert setup, the alert is not sent until 2 hours later.

Example 1 - Always, after the condition has occurred at least N times:

Example 1a. An alert is sent for each time monitor is in error after condition persists for at least three monitor runs. Compare this with Example 1b below.

Alert setup	Always, after the condition has occurred at least 3 times								
sample interval	0	1	2	3	4	5	6	7	8
status	✓	✗	✗	✗	✗	✗	✓	✗	✗
count	c=0	c=1	c=2	c=3 alert!	c=4 alert!	c=5 alert!	c=0	c=1	c=2

Example 1b. An alert is sent for each time monitor is in error after condition persists for at least three monitor runs. Shows how the count is reset when the monitor returns one non-error reading between consecutive error readings. Compare this with Example 1a above.

Alert setup	Always, after the condition has occurred at least 3 times								
sample interval	0	1	2	3	4	5	6	7	8
status	✓	✗	✗	✓	✗	✗	✗	⚠	✓
count	c=0	c=1	c=2	c=0	c=1	c=2	c=3 alert!	c=0	c=0

Figure 195: SiteScope: send alerts always

Example 2 - Once, after the condition has occurred exactly N times:

An alert is sent only once if monitor is in error for at least three monitor runs, regardless of how long the error is returned thereafter.

Alert setup	Once, after the condition has occurred exactly 3 times								
sample interval	0	1	2	3	4	5	6	7	8
status	✓	✗	✗	✗	✗	✗	✗	✗	✗
count	c=0	c=1	c=2	c=3 alert!	c=4	c=5	c=6	c=7	c=8

Example 3 - Initially, after X times, and repeat every Y times:

Example 3a. An alert is sent on the fifth time monitor is in error and for every third consecutive error reading thereafter. Compare this with Example 3b below.

Alert setup	Initially, after 5 times, and repeat every 3 times								
sample interval	0	1	2	3	4	5	6	7	8
status	✓	✗	✗	✗	✗	✗	✗	✗	✗
count	c=0	c=1	c=2	c=3	c=4	c=5 alert!	c=6	c=7	c=8 alert!

Example 3b. An alert is sent on the third time monitor is in error and for every fifth consecutive error reading thereafter. Compare this with Example 3a above.

Alert setup	Initially, after 3 times, and repeat every 5 times								
sample interval	0	1	2	3	4	5	6	7	8
status	✓	✗	✗	✗	✗	✗	✗	✗	✗
count	c=0	c=1	c=2	c=3 alert!	c=4	c=5	c=6	c=7	c=8 alert!

Figure 196: SiteScope: send alert once

13.10 User management preferences

You can manage SiteScope user accounts from the User Management Preferences page. This page enables you to administer the users that are allowed access to SiteScope.

1. To access, select **Preferences context** >**User Management Preferences**.

2. In the **User Management Preferences** page, click the arrow next to the **New User** [icon] button and select **New User.**

3. In the **Main Settings** panel, enter the user name, login name, and password, and select the groups that can be accessed by this user profile.

4. Select the permissions to be granted to this user from the **Permissions** panel, or use the default permissions.

   All permissions are granted except the **Add, Edit** or **Delete** user permissions.

5. Click **OK**.

   The new user profile is added to the **User Management Preferences** list.

The following table provides UI descriptions of the **User Management Preferences** page.

UI Element	Description
[icon]	**New.** Click the arrow next to the button, and select: • **New User.** Creates a new user profile. • **New User Role.** Creates a new user role profile.
[icon]	**Edit.** Enables editing the selected user or user role profile.
[icon]	**Delete User/User Role.** Deletes the selected user or user role profiles.
[icon]	**Copy to User Role.** Enables coping an existing SiteScope user's permissions to a new user role. **Note:**SiteScope users still need to have a user login and a security group assigned to them on the LDAP server. (LDAP users have their own LDAP user name and password for logging on to SiteScope.)
[icon]	**Select All.** Selects all listed user and user role profiles.
[icon]	**Clear Selection.** Clears the selection.

Table 12: User Management Preferences

Note

Only an administrator in SiteScope or a user with add, edit, or delete user preferences permissions can create or make changes to user settings and permissions for the current user or for other users.

By default, a regular user does not have Add, edit or delete user preferences permissions, which means that they can view their own user properties only.

13.11 Managing certificates

When monitoring a remote server, if the target server uses a self-signed certificate, the certificate must be added to a trusted keystore. If you are monitoring a URL, a VMware-based server, a WebSphere Application Server, or a secure connection, you can manage self-signed certificates from the **Certificate Management** page.

Use the following procedure to import certificates:

1. Select **Preferences context** > **Certificate Management**.

2. To add certificates, click the **Import Certificates** button.

The **Import Certificates** dialog box appears.

3. Select **File** or **Host** and enter the details of the source server.

4. From the **Loaded Certificates** table, select the server certificates that you want to import and click **Import**.

   The imported certificates are listed on the Certificate Management page.

5. To view certificate details, double-click a certificate.

Note

To view the **Certificate Management** page, you must be an administrator in SiteScope or a user granted with View certificates list permissions.

Chapter 14

Correlation and autonomous actions

The correlation and autonomous actions correspond to correlating traps received from NFV source and taking autonomous actions based on the NFV topology. Correlation is possible when the collection event is received from various NFV sources at the UCA EBC value pack. The event collection to UCA EBC is possible depending on the generic SNMP channel adaptor configuration.

14.1 Correlation and autonomous process
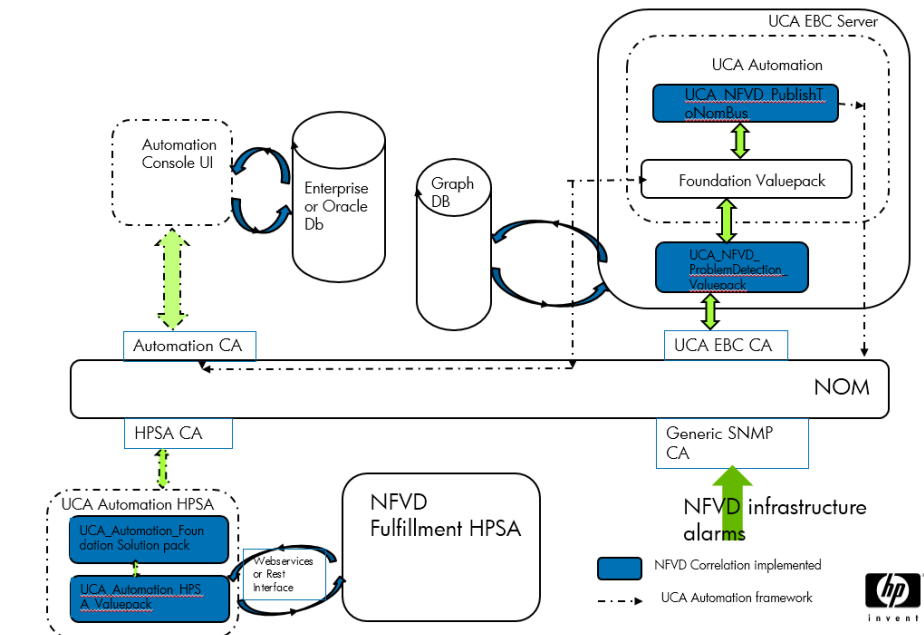
Figure 197: Correlation and autonomous action process diagram

Correlation and Autonomous actions can be categorized into the following:

- Alarm Enrichment

- Autonomous action

- Status of action and reporting

14.1.1 Alarm enrichment

The alarms received from the NFV source should be enriched with NFV topology information and parameters required for autonomous actions. The UCA NFVD Problem detection valuepack that is delivered with the NFV Director 1.0 enriches alarms. Depending on the action, it fetches the required parameters and finally publishes the alarms to Open Mediation.

The following is a sample of a raw alarm received from the NFV Director source.

```
<AlarmCreationInterface xmlns="http://hp.com/uca/expert/x733Alarm">
   <identifier>5:65d95213-00fd-4764-adfa-86b00af0881a</identifier>
          <sourceIdentifier>NFVD_Source</sourceIdentifier>
    <alarmRaisedTime>2014-06-23T11:55:00Z</alarmRaisedTime>
    <targetValuePack>SNMP-Customization-SiteScope-FlowTarget</targetValuePack>
   <originatingManagedEntity>KVM_TestVM</originatingManagedEntity>
   <originatingManagedEntityStructure>
    <classInstance instance="SiteScope\HP\KVM VM CPU Monitor\KVM_TestVM"
clazz="Generic Hypervisor"/>
   </originatingManagedEntityStructure>
   <alarmType>QUALITY_OF_SERVICE_ALARM</alarmType>
   <probableCause>calculatedCounterValue1 == 1 error</probableCause>
   <perceivedSeverity>WARNING</perceivedSeverity>
   <networkState>NOT_CLEARED</networkState>
   <operatorState>NOT_ACKNOWLEDGED</operatorState>
   <problemState>NOT_HANDLED</problemState>
   <specificProblem>cpu_usage_medium: 1</specificProblem>
          <additionalText>_customPropertiesValues=|_httpPort=8888|
                  _webserverAddress=15.154.72.226|
          alertHelpURL=http://15.154.72.226:8080/SiteScope/sisdocs/doc_lib/index.htm?single=f
alse&amp;context=system_avail&amp;topic=config_sis_alert|
                  diagnosticTraceRoute=|errorOnly=|goodOnly=cpu_usage_high: 0
cpu_usage_low: 0|FullGroupId=HP: KVM VM CPU Monitor|group=KVM VM CPU Monitor|
                  groupdescription=CPU |
                  groupID=201087530|
                  id=&amp;lt;id&amp;gt;|
                 mainStateProperties= groupID: 201087530
                 calculatedCounterValue1: 0
                 calculatedCounterValue2: 1
                 calculatedCounterValue3: 0|
          monitorDrilldownUrl=http://ossvm8.ind.hp.com:8080/SiteScope/servlet/Main?activeid=2
01087531&amp;activerighttop=dashboard&amp;view=new&amp;dashboard_view=Details&amp;
dashboard_model=true&amp;sis_silent_login_type=encrypted&amp;login=%28sisp%29knjxbqDE
SkAn5mKcvgTmj%2FyFwHH5Ke3m&amp;password=%28sisp%29EzqXbIXEFD%2BJbE1N1T%
2FZlELjja0DKaN7|

          monitorServiceId=SiteScopeMonitor:201087530:201087531|
          monitorTypeDisplayName=Generic Hypervisor|
          monitorUUID=9a145576-cefb-483f-beaa-bd3bd6f9158f|
          mountName=[/dev/mapper/VolGroup00-root_vol, /dev/mapper/VolGroup00-root_vol (/),
/dev/sda1, /dev/sda1 (/boot), /dev/mapper/VolGroup00-home_vol, /dev/mapper/VolGroup00-
home_vol (/home), /dev/mapper/VolGroup00-opt_vol, /dev/mapper/VolGroup00-opt_vol (/opt),
/dev/mapper/VolGroup00-tmp_vol, /dev/mapper/VolGroup00-tmp_vol (/tmp),
/dev/mapper/VolGroup00-usr_vol, /dev/mapper/VolGroup00-usr_vol (/usr),
/dev/mapper/VolGroup00-var_vol, /dev/mapper/VolGroup00-var_vol (/var),
/dev/mapper/VolGroup00-var_crash_vol, /dev/mapper/VolGroup00-var_crash_vol (/var/crash),
/dev/mapper/VolGroup00-var_log_audit, /dev/mapper/VolGroup00-var_log_audit (/var/log/audit)]|

          multiViewUrl=http://15.154.72.226:8080/SiteScope/MultiView|
          fullMonitorName=SiteScope\HP\KVM VM CPU Monitor\KVM_TestVM|
          newSiteScopeURL=http://15.154.72.226:8080/SiteScope|sample=5|
          SiteScopeBaseUrl=http://ossvm8.ind.hp.com:8080|
          SiteScopeHost=ossvm8.ind.hp.com|
          SiteScopeURL=http://15.154.72.226:8080/SiteScope|
          SiteScopeuserurl=http://15.154.72.226:8080/SiteScope/userhtml/SiteScope.html|
          state=Virttop Management/Domains
          Information/KVM_TestVM/Performance/%CPU=0.1, , cpu_usage_high=0,
          cpu_usage_medium=1, cpu_usage_low=0|
          tag=|targetHost=sheep.gre.hp.com|
          targetIP=16.16.94.139|
          targetIPVersion=IPV4|
          templateDeployPath=HP/KVM VM CPU Monitor|
          time=11:54 AM 6/23/14|
          warningOnly= cpu_usage_medium: 1|customerId=&amp;lt;customerId&amp;gt;
   </additionalText>
</AlarmCreationInterface>
```

The following is an enriched alarm, after processing by UCA NFVD Problem Detection.

```
Valuepack.
- alarmRaisedTime          = 2014-05-05T20:41:00.848+05:30
```

```
          - sourceIdentifier          = NFVD_Source
          - originatingManagedEntity      = KVM_TestVM
          - originatingManagedEntityStructure
              -> Host = ossvm1.ind.hp.com
          - alarmType              = QUALITY_OF_SERVICE_ALARM
          - probableCause            = UtilizationPercentage
          - perceivedSeverity          = CRITICAL
          - networkState            = NOT_CLEARED
          - operatorState            = NOT_ACKNOWLEDGED
          - problemState            = NOT_HANDLED
          - problemInformation          = Attribute not available
          - specificProblem          = ERROR
          - additionalInformation        = null
          - additionalText            = SiteScope alarm|MONITOR.cpuMonitor-
001|CONDITION=ERROR|group=KVM VM CPU
Monitor|groupdescription=CPU|groupID=201070542|id=1
          - proposedRepairActions        = null
          - notificationIdentifier      = 0
          - correlationNotificationIdentifiers = Attribute not available
          - timeInMilliseconds          = 1399302660848 [2014/05/05 20:41:00.848 +0530]
          - targetValuePack          = null
          - sourceScenarios          = [com.hp.uca.expert.vp.pd.ProblemDetection]
          - passingFilters            = [NfvdScenario]
          - passingFiltersTags          = {NfvdScenario=[Trigger, ProblemAlarm, SubAlarm]}
          - passingFiltersParams        = {NfvdScenario={}}
          - hasParents              = false
          - parentsNumber            = 0
          - parents              = null
          - hasChildren            = false
          - childrenNumber          = 0
          - children              = null
          - justInjected            = false
          - aboutToBeRetracted        = false
          - hasStateChanged          = false
          - stateChanges            = none
          - hasAVCChanged          = false
          - attributeValueChanges        = none
          - customFields
            -> userText = NFVD-PD
            -> NFVTopology =
<Start>
<VIRTUAL_MACHINE>
artifactCategory=GENERIC;
GENERAL.Description=A Virtual machine;
artifactId=KVMVM-2001;
GENERAL.Name=KVM_TestVM;
SERVICE_ACTION=CREATE;
templateId=template-512;
GENERAL.Type=VMtype;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=VIRTUAL_MACHINE;
GENERAL.hostname=KVM_TestVM;
GENERAL.Management_access=http://vm1.ind.com:8080;
</VIRTUAL_MACHINE>
<VNF_COMPONENT>
artifactCategory=GENERIC;
GENERAL.Description=This is VNFC component;
artifactId=VNFC-BLR1;
GENERAL.Name=vnfc-BNGALORE;
SERVICE_ACTION=CREATE;
templateId=template-8001;
lastUpdateTimestamp=15-04-2014 12:58;
creationTimestamp=15-04-2014 12:57;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=VNF_COMPONENT;
</VNF_COMPONENT>
<VIRTUAL_PORT>
artifactCategory=GENERIC;
artifactId=Ethe.1990;
SERVICE_ACTION=CREATE;
templateId=template-512;
lastUpdateTimestamp=15-04-2014 12:58;
SERVICE_TYPE=NFVD;
```

SERVICE_NAME=INSART;
artifactFamily=VIRTUAL_PORT;
INFO.Speed=10;
INFO.Name=EthernetPort-10GB;
INFO.ID=Ethe.1990;
INFO.Type=Port;
INFO.MAC=12:34:56:78:9A:BD;
</VIRTUAL_PORT>
<VIRTUAL_LUN>
artifactCategory=GENERIC;
artifactId=LUN-1001;
SERVICE_ACTION=CREATE;
templateId=template-512;
lastUpdateTimestamp=15-04-2014 12:58;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=VIRTUAL_LUN;
INFO.Name=LUN-1.2-BLR;
INFO.Amount=30;
INFO.ID=LUN-1001;
INFO.Type=Storage;
</VIRTUAL_LUN>
<VIRTUAL_DISK>
artifactCategory=GENERIC;
artifactId=vDisk1;
SERVICE_ACTION=CREATE;
templateId=template-512;
lastUpdateTimestamp=15-04-2014 12:58;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=VIRTUAL_DISK;
INFO.Name=Seagate-500GB-SATA;
INFO.Amount=2;
INFO.ID=PSATA-500;
INFO.Type=Memory;
</VIRTUAL_DISK>
<VIRTUAL_MEMORY>
artifactCategory=GENERIC;
artifactId=RAM-4001;
SERVICE_ACTION=CREATE;
templateId=template-512;
lastUpdateTimestamp=15-04-2014 12:58;
creationTimestamp=15-04-2014 12:57;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=VIRTUAL_MEMORY;
INFO.Name=DDR-RAM-8GB;
INFO.Amount=1;
INFO.ID=RAM-4001;
INFO.Type=Memory;
</VIRTUAL_MEMORY>
<VIRTUAL_CORE>
artifactCategory=GENERIC;
artifactId=VCore-1001;
SERVICE_ACTION=CREATE;
templateId=template-512;
lastUpdateTimestamp=15-04-2014 12:58;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=VIRTUAL_CORE;
INFO.Speed=2233;
INFO.Name=VCore-I5;
INFO.Amount=3;
INFO.ID=Serial-VCore-1001;
INFO.Type=Virtual Core;
</VIRTUAL_CORE>
<MONITOR>
artifactCategory=GENERIC;
GENERAL.Description=Network Monitor;
artifactId=networkMonitor-004;
GENERAL.Name=Network;
SERVICE_ACTION=CREATE;
templateId=template-004;
lastUpdateTimestamp=30-04-2014 12:57;
creationTimestamp=30-04-2014 12:57;

```
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=MONITOR;
GENERAL.Frequency=30;
GENERAL.DeploymentPath=;
</MONITOR>
<MONITOR>
artifactCategory=GENERIC;
GENERAL.Description=Disk Monitor;
artifactId=diskMonitor-003;
GENERAL.Name=Disk;
SERVICE_ACTION=CREATE;
templateId=template-003;
lastUpdateTimestamp=30-04-2014 12:57;
creationTimestamp=30-04-2014 12:57;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=MONITOR;
GENERAL.Frequency=30;
GENERAL.DeploymentPath=HP/NFVD/NS-Routing/VNF-K/VNFC-K;
</MONITOR>
<MONITOR>
artifactCategory=GENERIC;
GENERAL.Description=Memory Monitor;
artifactId=memoryMonitor-002;
GENERAL.Name=Memory;
SERVICE_ACTION=CREATE;
templateId=template-002;
lastUpdateTimestamp=30-04-2014 12:57;
creationTimestamp=30-04-2014 12:57;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=MONITOR;
GENERAL.Frequency=30;
GENERAL.DeploymentPath=HP/NFVD/NS-Routing/VNF-Z/VNFC-Z;
</MONITOR>
<MONITOR>
artifactCategory=GENERIC;
GENERAL.Description=CPU Monitor;
artifactId=cpuMonitor-001;
GENERAL.Name=CPU;
SERVICE_ACTION=CREATE;
templateId=template-001;
lastUpdateTimestamp=30-04-2014 12:57;
creationTimestamp=30-04-2014 12:57;
SERVICE_TYPE=NFVD;
SERVICE_NAME=INSART;
artifactFamily=MONITOR;
GENERAL.Frequency=30;
GENERAL.DeploymentPath=HP/NFVD/NS-Routing/VNF-Y/VNFC-Y;
</MONITOR>
<End>
    -> Evp = UCA_NFVD_PublishToNomBus
    -> Evpversion = 1.0
    -> Evpscenario = publishToNomBus
    -> Problem = NFVD:SCALE_OUT_CPU
    -> Parameternames = ArtifactInstanceId
    -> Parametervalues = KVMVM-2001
    -> Resourceinstance = KVMVM-2001
    -> Resourcetype = INVENTORY
    -> Actionpreset = true

    -> Action = SCALE_OUT
```

14.1.2 Autonomous action

The Autonomous action of NFV Director is based on UCA Automation Action Framework. For information on Automation framework, refer to the *HP UCA Automation - Administrator and User Interface Guide.*

The mandatory alarm attributes required for Autonomous action are provided in the following table.

Alarm Attribute Name	Alarm Attribute Value
Evp	UCA_NFVD_PublishToNomBus
Evpversion	1.0
Evpscenario	publishToNomBus
Problem	NFVD:<Problem Name as per UCA Automation>
Action	SCALE_IN | SCALE_OUT | SCALE_UP | SCALE_DOWN | SCRIPT If the Action parameter is not available, the Autonomous action is not triggered. The value is fetched from the Graph Database, depending on the raw alarm received.
Parameternames	< Value depending on Action Type>
Parametervalues	< Value depending on Action Type>
Resourcetype	INVENTORY
Actionpreset	TRUE

Table 13: Alarm attributes for Autonomous action

14.1.3 Status of Action and Reporting

The status of action and reporting is available in UCA Automation console.

Figure 198: Snapshot of Status of action

Print Preview   Print   ×

ID: 144
Action Name: SCALE_IN
Action ID: 107
Problem: SCALE_IN_CPU
Mode: Closed Loop
Action Originator: alarm
Originator: 201019788
State: Ok
Status: PASSED

Action Request: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <msg xmlns="http://types.ws.ucaautomation.hp.com/"> <header> <ActionRequest Originator="alarm" OpenLoop="false" Mode="real"> <ActionId></ActionId> <Operation>SCALE_IN</Operation> <OriginatorId></OriginatingManagedEntity> <Problem>SCALE_IN_CPU</Problem> <ActionPreset>true</ActionPreset> </ActionRequest> </header> <body> <Parameters> <Parameter> <attribute>ArtifactInstanceId</attribute> <value>KVMVM-2001</value> <resourceTypeID>INVENTORY</resourceTypeID> <resourceInstanceID>KVMVM-2001</resourceInstanceID> </Resource> </body> </msg>

Task Response: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <resp_msg xmlns="http://types.ws.ucaautomation.hp.com/"> <header> <ActionId>107</ActionId> <TaskId>144</TaskId> <ActionInstanceId>265</ActionInstanceId> <OriginatorId>201019788</OriginatorId> <response></Description> </MajorCode> <MinorCode> <Code>0</Code> <Description>VM Basic service call response</Description> </MinorCode> <TaskStatus>PASSED</TaskStatus> <Diagnostics><?xml version="1.0" encoding="UTF-8" standalone="yes"?><S xmlns:ns2="http://www.hp.com/soa/protocoladapter/ngws"><ns2:ssid>0.8783977740755039</ns2:ssid><ns2:userId>foo</ns2:userId><ns2:sosaCode>0</ns2:sosaCode><ns2:sosaDescription>VM Basic service call response</ response</ns2:description><ns2:creationTime>12:39:10.238+05:30</ns2:creationTime><ns2:startTime>12:39:10.238+05:30</ns2:startTime><ns2:finishTime>12:39:10.238+05:30</ns2:finishTime><ns2:creationDate>2014-06-19 19+05:30</ns2:finishDate><ns2:outputParams><ns2:param><ns2:name>ArtifactInstanceId</ns2:name><ns2:value>XXXX_12345678_YYY</ns2:value></ns2:param><ns2:param><ns2:name>Category.atribute1</ns2:n </Diagnostics> <Parameters> <Parameter> <attribute>outputparameters</attribute> <value>ArtifactInstanceId,XXXX_12345678_YYY,String;Category.atribute1,1024,String</value> </Parameters> </TaskResponse> </body> </resp_msg>

Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?><ServiceExecutionReport xmlns:ns2="http://www.hp.com/soa/protocoladapter/ngws"><ns2:ssid>0.8783977740755039</ns2:ssid><ns2:userId>foo</ns2:userId><ns2:sosaCode>0</ns2:sosaCode><ns2:r Basic service call response</ns2:description><ns2:creationTime>12:39:10.238+05:30</ns2:creationTime><ns2:startTime>12:39:10.238+05:30</ns2:startTime><ns2:finishTime>12:39:10.238+05:30</ns2:finishTime><ns2:creationDate>2014-06-19</ns2:creation <ns2:outputParams><ns2:param><ns2:name>ArtifactInstanceId</ns2:name><ns2:value>XXXX_12345678_YYY</ns2:value></ns2:param><ns2:param><ns2:name>Category.atribute1</ns2:name><ns2:value>1024</ns2:value></ns2:param></ns2:outputParams></Se

Start Time: 19-Jun-14 12:39:09 PM
End Time: 19-Jun-14 12:39:10 PM

Figure 199: Snapshot of Reporting of action

Troubleshooting NFV-D

This section describes how to identify and resolve problems that might occur when using NFV Director.

This chapter covers the following sections:

- Troubleshooting installation and configuration

-

Symptom	Possible cause	Possible Solution
During installation, an error message appears because the port is not configured.	Port is not configured properly.	Check if the related port is configured in `/opt/HP/nfvd/bin/nfvd_agw_env.sh`.
During installation, an error message appears because the path is not configured.	Path is not configured properly.	Check if the related path is configured in `/opt/HP/nfvd/bin/nfvd_agw_env.sh`.
Error message appears when installing RPM.	Dependent RPM is not installed.	Make sure that the sequence of RPM installation is followed as mentioned in the *Installation Guide*.
Uninstallation fails.	Uninstallation sequence is not followed.	Each component must be uninstalled in the given sequential manner as mentioned in the *Uninstallation* section of the *Installation Guide*.

Symptom	Possible cause	Possible Solution

Logging in to the NFVD fulfillment Service activator fails.	License might not be available or updated.	1. Check if the `/opt/OV/ServiceActivator/license.txt` file exists.
		
		a. If the `license.txt` file is not present, get the license from the HP site.
		
		b. If present, check the license status by running the following command:
		
		`./opt/OV/ServiceActivator/bin/checkLicense`
		
		c. If the license is outdated, update the license by running the following command:
		
		`./opt/OV/ServiceActivator/bin/updateLicense`
		
		2. Run the following commands to restart HPSA.
		
		`/etc/init.d/activator stop`
		
		`/etc/init.d/activator start`
		
		If the issue persists, refer to the *HPSA installation guide*.

Table 14: Troubleshooting installation and configurations

- Troubleshooting TopologyTroubleshooting Topology

    Table 15: Troubleshooting Topology

- Troubleshooting monitor deploymentsTroubleshooting monitor deployments

    Table 16: Troubleshooting Monitor Deployments

- Troubleshooting alarmsTroubleshooting alarms

    Table 17: Troubleshooting Alarms

- Troubleshooting synchronized NFVD Assurance and FulfillmentTroubleshooting synchronized NFVD Assurance and Fulfillment

    Table 18: Troubleshooting synchronized NFVD Assurance and Fulfillment

15.1 Troubleshooting installation and configuration

This section describes the possible problems and solutions faced during installing, uninstalling, and configuring the NFV Director.

15.1.1 Best practices

Make sure the system meets the desired hardware requirements, as available in the *Installation Guide*.

Make sure that you install each NFVD Assurance component in a given sequence. For example,

- nfvd-assur-gw-tpp-1.0-373.el6.noarch.rpm

- nfvd-assur-gw-core-1.0-373.el6.noarch.rpm

- nfvd-assur-gw-base-1.0-373.el6.noarch.rpm

- nfvd-correlation-1.0-373.el6.noarch.rpm

- nfvd-monitors-1.0-373.el6.noarch.rpm

Make sure that all property files are configured correctly and are present under the `/var/opt/HP/nfvd/conf/` directory.

- `fulfillment.properties`
- `monitoring.properties`
- `NFV_Assurance.properties`
- `topology.properties`

Stop the SiteScope if it is running and then install the NFVD Assurance RPMs.

15.1.2 Troubleshooting cases

Symptom	Possible cause	Possible Solution
During installation, an error message appears because the port is not configured.	Port is not configured properly.	Check if the related port is configured in `/opt/HP/nfvd/bin/nfvd_agw_env.sh`.
During installation, an error message appears because the path is not configured.	Path is not configured properly.	Check if the related path is configured in `/opt/HP/nfvd/bin/nfvd_agw_env.sh`.
Error message appears when installing RPM.	Dependent RPM is not installed.	Make sure that the sequence of RPM installation is followed as mentioned in the *Installation Guide*.
Uninstallation fails.	Uninstallation sequence is not followed.	Each component must be uninstalled in the given sequential manner as mentioned in the *Uninstallation* section of the *Installation Guide*.

Symptom	Possible cause	Possible Solution

Logging in to the NFVD fulfillment Service activator fails.	License might not be available or updated.	1. Check if the `/opt/OV/ServiceActivator/license.txt` file exists.
		
		a. If the `license.txt` file is not present, get the license from the HP site.
		
		b. If present, check the license status by running the following command:
		
		`./opt/OV/ServiceActivator/bin/checkLicense`
		
		c. If the license is outdated, update the license by running the following command:
		
		`./opt/OV/ServiceActivator/bin/updateLicense`
		
		2. Run the following commands to restart HPSA.
		
		`/etc/init.d/activator stop`
		
		`/etc/init.d/activator start`
		
		If the issue persists, refer to the *HPSA installation guide*.

Table 14: Troubleshooting installation and configurations

15.2 Troubleshooting Topology

This section describes possible problems that occur when creating the topology.

15.2.1 Best practices

Make sure that the following properties are set with correct values in `topology.properties` file.

- neo4j.protocol
- neo4j.host
- neo4j.port
- neo4j.db
- neo4j.data

Make sure that the HP UCA EBC component is up and running.

15.2.2 Troubleshooting cases

Symptom	Possible cause	Possible Solution
Error appears when creating a component.	Assurance gateway is down.	Check if the Assurance Gateway is up and running. Run the `/opt/HP/nfvd/bin/nfv-director.sh status` command. You should get the following output `HP Assurance Gateway application server is running`.
	Could not create topology.	Check whether the database is configured properly in the `topology.property` file. Check for errors in the `server.log`.

Symptom	Possible cause	Possible Solution
Cannot create relationship.	Parent component might not be available.	Check whether the child component already exists in the topology DB.
		If the problem persists and if you see a mismatch of data between fulfillment and topology, manually perform re-sync topology operation as mentioned in the *Synchronize NFVD Assurance and Fulfillment* section of the *Installation Guide*.
Delete component fails.	Component may not exist at topology.	Check if the desired component exists in topology.
Connection refused.	HP UCA-EBC configuration is missing.	Check if the database-related information is properly configured in the `/var/opt/HP/nfvd/conf/topology.properties` file. For more details, see the *HP NFV Director Installation Guide*.

Table 15: Troubleshooting Topology

15.3 Troubleshooting monitor deployments

This section discusses the possible causes and solutions for errors that occur when deploying and undeploying various types of monitors.

15.3.1 Best practices

Make sure that the following parameters are set with correct values:

- SiteScope.login
- SiteScope.password
- SiteScope.host
- SiteScope.port
- SiteScope.useSSL

Make sure that the SiteScope component is up and running.

Make sure that all KPIs are defined properly.

For custom monitors, make sure that the actual template path is available in the SiteScope server.

15.3.2 Troubleshooting cases

Symptom	Possible cause	Possible solution
Cannot deploy monitor.	SiteScope is not running.	Check if SiteScope is active by running the following command: `/opt/HP/nfvd/bin/nfv-director.sh status`. You should see the following message: `SiteScope is running`.

Symptom	Possible cause	Possible solution
	Assurance gateway is down.	Check the status of Assurance Gateway by running the following script: `/opt/HP/nfvd/bin/nfv-director.sh status` The output must contain `HP Assurance Gateway application server is running` along with the other components as mentioned in the *Installation Guide*.
	Incorrect sequence of ACTION.	During the deployment of a monitor follow this sequence: a. The component must be present at infrastructure. b. Deploy monitor action must be sent. c. Start monitor action must be sent.
	Incorrect SiteScope configuration.	Check if the SiteScope details like host, port, and user details are configured correctly in the `/var/opt/HP/nfvd/conf/monitoring.properties` file.
		Check if the configured host is accessible via deployed server.
Monitor deployment failure due to certificate error.	Certificate is not configured properly.	1. Log in to SiteScope. 2. Select **Preferences context** > **Certificate Management**. 3. To add certificates, click the **Import Certificates** button. The **Import Certificates** dialog box opens. 4. Select **File** or **Host** and enter the details of the source server. 5. From the **Loaded Certificates** table, select the server certificates to import and click **Import**. The imported certificates are listed on the **Certificate Management** page. 6. To view certificate details, double-click a certificate. To view the **Certificate Management** page, you must be an administrator in SiteScope or a user granted with View certificates list permissions.
Cannot deploy monitor via vCenter	Cannot fetch real-time counter from respective VM via vCenter server.	Make sure that the real-time counters are available on respective VM.
Monitor deployment fails displaying the `RemoteException` message.	SiteScope is not reachable.	Make sure that the SiteScope server is reachable for the NFVD server.

Symptom	Possible cause	Possible solution
Monitor deployment fails displaying the `No actual counter` error message.	Does not conform to proper KPI naming convention.	Refer to Figure 186: KPIs and counters supported matrix for various hypervisors.

Table 16: Troubleshooting Monitor Deployments

15.4 Troubleshooting alarms

15.4.1 Best practices

Make sure that the following components are up and running.

- All the components of UCA Automation v 1.1

- Generic SNMP CA

- OM HP SiteScope Customization for Generic SNMP CA

- Assurance Gateway v 1.0

- SiteScope v 11.23

Make sure UCA-EBC host and port are correctly configured via SiteScope preference setting.

15.4.2 Troubleshooting cases

Symptom	Possible cause	Possible Solution
Generating alarms fails.	Correlation engine is down.	Check if all the required components are up and running. Run the `/opt/HP/nfvd/bin/nfv-director.sh status` command. For a list of components, refer to the *HP NFVD Install Guide*.
UCA-EBC is not generating alarms.	UCA-EBC details are not configured properly in the SiteScope.	Use the following procedure: 1. Check the reports in SiteScope. 2. Go to the **Preferences** in the SiteSope and check whether the UCA-EBC host and port are configured correctly. 3. Check if the same port is configured in the `/var/opt/openmediation-V62/containers/instance-0/ips/generic-snmp-ca-V10/etc/config.properties` file. 4. Enable the `collector.log` by setting the `collector.logger.enabled=true` in the `/var/opt/UCA-EBC/instances/default/conf/uca-ebc.properties` file. 5. Check the respective alarm information in the logs at `/var/opt/UCA-EBC/instances/default/logs/uca-ebc-collector.log` file.

Symptom	Possible cause	Possible Solution
		Restart the UCA-EBC. 1. Log in as `su -uca`. 2. Stop UCA-EBC by running the `/opt/UCA-EBC/bin/uca-ebc stop` command. 3. Start UCA-EBC by running the `/opt/UCA-EBC/bin/uca-ebc start` command.
Fails to take auto action on alarms.	Action registry is missing.	Check if the Action registry is properly configured in the `/var/opt/UCA-EBC/instances/default/conf/ActionRegistry.xml` file.
		Check if all value packs are up and running. For more details, refer to the *HP NFVD Install Guide*.
		1. Enable the logs for Ebc-ebc value packs. 2. Check if alarms related to topologies are present in the Neo4J DB. For more details, see section 15.8 Browsing Neo4J DB for Topology.
UCA-ACB is not processing alarms.	SNMP OID prefix-flag in the SiteScope.	In the SiteScope, the **Add System OID as a prefix to SNMP Trap** flag must be disabled using the following menu options: **Preferences** > **SNMP Preferences** > **Send SNMP Trap Preference** > **Advance Settings** > **SNMP Object**.

Table 17: Troubleshooting Alarms

15.5 Troubleshooting synchronized NFVD Assurance and Fulfillment

15.5.1 Best practices

Make sure that all parameters in the `/var/opt/HP/nfvd/conf/fulfillment.properties` file are set properly.

15.5.2 Troubleshooting cases

Symptom	Possible cause	Possible solution
Re-sync does not start when starting NFVD.	Property is not set.	If re-synchronization is desired when starting NFVD, make sure that the **RESYNC_AT_STARTUP** parameter is set to `true`/yes in the `/var/opt/HP/nfvd/conf/fulfillment.properties` file.
Re-sync fails.	Topology DB property is not set.	Make sure that you point to the correct and valid database in the `/var/opt/HP/nfvd/conf/topology.properties` file.

Symptom	Possible cause	Possible solution
	Fulfillment details are missing.	Make sure that the fulfillment details are properly set in the `/var/opt/HP/nfvd/conf/fulfillment.properties` file.
	Fulfillment URL is incorrect.	Make sure that the fulfillment URL configured in the `/var/opt/HP/nfvd/conf/fulfillment.properties` file is valid. For an example of the `fulfillment.properties` file, refer to the *HP NFVD Install Guide*.
Re-sync completes successfully, but the topology does not change.	Data is already up-to-date.	An issue might occur when data between NFVD assurance and fulfillment are already synchronized.

Table 18: Troubleshooting synchronized NFVD Assurance and Fulfillment

15.6 Troubleshooting deletion process

15.6.1 Best practices

Make sure that the VIM and all the compute nodes are working and properly monitored from SW perspective and HW perspective.

15.6.2 Troubleshooting cases

Symptom	Possible cause	Possible solution
VMs are not deleted from DB and monitors are started but the compute nodes are in error.	Nova server is down or in error.	If it tries to delete a VM and nova fails, NFV director will stop and will not delete more VMs and the first VM that fails will not be deleted from inventory and monitors will not be started. Monitors will probably continue to inform that something is happening with the VM (as the whole compute node is failing), this may trigger extra actions like scale in/out depending on the monitor configuration.

Table 19: Troubleshooting Delete VMs

15.7 Troubleshooting with logs

Various components have their respective log places under respective component directories. If problems are not addressed using troubleshooting cases sections, the user can collect and provide the respective logs for further debugging.

15.7.1 NFVD Assurance

Logs related to monitoring, components, and topology are available here.

1. Enable logs using the following command:

```
$JBOSS_HOME/standalone/configuration/logging.properties
```

2. Change the value of the `logger.level`.

   The possible values are the following:

   - FINE
   - WARN
   - INFO
   - DEBUG
   - SEVER
   - ERROR

Logs are available at `$JBOSS_HOME/standalone/log/server.log`.

15.7.1.1 Installation or Uninstallation

Logs for all HP NFVD installation and uninstallation operations are available at the following locations:

- `/tmp/agw_postinstall_base.txt`
- `/tmp/agw_postinstall_monitors.txt`
- `/tmp/agw_postinstall.txt`
- `/tmp/agw_postUninstall.txt`
- `/tmp/agw_preinstall_base.txt`
- `/tmp/agw_preinstall_monitors.txt`
- `/tmp/agw_preinstall.txt`

15.7.2 SiteScopes

SiteScope related logs, such as the logs for monitoring deployment, KPI, and so on are available in the locations mentioned in this section.

Logs are available at the following locations:

- `/opt/HP/nfvd/tpp/jboss/standalone/configuration`
- `/opt/HP/SiteScope/logs`

15.7.3 UCA-EBC logs

All alarm-related logs are available in the UCA-EBC logs.

To enable/disable logs, set the `collector.logger.enabled=true` in the `/var/opt/UCA-EBC/instances/default/conf/uca-ebc.properties`.

Logs are available at `/var/opt/UCA-EBC/instances/default/logs/uca-ebc-collector.log`.

15.7.4 NFVD Fulfillment

NFV Director Fulfillment logs are distributed in the following directories:

- `/opt/HP/jboss/standalone/log`
- `/var/opt/OV/ServiceActivator/log/<hostname>`
- `/opt/OV/ServiceActivator/EP/SOSA/log`

- `/opt/OV/ServiceActivator/EP/LockManager/log`
- `/opt/OV/ServiceActivator/EP/ECP/log`

15.8 Browsing Neo4J DB for Topology

To check whether the alarm related topologies are present in the DB,

1. Note the OME name of the alarms from the `/var/opt/UCA-EBC/instances/default/logs/uca-ebc-collector.log` file.

2. Access Neo4J using a browser.

3. Run the following query.

```
start n=node(*) where has(n.`GENERAL.Name`) and
n.`GENERAL.Name`= "<OME Name>" return n
```

15.9 Contacting customer support

If problems persist even after going through all troubleshooting cases, collect the logs as mentioned in the troubleshooting sections and contact the customer care.

Cloud System 8 Operations

A.1 Login

The user can enter the login credentials based on the permissions the user has, which ranges from a super-admin to tenant (project) user.
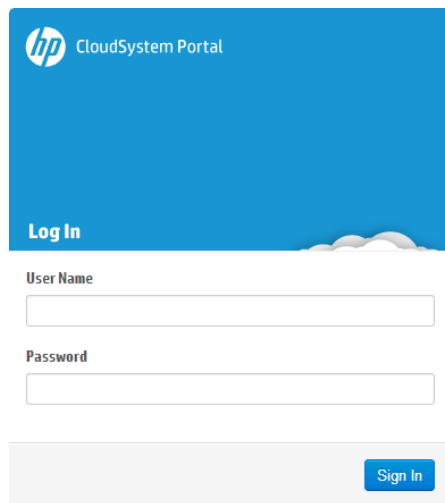

Figure 200: CloudSystem Login Portal

A.2 Overview

The following image shows the global status of the server, which includes allocated resources, active instances (usage summary), and the capacity of resources for an active tenant/user (limit summary).
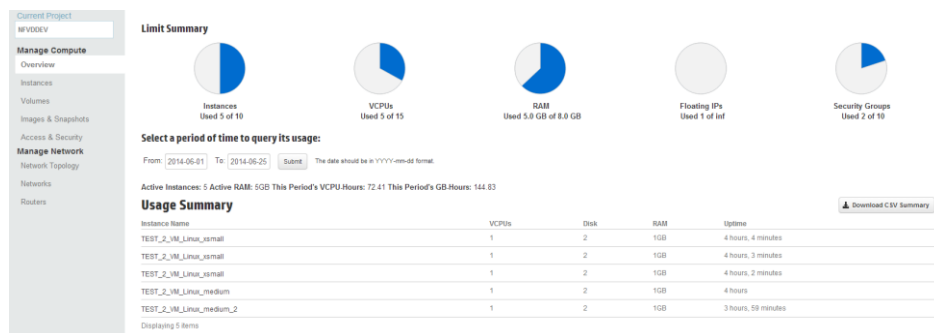

Figure 201: CloudSystem overview

A.3 Virtual machines (Instances)

This section provides information on the instances created on a server.

Name—Name of the virtual machine.
Image Name—Name of the image that was used to create the instance.
IP Address—IP addresses of the virtual machines (external and internal).
Size—Technical characteristics, flavor of the virtual machine (RAM, vCPU, disk).
Status—Status of the VM (Active, Error, shut-down, and so on).

Figure 202: CloudSystem VM Instances

You can run the following VM operations from this section:

- Create a Virtual Machine

- Update a Virtual Machine

- Query for a Virtual Machine

- Delete (Terminate) a Virtual Machine

- Start / Stop a Virtual Machine

A.4 Images

This section provides information on the disk images allocated on a server.

Figure 203: CloudSystem Images list

A.5 Networks

This section provides a list of available networks and subnets.

Figure 204: CloudSystem List of available networks and subnets

You can run the following operations from this screen:

- Create Network

- Create a Subnet from a Network

- Edit Network

- Edit Subnet

- Delete Network

Figure 205: CloudSystem Subnet

Automation workflows

B.1 WF_NFVD_CREATE_INSTANCES_FROM_TEMPLATE

This workflow creates a complete tree of artifact instances from an artifact template tree. This workflow has three child workflow policies:

- WF_NFVD_INSTANCE_ASSIGNMENT
- WF_NFVD_INSTANCE_VALIDATION
- WF_NFVD_CREATE_POLICY_INSTANCES

Three special nodes called policy nodes are available, as the artifact template tree can include three types of policy nodes, each one with its specific functionality:

- POLICY:ENTITY_ASSIGN
- POLICY:VALUE_VALIDATION
- POLICY:ENTIY_RANGE

The first child workflow sets the instance attributes of an artifact indicated by the ENTITY_ASSIGN node and the second child workflow validates those attributes' content. If the validation is incorrect, the instances cannot be created.

B.1.1 Using the Workflow

To launch the workflow, use the following procedure:

1. Right-click the desired template.

   That template defines the parent of the instances tree that is created.

2. From the pop-up menu, select the **Create Instances from Template**.

   On the right-hand side, a form appears.

3. Enter the Parent Instance ID and the Parent relationship type in the text boxes.

4. Click **OK**.

After some time, the instance tree is created according to the policies.

B.1.2 Results

B.1.2.1 ERRORS

The following list of errors might appear.

- 1001: Parent Artifact with instanceId %INPUT_PARENTARTIFACTID% does not exist in the system.
- 1002: Template ID is a mandatory parameter.
- 1003: TemplateID %INPUT_TEMPLATEID% does not exist.

- 1004: Error Assign Workflow does not run ok.

- 1005: Error Validation Workflow does not run ok.

- 1006: The instance creation was not possible. The validation failed.

- 1007: Error storing relationship. Parent = %INPUT_PARENTARTIFACTID%, child = %VAR_ARTIFACT.Id%, type = %VAR_RELATION.Type%

- 1008: Unexpected error executing child workflow.

- 1009:

- 1010: Unexpected error executing child workflow.

B.1.2.2 Successful Ends

0: Workflow ends ok

B.2 WF_NFVD_CREATE_POLICY_INSTANCES

B.2.1 General Description

This workflow is a child workflow of WF_NFVD_CREATE_INSTANCES_FROM_TEMPLATE. The workflow creates the instances of the policy templates in a Create Instance from Template operation.

B.2.2 Using the Workflow

This workflow is automatically launched when the WF_NFVD_CREATE_INSTANCES_FROM_TEMPLATE workflow is running.

B.2.3 Results

B.2.3.1 Errors

2001: Unexpected error executing child workflow

B.2.3.2 Successful ends

0: Workflow ends ok

B.3 WF_NFVD_INSTANCE_VALIDATION

B.3.1 General Description

Workflow launched inside of the Create Instance from Template execution. It is used to set the attributes of the instances to create according to the policy (POLICY:ENTITY_ASSIGN).

B.3.2 Using Workflow

This workflow is automatically launched when the WF_NFVD_CREATE_INSTANCES_FROM_TEMPLATE workflow is running, only if assigned policies exist.

B.3.3 Results

B.3.3.1 Errors

- 13001: Mandatory input parameter INPUT_TEMPLATEARTIFACTID is not present
- 13002: Mandatory input parameter INPUT_INSTANCEARTIFACTID is not present
- 13003: ERROR Artifact Template with templateId = %INPUT_TEMPLATEARTIFACTID% does not exist in the system
- 13004: Mandatory policy attribute TYPE is not present
- 13005: ERROR Assign workflow do not run OK
- 13006: ERROR Assign Script do not run ok
- 13007: ERROR Java Assignment was not be ok

B.3.3.2 Successful ends

0: Workflow ends ok

B.4 WF_NFVD_DELETE_INSTANCE

B.4.1 General Description

This workflow deletes a complete instance tree from the parent to the children.

B.4.2 Using Workflow

To launch it, use the following procedure:

1. Right-click the instance that is the parent of the tree and which should be deleted.
2. Select the **Delete Artifact Instance Tree** from the pop-up menu.
3. Click **OK**.

B.4.3 Results

B.4.3.1 Errors

- 3001: instanceID (mandatory) not present
- 3003: Unexpected error executing child workflow WF_NFVD_DELETE_INSTANCE
- 3004: child workflow error - %wf_ret_error_description%`
- 3005: error deleting instance %VAR_ARTIFACT.Id%
- 3006: ERROR WF_NFVD_ASSURANCE_MONITOR was not run ok
- 3007: ERROR WF_NFVD_DEACTIVATE_VM_CS8 was not run ok%

B.4.3.2 Successful Ends

0: Workflow ends ok

B.4.3.3 Warnings

0: artifact with instanceID %INPUT_INSTANCEID% not found

B.5 WF_NFVD_SCALE_OUT

In a scenario, where you have 1 Virtual Machine 1 child element (for example, 1 Virtual Core), by launching the Scale-out workflow, you can increase the number of Virtual Cores in an amount indicated by the Range Policy.

B.5.1 Using the Workflow

1. Right-click the instance that you want to increase the amount.
2. Select the **Scale In** option in the pop-up menu.
3. Click **OK**.

B.5.2 Results

B.5.2.1 Errors

- 4001: ArtifactInstanceId is a mandatory input parameter
- 4002: Artifact Instance with instanceId = %INPUT_INSTANCEARTIFACTID% does not exist in the system
- 4003: Scale out operation is only supported for instances created from template. InstanceId = %INPUT_INSTANCEARTIFACTID%
- 4004: Recursive call failed
- 4005: Malformed Template: The number of children of tempalteId = %VAR_TEMPLATE_PR.Id% is not 1
- 4006: Malformed template: parent (%VAR_ARTIFACT_TEMPLATEID%) of the policy (%VAR_TEMPLATE_PR%), must be parent of the OVER Child (%VAR_TEMPLATE_PR_OVER_CHILD%) too
- 4007: ERROR Create Instance From Template WF
- 4008: ERROR Create New Child Relationship

B.5.2.2 Successful Ends

- 0: Workflow ends ok
- 0: OK. SCALE OUT Operation was not possible to do in all artifacts.

B.6 WF_NFVD_SCALE_IN

This workflow is the opposite of the Scale Out operation. In the previous example, to decrease the number of Virtual Cores in an amount indicated by the POLICY:ENTITY_RANGE, launch the workflow over the Virtual Core.

For more information about Policy Range, refer to the *8.3 Policies* section.

B.6.1 Using the Workflow

1. Right-click the instance to be decreased.
2. Select the **Scale In** option.
3. Click **OK**.

B.6.2 Results

B.6.2.1 Errors

- 5001: Mandatory input parameter ArtifactInstanceId is nor present

- 5002: Artifact Instance with instanceId = %INPUT_INSTANCEARTIFACTID% does not exist in the system

- 5003: ERROR Delete Instance From Template WF

- 5004: Recursive call failed

B.6.2.2 Successful Ends

- 0: Workflow ends ok

- 0: Fin flujo OK. SCALE IN Operation was not possible to do in all artifacts.

B.7 WF_NFVD_SCALE_UPDOWN

Talking about of the Virtual Core of the example, suppose that its speed is 2 GHz (this amount is stored in the attribute Amount inside of the INFO Category).

Launching the Scale Up/Down operation it is possible to increase (SCALE_UP) or decrease (SCALE_DOWN) the Virtual Core speed.

B.7.1 Using the Workflow

To launch this operation, right-click on the instance to do the scale and select Scale Up/Down in the pop-up menu. When the form appears on the right-hand side, the empty fields must be filled.

First, choose your operation, typing SCALE_UP or SCALE_DOWN into the field named Scale Operation.

The function of the last field gives the possibility to do the operation over all the tree starting by the instance where you apply the scale. Type TRUE if you want this, otherwise type FALSE.

Finally click **OK** to start the operation.

B.7.2 Results

B.7.2.1 Errors

9001: Mandatory input parameter INPUT_INSTANCEARTIFACTID is nor present.