VNF SDK Project Home Page

Project Detail

Project Description	The "VNF Supplier APIs and SDK" (short name: VNF SDK) project delivers functionality to enable a rich OPEN-O community ecosystem of VNF product suppliers and operators, with high degree of interoperability.
Project PTL	Lizongbiao, Huawei
Participants	Cannonical, China Mobile, China Telecom, GigaSpaces, Huawei, Intel, ZTE
Personnel	Resources

Project Proposal and Planning

Project Proposal	VNF SDK Project Proposal
Project Planning	VNF SDK Mercury Release Planning (M1) VNF SDK Mercury Release Planning Delverables (M1) VNF SDK Mercury Release Functionality Freeze Deliverables (M2)
Testing	VNF SDK Testing
Documentation	VNF SDK Documentation
Legal	

Sub-Projects

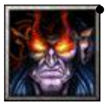
Sub-Project	Description	Owner	Details

Design & Package Tool		Huabing Zhao & Krzysztof Frukacz	
Validate & Lifecycle Test	1. Develop Validate component to help validate VNF Package. 2. Develop Lifecycle Test component to help test VNF lifecycle, e.g. instantiation, termination, etc.	Yuan Liu & Anbing Zhang	**Goto Validate & Lifecycle Test Page**
Function Test	1. Develop Function Test Framework to help test VNF functionality. 2. Develop guidelines to help VNF vendors to plugin their VNF to the framework.	Murali Mohan Murthy	**Goto Function Test Page**
Market Place		Alex & Sukesh	

Meeting Schedules

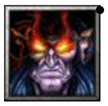
Logistics and Meeting Summary

Recently Updated



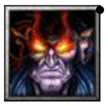
- **gao weitao**
  - Validate & Lifecycle Test updated Apr 28, 2017 • view change
  - VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC1) created Apr 13, 2017
  - VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC2) updated Apr 13, 2017 • view change
  - VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC0) updated Apr 13, 2017 • view change

- **Gildas Lanilis**
  - VNF SDK Project Home Page updated Mar 17, 2017 • view change



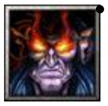
- **gao weitao**
  - VNF SDK Mercury Release Code Freeze Milestone Deliverables (M4) updated Mar 07, 2017 • view change

- **Brian Hedstrom**
  - Mercury Release updated Mar 03, 2017 • view change
  - Project Planning commented Mar 03, 2017
  - VNF SDK References updated Mar 03, 2017 • view change
  - VNF SDK Testing updated Mar 03, 2017 • view change



- **gao weitao**
  - VNF SDK Flow Diagram updated Mar 03, 2017 • view change

- **Brian Hedstrom**
  - VNF SDK Use Case Definitions updated Mar 02, 2017 • view change
  - VNF SDK Use Case Diagram.png attached Mar 02, 2017



- **gao weitao**
  - Relationship between VNF-SDK and other project.png attached Mar 02, 2017

- **Murali Mohan Murthy Potham**
  - VNF SDK Validation / Lifecycle Test CSIT use cases updated Mar 02, 2017 • view change

Project Planning

OPEN-O VNF SDK Design(11.15)

OPEN-O_VNF_SDK_Design(11.15) (1).pptx

Mercury Plan 2016-12-14

OPEN-O VNF-SDK Planning in Mercury Release(2106.12.15).pptx

Mercury Release

- VNF SDK Mercury Release Architecture Milestone Deliverables (M3)
- VNF SDK Mercury Release Code Freeze Milestone Deliverables (M4)
- VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC0)
- VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC1)
- VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC2)
- VNF SDK Mercury Release Functionality Freeze Deliverables (M2)
- VNF SDK Mercury Release Planning (M1)
- VNF SDK Mercury Release Planning Delverables (M1)

VNF SDK Mercury Release Architecture Milestone Deliverables (M3)

Practice Area	Checkpoint	Yes/No	Evidences	How to?
Architecture	Has the Project team reviewed the APIs with the Architecture Committee (ARC)?	Yes		Architecture walkthrough to understand how each project contributes on Release Use Case. ARC to organize the walkthrough.
	Is there a plan to address the findings the API review?	Yes	There have been no findings for API	The plan could be as simple as a Jira issue to track the implementation of findings or a documented plan within the wiki.
	Does the team clearly understand that no changes in the API definition is allowed without formal TSC review and approval?	Yes	NA	In the case some changes are necessary, bring the request to the TSC for review and approval.
	Is there any changes in the scope, functionalities, deliverable, resources, API, repositories since M1 milestone?	No		

Release Management	Are committed Sprint Backlog Stories been marked as "Done" in Jira board?	Yes	Provide Link to Project backlog	
	Are all tasks associated with Sprint Backlog Stories been marked as "Done" in Jira?	Yes		
	Have all findings from previous milestones been addressed?	Yes	There are no findings from previous milestones.	
Development	Has the project team reach the Automated Unit Test Code Coverage expectation? (Refer to artifacts available in Sonar)	Yes	vnf-sdk-function test code Coverage 60.6% vnf-sdk-validate-lc-test Code Coverage 100 % vnf-sdk-marketplace Code Coverage 63% vnf-sdk-design-pkg Code Coverage 71%	Guidance on Code Coverage and Static Code Analysis Tools: Sonar
	Is there any pending commit request older than 24 hours in Gerrit?	No		
	Are all the Jenkins jobs successfully passed (Merge-Jobs)?	Yes	VNFSDK Jenkins	
	Are all binaries available in Nexus?	Yes	VNF-SDK Nexus	
Integration and Testing	Have 50 % of Continuous System Integration Testing (CSIT) Use Cases been implemented successfully in Jenkins?	Yes		https://jenkins.open-o.org/view/vnf-sdk/
	Have you reviewed all the CSIT Test Case with Integration Team?	Yes	Confirmed with Kang Xi	Schedule a meeting with Integration Team to review the deliverable.
	Has the project code successfully passed the Daily Build process?	Yes	VNFSDK Jenkins	Goal is to ensure the latest project commit has not broken the Integration Daily Build

VNF SDK Mercury Release Code Freeze Milestone Deliverables (M4)

The following items are expected to be completed for the project to Pass the M4 Code Freeze Milestone.

M4 Release Code Freeze Milestone overview is available in wiki.

Usage

1. Copy and Paste this template in the wiki Space along with your Release document
2. Fill out the Yes/No column
3. Provide link to evidence (when necessary)

Practice Area	Checkpoint	Yes/No	Evidences	How to?
Product Management	All all JIRA Stories supporting the release use case been implemented?	Yes		
	List the Stories that will not be implemented in this current Release.	Yes	Venus Story List	Setup JIRA fixVersion=Venus
	Are committed Sprint Backlog Stories been coded and marked as "Done" in Jira?	In Progress	will done by 3/8	
	Are all tasks associated with committed Sprint Backlog Stories been marked as "Done" in Jira?	Yes	Task List	
Release Management	Have all issues pertaining to FOSS been addressed?	Yes		
	Have all findings from previous milestones been addressed?		Provide link to JIRA findings	
Development	Are all Defects of priority Highest and High in status "Done" in Jira?	Yes	VNF-SDK Highest and high Bug	
	Have all Google Coding Style issues been addressed?	Yes	OPEN-O Java code style	
	Is there any pending commit request older than 24 hours in Gerrit?	No		

	Are all the Jenkins jobs successfully passed (verify + merge jobs)?	Yes	Provide link to "Merge job" as evidence in Jenkins project tab	
	Are all snapshot binaries available in Nexus?	Yes	VNF-SDK Nexus	
Integration and Testing	Have 100% of Continuous System Integration Testing (CSIT) Use Cases been implemented successfully in Jenkins?	Yes	CSIT Status	
	Is there a Docker images available for each repositories?	In Progress	Will Done by 3.8	
	Has the project code successfully passed the Daily Build process?	Yes		Goal is to ensure the latestproject commit has not broken the Integration Daily Build

VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC0)

Practice Area	Checkpoint	Yes/No	Evidences	How to?
Product Management	Are all tasks associated with the release been marked as "Done" in Jira?	Yes	VNF-SDK JIRA Task	
Release Management	As the project team identified the reduced set of Committers who will proceed with Code Merge after RC0?	Yes	Victor Gao	
	Have all findings from previous milestones been addressed?	Yes	VNF-SDK JIRA Issues	
Development	Have all Defects of priority Highest and High been in status "Done" in Jira?	Yes	VNF-SDK JIRA Issues	
	Is there any pending commit request older than 24 hours in Gerrit ?	No		
	Are all the Jenkins jobs successfully passed (verify + merge jobs)?	Yes	Verify Jobs Merge Jobs	
Integration and Testing	Have all CSIT Use Cases (created by each project team) passed ?	Yes	CSIT Jobs	
	Have all end-to-end CSIT (created by Integration Team) been identified?	Yes	Link to evidence	
	Is there a Docker images available for each repository?	Yes	VNF-SDK Docker	
	Has the project code successfully passed the Daily Build process?	Yes	VNF-SDK Daily Build	Goal is to ensure the latest project commit hasnot broken the Integration Daily Build
Documentation	Is the team ready to contribute in the following 3 documentations: 1. Mercury Release Notes 2. Mercury Tarball Installation 3. Mercury Docker Installation	Yes		Each service in VNF-SDK already have docker, so we dont need Tarball installation

VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC1)

Practice Area	Checkpoint	Yes/No	Evidences	How to?
Product Management	Are all tasks associated with the release been marked as "Done" in Jira?	Yes	VNF-SDK JIRA Task	
Release Management	As the project team identified the reduced set of Committers who will proceed with Code Merge after RC0?	Yes	Victor Gao	
	Have all findings from previous milestones been addressed?	Yes	VNF-SDK JIRA Issues	

Development	Have all Defects of priority Highest and High been in status "Done" in Jira?	Yes	VNF-SDK JIRA Issues	
	Is there any pending commit request older than 24 hours in Gerrit ?	No		
	Are all the Jenkins jobs successfully passed (verify + merge jobs)?	Yes	Verify Jobs Merge Jobs	
Integration and Testing	Have all CSIT Use Cases (created by each project team) passed ?	Yes	CSIT Jobs	
	Have all end-to-end CSIT (created by Integration Team) been identified?	Yes	Link to evidence	
	Is there a Docker images available for each repository?	Yes	VNF-SDK Docker	
	Has the project code successfully passed the Daily Build process?	Yes	VNF-SDK Daily Build	Goal is to ensure the latest project commit hasnot broken the Integration Daily Build
Documentation	Is the team ready to contribute in the following 3 documentations: 1. Mercury Release Notes 2. Mercury Tarball Installation 3. Mercury Docker Installation	Yes		Each service in VNF-SDK already have docker, so we dont need Tarball installation

VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC2)

Practice Area	Checkpoint	Yes/No	Evidences	How to?
Product Management	Are all tasks associated with the release been marked as "Done" in Jira?	Yes	VNF-SDK JIRA Task	
Release Management	(For RC0 only) As the project team identified the reduced set of Committers who will proceed with Code Merge after RC0?	NA	Provide the name of committers. Recommendation is 3 committers max per project (1 per geographic area)?	
	Have all findings from previous milestones been addressed?	Yes	VNF-SDK JIRA Issues	
Development	Have all Defects of priority Highest and High been in status "Done" in Jira?	Yes	VNF-SDK JIRA Issues	
	Are all JIRA issues used as a reference in a commit in status "Done" in Jira?	Yes		Very often, some JIRA issue remains in status "Todo" in JIRA while the commit they are referenced to is successful.
	Is there any pending commit request older than 24 hours in Gerrit?	No		
	Are all the Jenkins jobs successfully passed (verify + merge jobs)?	Yes	Verify Jobs Merge Jobs	
Integration and Testing	Have all CSIT Use Cases (created by each project team) passed?	Yes	CSIT Jobs	
	Have all end-to-end CSIT (created by Integration Team) been identified (for RC0) and passed (for RC1)?	Yes	Link to evidence	
	Is there a Docker images available for each repository?	Yes	VNF-SDK Docker	
	Has the project code successfully passed the Daily Build process?	Yes	VNF-SDK Daily Build	Goal is to ensure the latestproject commit has not broken the Integration Daily Build

Documentation	Is the team ready to contribute in the following 3 documentations: 1. Mercury Release Notes 2. Mercury Tarball Installation 3. Mercury Docker Installation	Yes		

VNF SDK Mercury Release Functionality Freeze Deliverables (M2)

Practice Area	Checkpoint	Yes/No	Evidence - Comment	How to?
Product Management	Are all provisional APIs interface (stub) been defined (at beta-quality level)?	Yes	Function Test Interface	
	Is there a final list of externally consumable APIs available?	Yes	VNF-SDK Outgoing APIs Market Place API	
	For all completed Sprints, have Sprint Backlog Stories been marked as "Done" in Jira?	Yes	VNF-SDK Sprint 1	Difference between a Product and Sprint Backlog
	Are all tasks associated with the completed Sprint Backlog Stories been marked as "Done" in Jira?	Yes	VNF-SDK Sprint 1	
	Has all Product Backlog not implemented in current release been scoped for next release?	Yes	JIRA Venus Release Backlog	to document
Release Management	Have all source code files been updated with License Apache 2 header?	Yes		Definition of Source code file
	Has the year format in copyright header of all source code files been updated? (Rules for new files created in 2017 and existing files modified in 2017 are different)	Yes		Guidance on year format
	In case source code can't be edited, has a "License.txt" file been placed at the root directory for which the license is applicable?	Yes		Guidance for source code file that can't be edited
	Has the project FOSS Table been updated with latest third party code?	Yes	VNF-SDK FOSS	
	Do you have a plan to address any issue raised by Fossology?	Yes	TBD	
Development	For new projects approved for this release, has all source code been placed into Gerrit?	Yes	VNF-SDK Gerrit MarketPlace/Design Pkg Tool/Function Test already committed Source code, lc test still coding now	
	Has the project team reach the Automated Unit Test Code Coverage expectation? (Refer to artifacts available in Sonar)	No	Goal: 50% for Incubation project Now Only UI Code. UT Coverage not involved. 71% Code Coverage reached for repo vnf-sdk-design-pkg (code in python, report made on local env)	Guidance on Code Coverage and Static Code Analysis Tools: Sonar
	Is there any binaries (jar, war, tar, gz, gzip, zip files) in Gerrit project repository?	No	• **NEVER** embed jar, war, tar, gz, gzip, zip in Gerrit	Practices for Mercury Release
	Could you ensure that all proprietary trademarks, logos, product names, company name, etc. have been removed? All OPEN-O deliverables must comply with this rule and be agnostic of any proprietary symbols.	Yes		
	Is there any pending commit request older than 24 hours in Gerrit?	No		
	Are all the Jenkins jobs successfully passed (merge jobs)?	Yes	VNF-SDK Jenkins Job	

	Are all snapshot binaries available in Nexus?	In Progress	VNF-SDK Nexus Sth Wrong with lc-test repo	
Integration and Testing	Have Continuous System Integration Testing (CSIT) Use Cases been documented in wiki?	Yes	VNF SDK Testing	
	Have you implemented in Jenkins at least 1 CSIT test case for each of the project repository?	In Progress	As an evidence, provide a link to Jenkins (CSIT Jobs) that shows a sample test case implemented (1 job for each repo). VNF SDK Function Test CSIT VNFSDK-55	As an example (provided by Integration Team)
	Has the project code successfully passed the Daily Build process?	Yes	VNF-SDK Merge Job	

VNF SDK Mercury Release Planning (M1)

Overview

Project Name	**Enter the name of the project**
Target Release Name	VNF SDK
Target Release Name	Mercury Release

Project Lifecycle State	Incubation
Participating Company	CMCC, Gigaspaces, Huawei, Intel, ZTE

Scope

What is this release trying to address?

- Currently VNF on-boarding, instantiation and maintenance procedures, used by NFVI operators are highly manual, time consuming and specifically tailored to individual operators, VIMs, NFV infrastructures and VNF types
- VNF suppliers want to participate in a large ecosystem for more market opportunities, and to develop their VNF products in a way that is attractive to the largest range of buyers
- In order to support multiple service and business scenarios, VNF suppliers want to package and deliver their VNF products in a way that is highly standardized, automated and configurable
- VNF suppliers, who rely on specific NFVI and VIM capabilities, as well as telemetry feedback, want to ensure that their VNF infrastructure and VIM dependencies are met, so that their VNF product can function as designed
- Current versions of ETSI/MANO VNF package (VNFD) and VNF record(VNFR) definitions prevent VNF suppliers from expressing NFVI and VIM dependencies and requirements*

Use Cases

This release will also add support of the Mercury release Customer Use Case: Mercury Use Case.

Minimum Viable Product

MVP has the following features:

- Design & Package Tools
- Validate & Lifecycle Test
- Functon Test
- Market Place

Functionalities

List the functionalities that this release is committing to deliver.

Functionality Name	In or Out	Priority	Stretch
Package & Designer Tools	IN	H	VNF Package Designer, part of VNF Supplier SDK tools. The below are the tasks we need to do: • The command line interface(CLI) for designer (optional?). • Provider a graphical tool to define the VNF product model and package for VNF vendor engineer. • Standardized VNF product packaging based on TOSCA • VNF Package builder • VNF Package validator • VNF Package extractor • VNF Package Parser • VNF Package Dry Run • VNF Model
Validate & Lifecycle Test	IN	H	Lifecycle test should do the task as below: • Reuse the onboarding operation for VNF package in NFVO • Reuse the installation operation for VNF Package in NFVO • Reuse the resource allocation operation for VNF package in NFVO • Reuse the workflow in NFVO
Functional Test	IN	H	The function test for VNF-SDK should do as below: • Supply a FT framework for VNF product developer/ NFV operator/VNF product Dev-Ops teams • Support some measurement(like VNF KPI) about the FT

Market Place	IN	H	Market place for VNF-SDK should do the task as below: • Upload and download of VNF products and offerings • Supply a user-friendly GUI for operator/User • Show the interface about other component in VNF-SDK(just like Lifecycle test, Functional Test, Upload, Download, etc)

Longer term roadmap

- SDN-O consideration
    - PNF
    - VNF FT that SDN-O related(connectivity)
    - Controller/Driver certification
- Network Service On-boarding consideration

Release Deliverables

Indicate the outcome (Executable, Source Code, Library, API description, Tool, Documentation, Release Note...) of this release.

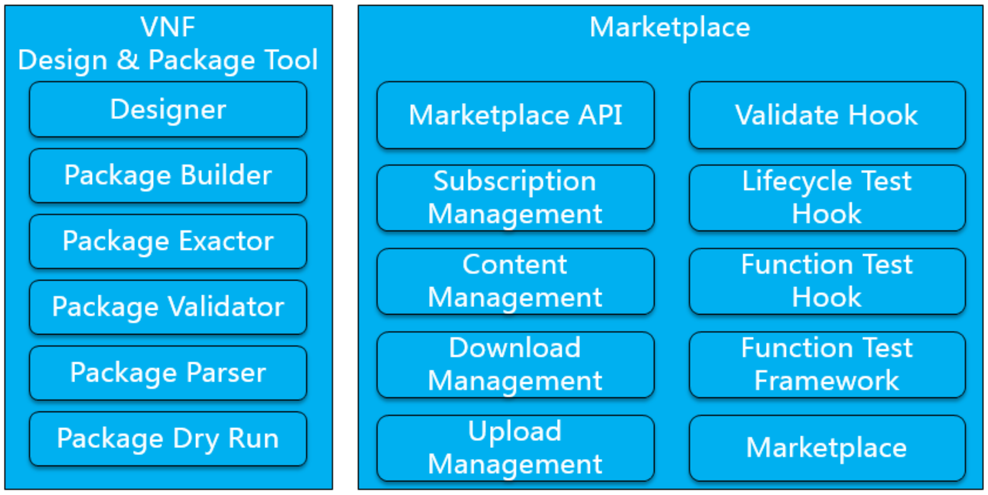
Deliverable Name	Deliverable Description
Design & Package Tool	Provider a graphical tool to define the VNF product model and package for vnf vendor engineer
Validate & Lifecycle Test	Validate the package from vendor, and do the lifecycle test for it.
Function Test	Supply a test framework for vendor, so that before upload to Market place, provider can know the basic function is OK
Market Place	For publication, upload and download of VNF products and offerings

Sub-Components

See the repository table.

Architecture

Include High level architecture diagram.



API Dependencies

List the API this release is expecting from other releases.
Prior to Release Planning review, Team Leads must agreed on the date by which the API will be fully defined. The API Delivery date must not be later than the release Functionality Freeze date.

Prior to the delivery date, it is a good practice to organize an API review.

API Incoming Dependencies

API Name	API Description	API Definition Date	API Delivery date	API Definition link (i.e.swagger)
Mircro Service Bus API	API for registration of the microservices		Feb 16, 2017	Microservice_Bus_API_Documentation
NFVO Lifecycle APi	API for test NFVO lifecycle and Validation		Feb 16, 2017	NFVO LCM API
Model Designer APi	API for common tosca model designer		Feb 16, 2017	TBD
Function Test API	API for Function framework API		Feb 16, 2017	Function Test

API Outgoing Dependencies

API this release is delivering to other releases.

API Name	API Description	API Definition Date	API Delivery date	API Definition link
Download API	API for Operator download VNF Marketplace's Package	Jan 24, 2017	Feb 16, 2017	Market Place
Upload API	API for VNF vendor upload VNF Package to VNF Marketplace	Jan 24, 2017	Feb 16, 2017	Market Place

Third Party Products Dependencies

Currently, VNF SDK didnt have third party dependencies.

Testing and Integration Plans

Function Test Plan and Resources

Describe the plan to integrate and test the release deliverables with the overall OPEN-O system.
Confirm that resources have been allocated to perform such activities.
The Release lifecycle accounts for an Testing and Integration Review where detailed plan are expected.

This activity must be in sync with Release Integration Review milestone (RC0).

Projects may apply for a system test waiver if they think they have top-level features not requiring system test or covered by other top-level features test.

Projects must specify whether they plan to use Open-O CI infrastructure for system test. It is recommended to use the Open-O CI infrastructure unless there is some HW or SW resource that cannot be installed there. Projects running system test in external Labs are required to report system test results in a timely fashion after release creations, e.g., weekly, RC, and formal releases.

Gaps

This section is used to document a limitation on a functionality or platform support. We are currently aware of this limitation and it will be delivered in the future.
List identified release gaps (if any), and its impact.

Gaps identified	Impact
To fill out	To fill out

Known Defects and Issues

Provide a link toward the list of all known project defects.

Risks

List the risks identified for this release along with the plan to prevent the risk to occur (mitigation) and the plan of action in the case the risk would materialized (contingency).

Risk identified	Mitigation Plan	Contingency Plan
None	None	None

Resources

Resources assigned to project are consolidated in a single place. Edit all the table with the **ALL** information, that helps to expedite Linux Foundation operations.

Release Milestone

The milestones are defined at the Release Level and all the supporting project agreed to comply with these dates.

Team Internal Milestone

Project Plan

Edit Document

Sub-Project	Owner	task	Participate
GUI Design&Package Tool	zhaohuabing	Designer	committer victor. wan.dong@zte.co lv.bo163@zte.com.c zhao.huabing@zte. qin.lihan@zte.com.cr
Package Tools		VNF Model	Committer krzysztof.frukacz@gi tal@gigaspaces.com
	Krzysztof	Package Builder	
		Package Exactor	
		Package Validator	
		Package Parser	
		Package Dry Run	
Validate & Lifecycle Test	zhanganbing	API between Validation and NFVO	committer: gong.mingsheng@z zhanganbing@china
		Validate Hook	
		Hook for reusing NFVO onboarding Operation	
		Hook for reusing NFVO installation operation	

		Hook for reusing NFVO resource allocation operation API between LC Test and NFVO/VNFM Hook for reusing NFVO workflow	contributor wang.fe liuyuanyjy@chinamol
Function Test	Murali	Test Framework Selection & guidance of framework Measurement for Function Test API between FT and VNFM/NFVO Function Test hook	committer: mura gong.mingsheng@z zhanganbing@china contributor wang.fe liuyuanyjy@chinamol
Market Place	Sukesh/Alex	Upload Management Download Management Subscription Management Market Place GUI API between GUI and MicroSerivce	sukeshac@huawe Alex.vul@intel.com

Vendor Neutral

If this project is coming from an existing proprietary codebase, ensure that all proprietary trademarks, logos, product names, etc. have been removed. All OPEN-O deliverables must comply with this rule and be agnostic of any proprietary symbols.

Free and Open Source Software

FOSS activities are critical to the delivery of the whole OPEN-O initiative. The information may not be fully available at Release Planning, however to avoid late refactoring, it is critical to accomplish this task as early as possible.
List all third party Free and Open Source Software used within the release and provide License type (BSD, MIT, Apache, GNU GPL,...).
In the case non Apache License are found inform immediately the TSC and the Release Manager and document your reasoning on why you believe we can use a non Apache version 2 license.

Each project must edit its table within the FOSS VNF SDK

Documentation, Training, Tutorial

- High level list of documentation, training and tutorials necessary to understand the release capabilities, configuration and operation.
- Documentation includes items such as:
  - Installation instructions
  - Configuration instructions
  - Developer guide
  - End User guide
  - Admin guide

Note
It is expected all materials to be published directly either in the wiki or in OPEN-O website. Do not provide deliverable in PDF, powerpoint or word documents (these format are not easy to edit and lead to versioning issues).

Board policy (including IPR)

Indicate if the release meets the Board policy.

Release key facts

Link to VNF SDK Release Key Facts

VNF SDK Mercury Release Planning Delverables (M1)

The following items are expected to be completed for the project to **pass the M1 Release Planning Milestone**.

M1 Release Planning Milestone definition.

Practice Area	Checkpoint	Yes/No	Evidence - Comment	How to?
Product Management	Are Product Backlog Epics entered in Jira?	Yes	12 issues	Create a Backlog item Difference between a Product and Sprint Backlog
	Are Product Backlog Stories entered in Jira?	Yes	28 issues	Create a Backlog item
	Are Product Backlog Stories linked to Product Backlog Epics?	Yes	VNF SDK Backlog	Work in a Sprint
	Are Product Backlog Stories prioritized?	Yes		Prioritize a Backlog item
	Is the project team ready to perform estimate for the top Story (for coming Sprint) in Product backlog?	Yes		Estimate a Backlog item
	Is the project team ready to create a 2 weeks Sprint in Jira?	Yes		Create a Sprint
	Is Team Member willing to create Tasks and associate them with Stories in Jira?	Yes		Create a Backlog item
Release Management	Is there a Release Planning Template availabl e and completed in wiki?	Yes	VNF SDK Mercury Release Plan	
	Have all the **"Release Components Name"** b een defined inResources and Repositories for your project? (this includes all Sub-Components Names, Sub-Components Repositories Names, Maven Group ID, Sub-Components Description)	Yes	Resources and Repositories#ReleaseComponentsName.8	
	Have all the **"Resources committed to the Release"** been defined in Resources and Repositories for your project? This includes First and Last names, LFID, Email Address and Location for PTL, Project Manager, Committers and Contributors.	Yes	Resources and Repositories#ResourcescommittedtotheRelease.7	
	Have new developers made themself familiar on theOnboarding Process?	Yes		Onboarding
	Is the project team aware of the Release milestone? Any misses will required TSC exception.	Yes		

Integration and Testing	Is the Project Team committed to automate Unit Test?	Yes	UT Coverage >= 50%	
	Is the Automated Unit Test infrastructure in place?	Yes		
	Is the project team committed to create Continuous System Integration Testing (CSIT) test case?	Yes		
Development	Is the project team committed to perform Scrum ceremonies?	Yes		Scrum Overview
	Are the project team member aware of Continuous Integration Principles (don't break the build, Fix the build,...)?	Yes		Continuous Integration
	Has the project team clear understanding on the Code Coverage expectations?	Yes		Code Coverage and Static Code Analysis
	Does the project team understand the Free and Open Source Software (FOSS) process?	Yes		Free and Open Source Software
	Is the project team willing to fill out accordingly the FOSS table?	Yes		Fill out sub-pages for each project under Free and Open Source Software
	Is the project team aware of the Commit Process?	Yes		Commit Messages
	Does the project team understand the purpose of Code Review?	Yes		Code Review
	Is the project team aware of the Coding Guidelines?	Yes		Development Practices (section related to Coding Guidelines)

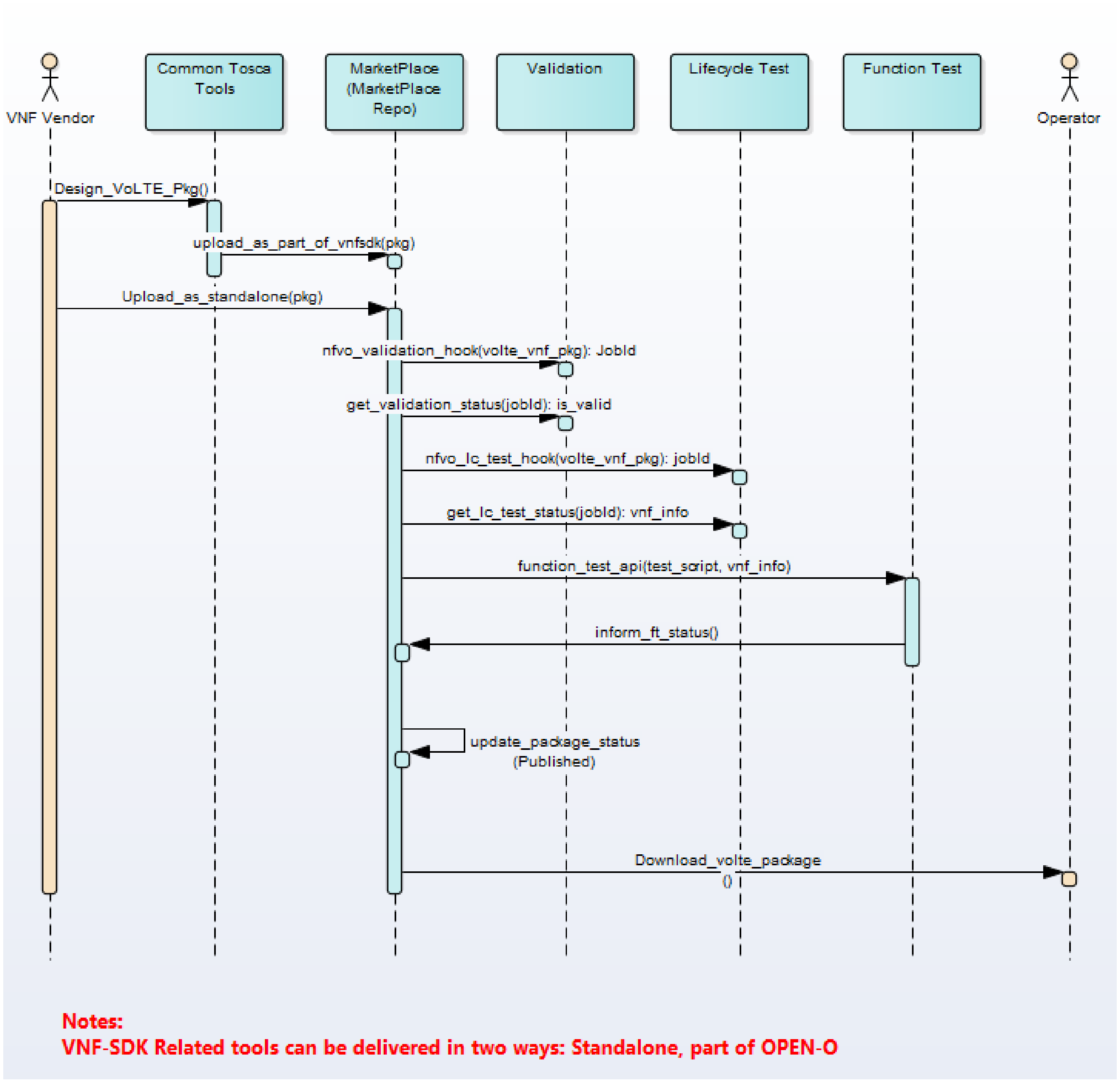
VNF SDK Architecture and Design

This page and the child pages capture the overall VNF SDK Architecture and Design, including various artifacts such as Information Models, including static diagrams such as component, class, deployment, object and package diagrams and behavioral diagrams including use case, sequence, activity and state diagrams.

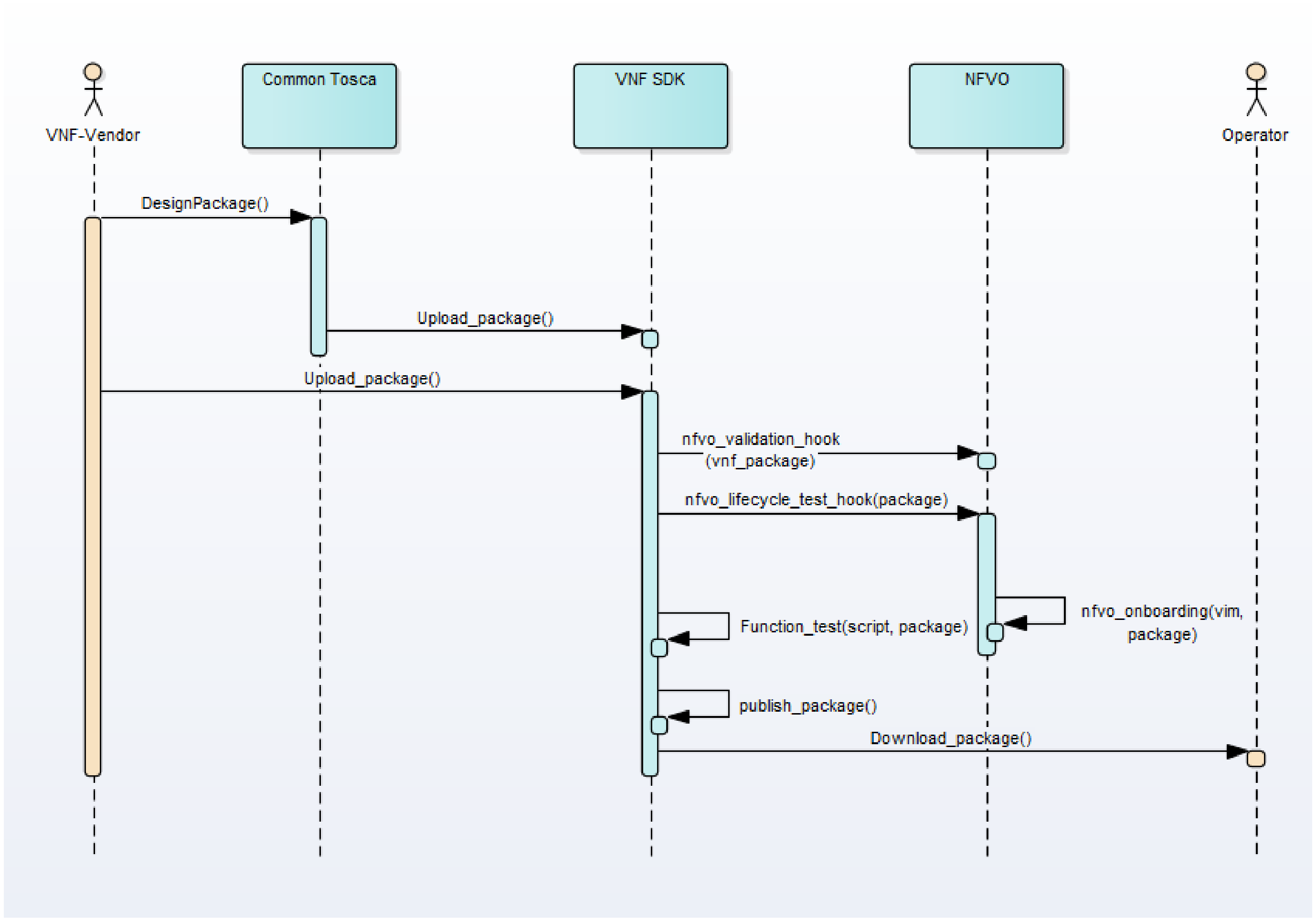
- VNF SDK Flow Diagram
- VNF SDK Use Case Definitions

VNF SDK Flow Diagram

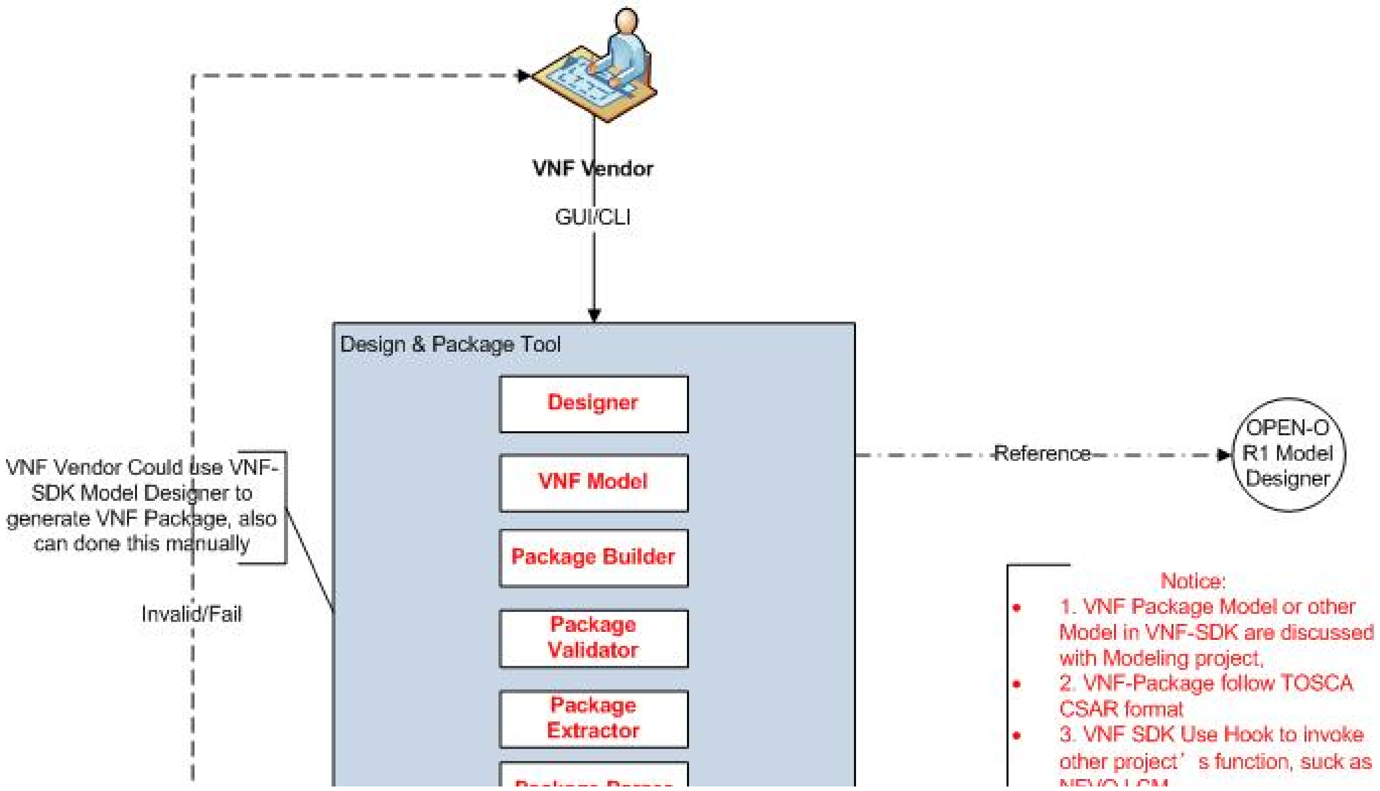
High-Level Sequence Diagram Create Package

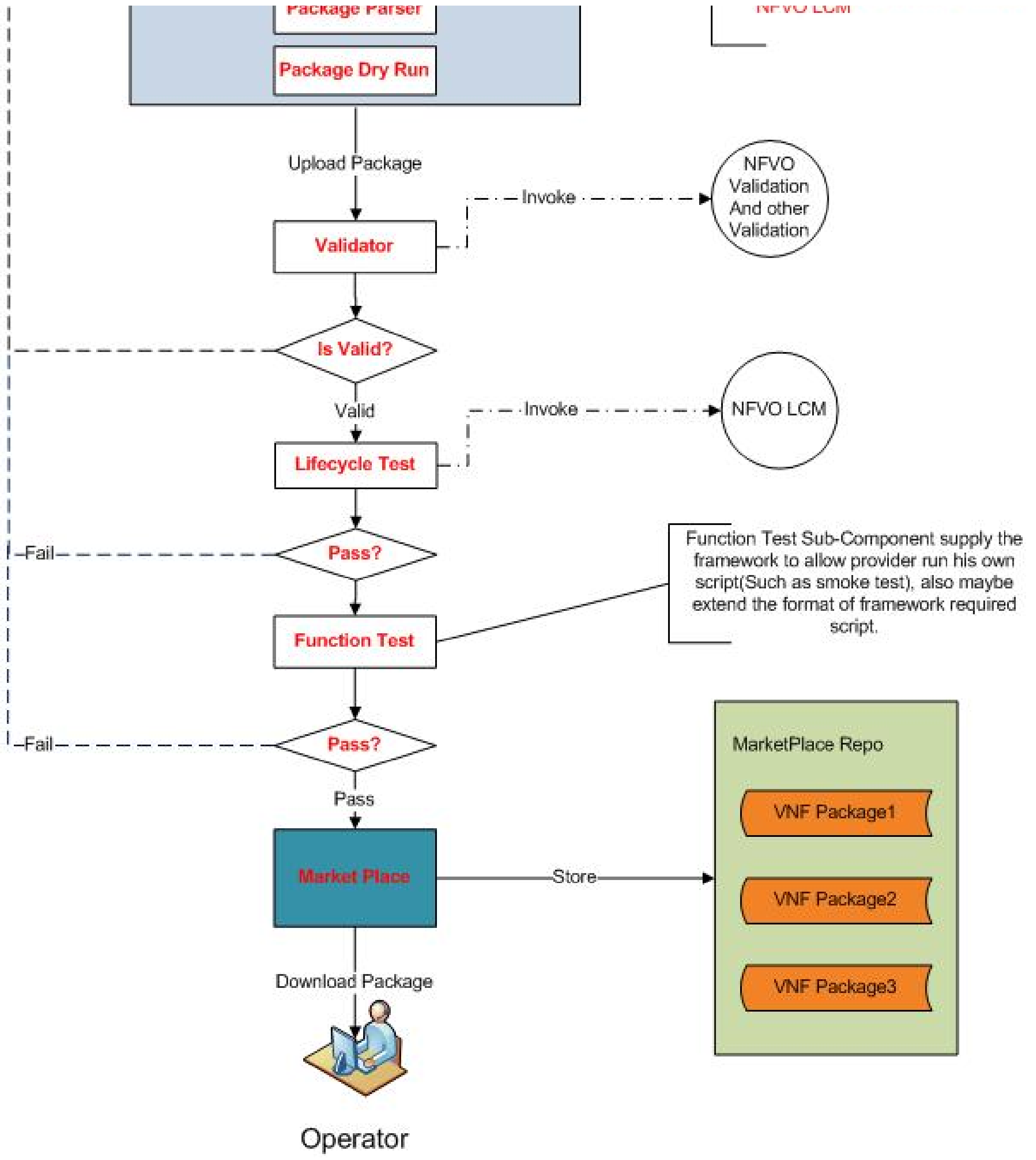

VNF Vendor
Common Tosca Tools
MarketPlace (MarketPlace Repo)
Validation
Lifecycle Test
Function Test
Operator
Design_VoLTE_Pkg()
upload_as_part_of_vnfsdk(pkg)
Upload_as_standalone(pkg)
nfvo_validation_hook(volte_vnf_pkg): JobId
get_validation_status(jobId): is_valid
nfvo_lc_test_hook(volte_vnf_pkg): jobId
get_lc_test_status(jobId): vnf_info
function_test_api(test_script, vnf_info)
inform_ft_status()
update_package_status (Published)
Download_volte_package ()
Notes:
VNF-SDK Related tools can be delivered in two ways: Standalone, part of OPEN-O

Relationship Between VNF-SDK and Other Project


VNF-Vendor
Common Tosca
VNF SDK
NFVO
Operator
DesignPackage()
Upload_package()
Upload_package()
nfvo_validation_hook (vnf_package)
nfvo_lifecycle_test_hook(package)
Function_test(script, package)
nfvo_onboarding(vim, package)
publish_package()
Download_package()

User(Vendor/Operator) Perspective Flow


VNF Vendor
GUI/CLI
Design & Package Tool
Designer
VNF Model
Package Builder
Package Validator
Package Extractor
Reference
OPEN-O R1 Model Designer
VNF Vendor Could use VNF-SDK Model Designer to generate VNF Package, also can done this manually
Invalid/Fail
Notice:
1. VNF Package Model or other Model in VNF-SDK are discussed with Modeling project,
2. VNF-Package follow TOSCA CSAR format
3. VNF SDK Use Hook to invoke other project's function, suck as



VNF SDK Use Case Definitions

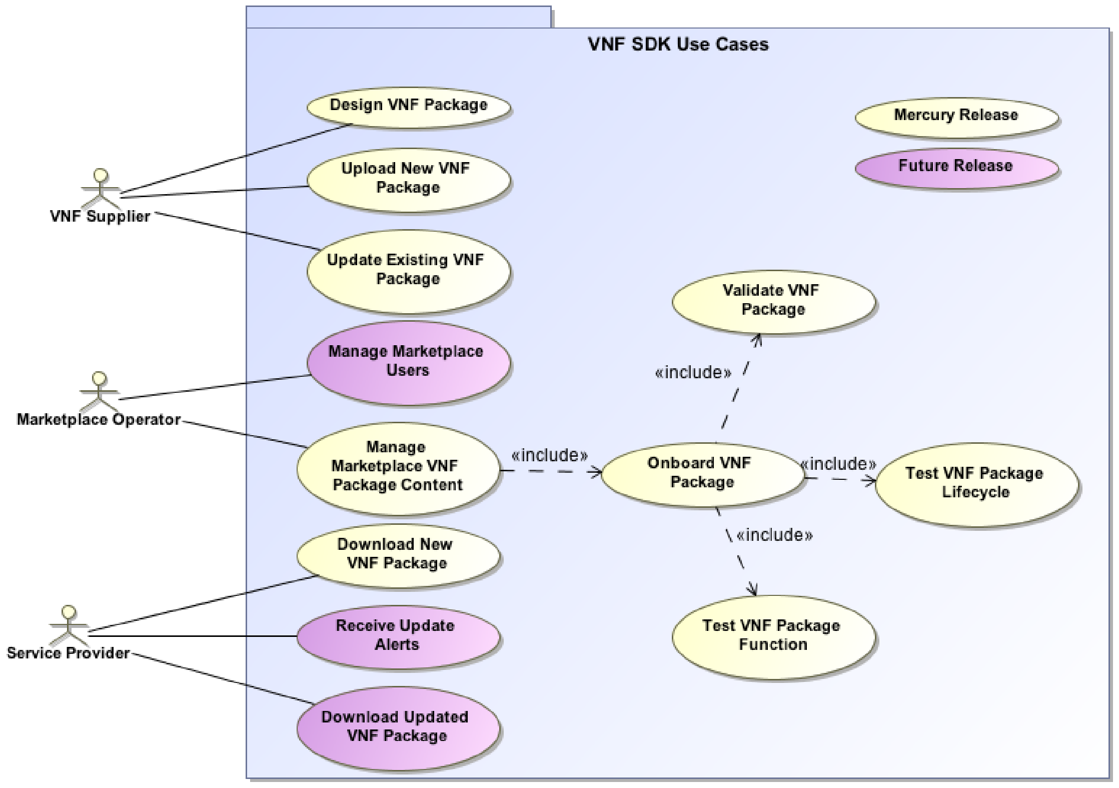
The following use cases are work in progress.

- Mercury Use Cases
    - Detailed Use Case Definitions
        - Design VNF Package
        - Upload New VNF Package
        - Update Existing VNF Package
        - Manage Marketplace Users (Deferred)
        - Manage Marketplace VNF Package Content
            - Onboard VNF Package
                - Validate VNF Package
                - Test VNF Package Lifecycle
                - Test VNF Package Function

    - Download New VNF Package
    - Receive Update Alerts (Deferred)
    - Download Updated VNF Package (Deferred)
- Use Case Personas
    - VNF Supplier
    - Marketplace Operator
    - Service Provider

Mercury Use Cases

Use cases define how different users interact with a system under design. Each use case represents an action that may be performed by a user (defined in UML as an Actor with a user persona).



Detailed Use Case Definitions

This section defines each use case in greater details.

Design VNF Package

Section	Description
ID	VNF-SDK-UC-01
Title	Design VNF Package
Description	In this use case, the VNF supplier uses the TOSCA Common Toolkit or the VNF SDK to design and create a new VNF package which performs a specified network function.
Primary Actor	VNF Supplier
Preconditions	
Main Success Scenario	

Alternate Scenarios	
Exception Scenarios	
Post Conditions	A fully designed and specified VNF package ready for the market.

Upload New VNF Package

Section	Description
ID	VNF-SDK-UC-02
Title	Upload New VNF Package
Description	In this use case, the VNF Supplier uploads the market-ready VNF package to the marketplace staging area.
Primary Actor	VNF Supplier
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	VNF package resides in the staging area of the marketplace and Marketplace Operator has been notified

Update Existing VNF Package

Section	Description
ID	VNF-SDK-UC-03
Title	Update Existing VNF Package
Description	In this use case, the VNF Supplier needs to update a VNF package that has already been published, perhaps fixing a defect in the existing package, or adding a new feature to the package.
Primary Actor	VNF Supplier
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	The updated VNF package resides in the staging area of the marketplace and Marketplace Operator has been notified

Manage Marketplace Users (Deferred)

Section	Description
ID	VNF-SDK-UC-04
Title	Manage Marketplace Users

Description	
Primary Actor	Marketplace Operator
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	

Manage Marketplace VNF Package Content

Section	Description
ID	VNF-SDK-UC-05
Title	Manage Marketplace VNF Package Content
Description	In this Use Case, the Marketplace Operator has the responsibility of managing all VNF Package Content in the marketplace.
Primary Actor	Marketplace Operator
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	All new and updated VNF package content is managed and controlled according to Marketplace Operator policies.

Onboard VNF Package

Section	Description
ID	VNF-SDK-UC-05-01
Title	Onboard VNF Package
Description	In this Use Case, the Marketplace Operator onboards a VNF package placed in the staging area (e.g., newly uploaded package).
Primary Actor	Marketplace Operator
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	The new VNF package has been successfully onboarded

Validate VNF Package

Section	Description
ID	VNF-SDK-UC-05-01-01
Title	Validate VNF Package

Description	In this Use Case, the Marketplace Operator validates the VNF package to ensure it complies with package rules.
Primary Actor	Marketplace Operator
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	The VNF package is validated.

Test VNF Package Lifecycle

Section	Description
ID	VNF-SDK-UC-05-01-02
Title	Test VNF Package Lifecycle
Description	In this Use Case, the Marketplace Operator tests the end-to-end lifecycle of the VNF package.
Primary Actor	Marketplace Operator
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	The VNF Package passed the lifecycle testing.

Test VNF Package Function

Section	Description
ID	VNF-SDK-UC-05-01-03
Title	Test VNF Package Function
Description	In this Use Case, the Marketplace Operator tests the network function of the VNF package.
Primary Actor	Marketplace Operator
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	The VNF Package passed the function testing.

Download New VNF Package

Section	Description
ID	VNF-SDK-UC-06
Title	Download New VNF Package
Description	In this Use Case, the Service Provider searches, finds and downloads a VNF package that meets their network service needs.

Primary Actor	Service Provider
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	VNF SDK package successfully downloaded by Service Provider

Receive Update Alerts (Deferred)

Section	Description
ID	VNF-SDK-UC-07
Title	Receive Update Alerts
Description	
Primary Actor	Service Provider
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	

Download Updated VNF Package (Deferred)

Section	Description
ID	VNF-SDK-UC-08
Title	Download Updated VNF Package
Description	
Primary Actor	Service Provider
Preconditions	
Main Success Scenario	
Alternate Scenarios	
Exception Scenarios	
Post Conditions	

Use Case Personas

This section describes the Use Case personas or UML Actors.

VNF Supplier

The VNF Supplier has responsibility for designing, defining and developing a VNF package which Service Provider has a need for.

Marketplace Operator

The Marketplace Operator has responsibility for managing and controlling the marketplace environment, including managing users and VNF content placed in the marketplace.

Service Provider

The Service Provider is the consumer of the VNF packages in the marketplace. A particular VNF is downloaded to meet a business need the Service Provider has.

VNF SDK Blueprint

This page presents our work in progress and discussion towards defining a TOSCA "template" for VNF vendors to use when bundling their VNFs.

History

Draft 1: Nov 24 2016. Opening the discussion. Based on input from Krzysztof Frukacz, Tal Liron, Alex Vul, Michael Brenner, Nati Shalom.

Draft 2: Nov 29 2016. Performance profiles example corrected. Krzysztof Frukacz.

Draft 3: Dec 8 2016. Added KPI and simple Ping Pong example.

Draft 4: 4 Jan 2016. Added analysis document for EPA in Openstack vs VNF profile

Draft 5: 24 Jan 2017 Updated the definition

Draft 6 10 Feb 2017 Removed references to old version. Updated description

Files

pingpong.yaml - Simple example VNF blueprint.

vrouter.yaml - Fuller example blueprint for a complete virtual router VNF.

vnfsdk-profile-1.0.yaml - Imported types for the above profiles.

compute-profiles.yaml - Groups dependencies together as “compute profiles”.

Guiding Principles

1. **Adhere** to TOSCA standard and its simple profile for NFV
    a. By continuing to iterate, we will be able to identify the pain points and advise the OASIS community on fixing and enhancing the profile
2. **Ensure** interoperability with greater community
    a. TOSCA is open-ended, so our challenge is to make sure that Open-O’s VNF blueprints could be consumed by other MANO implementations
3. **Expose** infrastructure optimizations
    a. Hardware platform (e.g. Intel EPA)
    b. Consideration for containers (in addition to virtual machines and bare metal)
    c. Entry points for quality-assurance telemetry

Reading the Blueprint

We try to make the blueprint as humanly readable as possible, but until there is a good TOSCA IDE that can show you inheritance and do live validation (ARIA is designed to support one in the future), you will have to look up the node/capability/relationship types yourself.

Please use the standard profiles as reference while you read.

Working with the Blueprint

We recommend that you make sure that the blueprint is valid as you edit it. You can use ARIA to test your blueprint and see the final deployment plan.

Install requirements on Ubuntu:

```
sudo apt install python-setuptools
sudo -H easy_install pip
sudo -H pip install virtualenv
```

Install ARIA:

```
virtualenv env
. env/bin/activate
pip install
git+http://git-wip-us.apache.org/repos/asf/incubator-ariatosca.git
```

Run ARIA parser:

```
aria parse vrouter.yaml
```

Current Limitations

TOSCA currently has two profiles: **tosca_simple_yaml_1_0**, and **tosca_simple_profile_for_nfv_1_0**, which extends the former.

But they are too open ended. For example, there are basic policy types (**Scaling**, **Placement**, **Performance**) but they are empty, waiting to be extended by actual implementations.

Unfortunately, for VNF vendors this means that there is no reference implementation for them to target, and that work would have to be duplicated for each implementation.

An analisis provided here contains a few example EPA features that are supported by Openstack. A lot of them are not covered by TOSCA simple profile for NFV v1.0.

Extensions

We are suggesting that extensions to the profile are necessary to create a shared common ground between VNF vendor and operator. This **tosca _vnfsdk_profile_1_0** extends on the two other profiles and introduces more specific types (policies and relationships), while still leaving room for implementations to provide their own algorithms and systems and compete.

Combining the Worlds of VNF and Compute

As it stands, the TOSCA NFV profile CSD03 types are mostly isolated from the basic profile types. Specifically, **tosca.nodes.nfv.VDU** duplicates the logic of **tosca.nodes.Compute**. It is impossible to relate them as is, because **Compute**'s host capability only allows for **tosca.nodes.Softwar**

eComponent sources, and VDU does not derive from **SoftwareComponent**.

Without changing the NFV profile for now, we are suggesting a workaround: use the dependency requirement, which all nodes support, but extend **tosca.relationships.DependsOn** to allow us to configure the relationship between the **VDU** and **Compute** nodes. This allows us to reuse types and mechanizms defined in TOSCA simple YAML profile with our NFV profile nodes, like **VDU**.

We will expand on this example relationship later on:

```
node_templates:

 router_vdu:
    type: tosca.nodes.nfv.VDU
    requirements:
      - dependency:
          node: router_host
          relationship:
             type: vnfsdk.DependsOn

 router_host:
    type: tosca.nodes.Compute
    capabilities:
      scalable:
        properties:
          max_instances: 4
```

Challenge of Infrastructure Optimization

We believe that **tosca.capabilities.Compute.Container.Architecture** as defined in the TOSCA NFV profile is entirely insufficient for VNF vendors.

However, we also believe that extending this basic type with a large "shopping list" of properties would be worse, as:

1. we cannot anticipate future (and even near-future) technology
2. the current state of resource orchestration (there is none) means that shopping lists could not be fulfilled to satisfaction.

Compute Dependencies

Instead, we suggest a three-way approach:

1. Support the TOSCA NFV properties as is, as a lowest-common denominator of absolute requirements for the VDU, and recommend that this feature be deprecated.
2. Allow VNF vendors to specify arbitrary dependencies for the algorithm via an extensible spec that allows for future technological innovation.
3. Group features together as "compute profiles" (e.g. "high_availability", "dpdk_enabled", "ovs_ready") and use TOSCA policies to map the VNF vendor's annotations to profiles using an algorithm. These performance profiles will be configured by the operator rather than the VNF vendor.

Compute Dependencies

```
ping_vdu:
      description: >-
        The "ping" VDU generates traffic.
      type: tosca.nodes.nfv.VDU
      interfaces:
        Standard:
          configure: scripts/vdu/ping_configure.sh # included in the CSAR
      requirements:
        - dependency:
            node: ping_host # our Compute node
            relationship:
              type: vnfsdk.DependsOn
              properties:
                # For the "ping" VDU we are choosing various compute
dependencies:
                compute_dependencies:
                  cpu_allocation:
                    string_map:
                      cpu_affinity: { get_input: pong.cpu_affinity }
                      thread_alocation: isolate
                  container:
                    boolean: false
                  dpdk:
                    string: '16.07'
```

Compute Profile Definitions

We suggest that **Compute Profiles** can be defined by the operator using TOSCA to create a "bucket" of copute dependencies. The profiles are suppose to simplfy the definitions of Compute Dependencies, as in a lot of deployments there are many VDUs with the same required EPA features running on the same infrastructure hardware. If a Compute Profile is not used, then all Compute Dependencies have to specified for a given VDU.

```
dpdk:
  description: >-
    DPDK compute profile.
  compute_dependencies:
    dpdk:
      string: { get_input: dpdk.version }
    mem_page_size:
      string: large
      optional: true
    sr-iov:
      boolean: true
    cpu_allocation:
      string_map:
        cpu_affinity: dedicated
```

Below is an example of ping_vdu that specifies a pre-defined profile instead of specifying every Compute Dependency explicitly.

```
ping_vdu:
      description: >-
        The "ping" VDU generates traffic.
      type: tosca.nodes.nfv.VDU
      interfaces:
        Standard:
          configure: scripts/vdu/ping_configure.sh # included in the CSAR
      requirements:
        - dependency:
            node: ping_host # our Compute node
            relationship:
              type: vnfsdk.DependsOn
              properties:
                # For the "ping" VDU we are choosing a profile that already
includes
                # various compute dependencies:
                compute_profile: dpdk
```

Telemetry

Our goal here is neither comprehensive FCAPS nor to step into the domain of NETCONF and YANG, but rather to allow VNF vendors to define arbitrary indicators for measuring acceptable and good performance.

```
ping_vdu:
      description: >-
        The "ping" VDU generates traffic.
      type: tosca.nodes.nfv.VDU
      interfaces:
        Standard:
          configure: scripts/vdu/ping_configure.sh # included in the CSAR
      requirements:
        - dependency:
            node: ping_host # our Compute node
            relationship:
              type: vnfsdk.DependsOn
              properties:
                # We'll expose two indicators from "ping" that can be used
by monitoring tools:
                indicators:
                  hits:
                    type: int
                    implementation: scripts/vdu/ping_get_hits.sh # included
in the CSAR
                  connectivity:
                    type: string
                    implementation: { get_input:
ping.connectivity.indicator }
```

Integration with Service Chains

The VNF's CSAR will need to be integrated into service chains. We will use TOSCA substitution mapping to treat the entire blueprint as one logical VNF, specifically mapping our internal connection to the consuming blueprint:

```
substitution_mappings:
   node_type: tosca.nodes.nfv.VNF
   requirements:
     virtual_link: [ ingress_route, virtual_link ]
```

See Also

- Tacker vnfd
- T-Nova VNF

VNF SDK - Guide for Bundling VNFs

DRAFT

Step 1: Requirements

1. Virtual machine images for each of your VDUs (likely in qcow2 format).
2. Scripts for installing and configuring each VDU per the standard TOSCA lifecycle workflows.
3. Optional: scripts for retrieving indicators from each VDU.
4. The VNF SDK.

Step 2: Start with the provided VNF blueprint as a template

The VNF SDK comes with a "pingpong.yaml" template.

Copy the file and rename it as is appropriate for your VNF.

Step 3: Edit the VNF blueprint

You will need a node template for each VDU, which in turn has a "dependency" requirement on a host (a Compute node template).

The "dependency" requirement should have a relationship of type "vnfsdk.DependsOn".

There are two optional properties you can assign to this relationship:

1. "indicators" is a map of indicators. The keys are arbitrary, for you to name, and depend on the specific capabilities of your VNF. These provide the link between TOSCA and your indicator retrieval scripts.
2. "compute_dependencies" is a map of features you require the infrastructure to provide for the compute node hosting your VDU. The keys are arbitrary, however we will maintain a master list of names supported by various orchestrators (TBD). Note that new names will likely be supported in the future as technological innovation progresses, while deprecated ones will be ignored by orchestrators. Likewise, names unsupported by a particular orchestrator will be ignored.

Notes:

1. For "compute_dependencies", you have the option of specifying a single special key, "profile", that refers to a bundle of compute dependencies specified in "compute-profiles.yaml". Optionally, you can override any of the dependencies in the bundle by specifying them explicitly under "compute_dependencies".
2. By default, compute dependencies are considered to be absolute requirements for your VNF to function. However, if your VNF can also function without them, you can set "optional: true". This way the orchestrator will try to satisfy the optional requirements, but will not report an error if it cannot.
3. You may optionally consider allowing operators to override your dependencies. This is useful for allowing the VNF to run on various compute infrastructures other than the defaults you specify, as long as they are indeed supported by your VNF. In TOSCA, this is easily

handled by declaring "inputs" for your blueprint, with "default" values, and then using the "get_input" intrinsic function to apply the provided value to any specific "compute_dependencies". The "pingpong.yaml" has an example of such usage. The same technique can be used to allow the operator to override your indicator scripts, and there is an example of that as well.

Otherwise, this is a standard TOSCA blueprint using the Simple Profile for NFV, and you may extend the blueprint as is required.

Step 4: Bundle the VNF

Arrange all your files (the TOSCA template, your scripts, and your virtual machine images) in the correct directory structure. File references provided in the blueprint must match the directory structure. If a script is referenced using a path with sub-directory "example" the initial directory must contain a directory called "example" with that script file inside. The directory structure will be kept inside the CSAR archive.

Run the bundling tool that is included in the VNF SDK.

The result will be a ".csar" file that can now be deployed by operators.

VNF SDK Documentation

Filter By File Extension: Filter By Label:

Name	Size	Creator	Creation Date	Last Modification Date	Labels	Attached To
VNF SDK Use Case Diagram.png	98 kB	Brian Hedstrom	Mar 02, 2017	Mar 02, 2017	No labels	VNF SDK Use Case Definitions
Relationship between VNF-SDK and other project.png	34 kB	gao weitao	Mar 02, 2017	Mar 02, 2017	No labels	VNF SDK Flow Diagram
marketplace ut.png	17 kB	gao weitao	Feb 22, 2017	Feb 22, 2017	No labels	VNF SDK Mercury Release Architecture Milestone Deliverables (M3)
package tool ut.png	20 kB	gao weitao	Feb 22, 2017	Feb 22, 2017	No labels	VNF SDK Mercury Release Architecture Milestone Deliverables (M3)
vnf_sdk_highlevel_sequence_diagram_new.png	40 kB	gao weitao	Feb 21, 2017	Feb 21, 2017	No labels	VNF SDK Flow Diagram
vnf_sdk_highlevel_sequence_diagram_new.png	40 kB	gao weitao	Feb 21, 2017	Feb 21, 2017	No labels	VNF SDK Flow Diagram
Open-O VNF SDK	75 kB	Brian Hedstrom	Feb 16, 2017	Feb 16, 2017	No labels	VNF SDK Documentation

Model.mdzip						
VNF SDK Use Case Diagram.png	61 kB	Brian Hedstrom	Feb 16, 2017	Feb 16, 2017	No labels	VNF SDK Architecture and Design
vnf_sdk_highlevel_sequence_diagram.png	33 kB	gao weitao	Feb 15, 2017	Feb 15, 2017	No labels	VNF SDK Project Proposal
vnf sdk_ validate LC test.PNG	56 kB	Yuan Liu	Feb 10, 2017	Feb 10, 2017	No labels	Validate & Lifecycle Test
.PNG	56 kB	Yuan Liu	Feb 10, 2017	Feb 10, 2017	No labels	Validate & Lifecycle Test
image2017-2-3 10:39:52.png	29 kB	Murali Mohan Murthy Potham	Feb 03, 2017	Feb 03, 2017	No labels	Function Test
Suggestion for some argues in VNF SDK Project(2017…	297 kB	gao weitao	Jan 24, 2017	Jan 24, 2017	No labels	Meeting Materials
pingpong.yaml	6 kB	Krzysztof Frukacz	Jan 24, 2017	Jan 24, 2017	No labels	VNF SDK Blueprint
compute-profiles.yaml	0.6 kB	Krzysztof Frukacz	Jan 24, 2017	Jan 24, 2017	No labels	VNF SDK Blueprint
vnfsdk-profile-1.0.yaml	4 kB	Krzysztof Frukacz	Jan 24, 2017	Jan 24, 2017	No labels	VNF SDK Blueprint
slide1.jpg	135 kB	Murali Mohan Murthy Potham	Jan 24, 2017	Jan 24, 2017	No labels	VNF SDK Old Home Page
slide2.jpg	141 kB	Murali Mohan Murthy Potham	Jan 24, 2017	Jan 24, 2017	No labels	VNF SDK Old Home Page
Enhanced REST Library for Robot.pptx	250 kB	Murali Mohan Murthy Potham	Jan 23, 2017	Jan 23, 2017	No labels	Function Test
Automatic Test Framework Selection for Current Rel…	81 kB	Murali Mohan Murthy Potham	Jan 23, 2017	Jan 23, 2017	No labels	Function Test

VNF SDK Meetings

- Logistics
- VNF SDK Meetings Minutes
  - December 1st, 2016: Meeting Notes
  - December 14th, 2016: Meeting Notes (TBD)
  - January 20, 2017 Meeting Notes
  - February 9, 2017 Meeting Notes

Logistics

Thursdays, 15:30 UTC, 7:30AM PST, 10:30AM EST, 11:30PM China, 4:30PM CET
Slack: http://open-o-tsc.slack.com channel: #vnf-sdk

Info for joining the meeting

Join the Meeting
Join from PC, Mac, Linux, iOS or Android: https://zoom.us/j/978104642
Or iPhone one-tap (US Toll): +14086380968,9781046429# or +16465588656,9781046429#
Or Telephone:
Dial: +1 408 638 0968 (US Toll) or +1 646 558 8656 (US Toll)
Meeting ID: 978 104 642
International numbers available: https://zoom.us/zoomconference?m=tujk2ELwg43SZgDPqB8fffEetH5MNGpp

iCal: OPEN-O VNF-SDK Project Discussion.ics

info for VNF-SDK each Friday meeting

Join the Meeting
Join from PC, Mac, Linux, iOS or Android: https://huawei.zoom.us/j/727367296
Or iPhone one-tap (US Toll): +14086380968,727367296# or +16465588656,727367296#
Or Telephone:
Dial: +1 408 638 0968 (US Toll) or +1 646 558 8656 (US Toll)
Meeting ID: 727 367 296
International numbers available: https://huawei.zoom.us/zoomconference?m=Ih0_jcjjAelLfiOWXvAPbwv8_UOGhd7H

iCal: OPEN-O VNF-SDK Project Discussion -- Friday.ics

VNF SDK Meetings Minutes

December 1st, 2016: Meeting Notes

Participants: Gigaspace, Intel, Huawei, ZTE

Agenda

- Gigaspaces introduce the details info about VNF-Onboarding(like HW awareness, container,etc)
- Alex/Huabing suggest we should decompose each model into more detail(like microservice level)
- Tal want more guys add comment on gigaspaces blueprint
- We talk about detail of VNF-SDK repo
- Maketplace:
    1. Alex suggest Canonical contribute this?
    2. Alex said maketplace should automatically download package and upload package, meanwhile do the VNF on-boarding

December 14th, 2016: Meeting Notes (TBD)

Agenda:

- We discuss the detail of VNF-SDK
- See detail meeting notes and decision at here

Join from PC, Mac, Linux, iOS or Android: https://huawei.zoom.us/j/817298236

Or iPhone one-tap (US Toll): +14086380968,817298236# or +16465588656,817298236#

Or Telephone:
Dial: +1 408 638 0968 (US Toll) or +1 646 558 8656 (US Toll)
Meeting ID: 817 298 236

International numbers available: https://huawei.zoom.us/zoomconference?m=fPXGPQ61eAZ0QYI4n9jzhbNlh57Ti7kq

January 20, 2017 Meeting Notes

Agenda:

1. Discussion on VNF-SDK Design & Package Tool Huabing/Krzysztof (40min)
2. Sub Project status update Sub-project owner (20min)

February 9, 2017 Meeting Notes

Agenda:

1. VNF Package (CSAR) Spec Discussion Victor Gao(20min)
2. VNF SDK sub-project status update sub PTL

February 9, 2017 VNF SDK Meeting Minutes

January 20, 2017 VNF SDK Meeting Minutes

Participate: CMCC, Intel, Huawei, ZTE

Agenda

1. Huabing give his understanding of Design and package tool
2. Discussion on design&package tool's GUI & CLI
3. Discussion on dry run functionality
4. Discussion on VNF-SDK weekly meeting time
   a. Since VNF-SDK time is too late for China team, we decide have the weekly meeting on China Time morning and PST time morning in turns.

Action item:

1. Alex & Krzysztof update the detail info about package tool(dry run, package validator, package parser)
2. Victor arrange a time to discuss Market place with Alex and Sukesh
3. Victor arrange a time for VNF-SDK weekly meeting on Morning (China Time)

Meeting Materials

File	Modified
Suggestion for some argues in VNF SDK Project(2017.01.24) .pptx Jan 24 TSC topics for VNF-SDK	Jan 24, 2017 by gao weitao

Drag and drop to upload or browse for files

VNF SDK Old Home Page

Project Description

The "VNF Supplier APIs and SDK" (short name: VNF SDK) project delivers code, documentation and best practices for the following functionality:

- Functionality for automated delivery and publication of VNF products with explicit VNFI and VIM dependencies in a supplier/provider neutral and standardized way
- Functionality to enable automated download and CI/CD of VNF products in a supplier/provider neutral and standardized way
- Hooks for automated KPI specification, telemetry generation and VNF performance testing
- Functionality to enable a rich OPEN-O community ecosystem of VNF product suppliers and operators, with high degree of interoperability

This functionality is intended for use by VNF product developers, VNF product DevOps teams, as well as operators of OPEN-O managed infrastructure. As a stretch goal, this project will deliver authoring tools to ease the creation of TOSCA blueprints by VNF developers.

Project Status

Approved on August 31, 2016

Participants

- CMCC
- Intel
- GigaSpaces
- Huawei
- ZTE
- Canonical

Team Meeting Calendar

Team Calendars

VNF SDK Functional Components & Interfaces

The following diagram illustrates the VNF SDK functional components and their public interfaces. Additional information about each project can be found by following the links below.

Functional Components

VNF Product Model/Blueprint

VNF product model/blueprintt provides a declarative way to define deployment, operational and functional attributes of a VNF product. The VNF product is defined in terms of deployment time requirements and dependencies and exposed telemetry indicator definitions.

The deployment time requirements and dependencies define any and all compute infrastructure needs of the VNF product, such as specific hardware architecture, on-chip features, instruction set availability and hypervisor capabilities.

The telemetry indicator definitions define a set of default indicators exposed by a given VNF product for use by monitoring and assurance tools. This list can be extended and customized once a given VNF product is on-boarded and instantiated ar run-time.

The VNF product model is specified using the TOSCA NFV simple profile. It is persisted, along with the product executables and data, using TOSCA CSAR files.

Intel, Gigaspaces

VNF SDK Blueprint

Sub Project List

Format Transform

• **Suggestion**
`Murali: Whether we need to have an GUI to automatically generate VNF/NSD packages similar to OSM ? Is this being considered at this stage of project ? (https://riftio.com/vnf-package-generator/)`

- Contributor/Committer:
  - Dan Kilman - LFID dankilman - GigaSpaces
  - Nir Biran - LFID nirb - GigaSpaces
  - Alex Vul - LFID avul - Intel
  - Marcus Williams - LFID mgkwill - Intel

Validate

• **Suggestion**

- Contributor/Committer:
  - Tal Liron - LFID tal.liron - GigaSpaces
  - Avia Efrat - LFID avia - GigaSpaces
  - Ran Ziv - LFID RanZiv - GigaSpaces
  - Krzysztof Frukacz - LFID - kfrukacz-gs
  - Alex Vul - LFID avul - Intel
  - Marcus Williams - LFID mgkwill - Intel

Lifecycle Test

-

- Contributor/Committer:
  - Maxim Orlov - LFID mxmrlv - GigaSpaces
  - Alex Vul - LFID avul - Intel
  - Marcus Williams - LFID mgkwill - Intel

Function test

• **Suggestion**
`Amir Levy: Suggest to focus on func-test interfaces - and leave the actual func-test to VNF's that specialized`

VNF Product Package Designer

VNF Package Designer, provides VNF product DevOps engineers with a graphical tool to define the VNF product model and package content. It's is made available as part of the VNF Supplier SDK tools.The package designer makes use of the VNF SDK command line interfaces (CLIs) and client-side API language bindings in order to define the model and the package content. As such, it is functionally equivalent to the VNF SDK tools.

VNF SDK Tools

VNF SDK tools provide VNF product DevOps engineers with command line tools and client side API language bindings to define the VNF product model and package content. The following tools are included...

- VNF Package Builder - creates a CSAR file based on inputs provided by the VNF product DevOps engineer
- VNF Package Validator - validates the content of the VNF packages to ensure that everything has been built correctly
- VNF Package Extractor - extracts VNF product model and executables from the CSAR file
- VNF Package Parser - translates VNF proeduct blueprint into a format consumable by OPEN-O components
- VNF Package Dry Run - performs a "dry run" install to ensure that the package can be deployed during instantiation

NF Marketplace

OPEN-O Marketplace Tests

OPEN-O VNF SDK Microservice

OPEN-O NFV-O workflow extensions (in collaboration with the NFV-O project)

Interfaces

Sdk-Md

Exposed by the VNF SDK as commands and client side language bindings. Enables use of VNF SDK functions from graphical user interfaces, shell scripts, programs and command line interfaces. Consumed by the VNF Package Designer.

Sdk-Mp

Exposed by the VNF SDK as commands and client side language bindings. Enables use of NF Marketplace functions from graphical user interfaces, shell scripts, programs and command line interfaces. Consumed by the VNF Package Designer and OPEN-O VNF SDK micro-service adapter.

Mp-Api

Exposed by the NF Marketplace as a programmatic API. Enables use of NF marketplace functions from command and graphical user interfaces. Consumed by the marketplace portal.

Mp-Tf

Exposed by the NF Marketplace as a programmatic API. Enables use of the NF Marketplace's PnP test framework for validation of uploaded VNF products.Consumed by the OPEN-O functional tests.

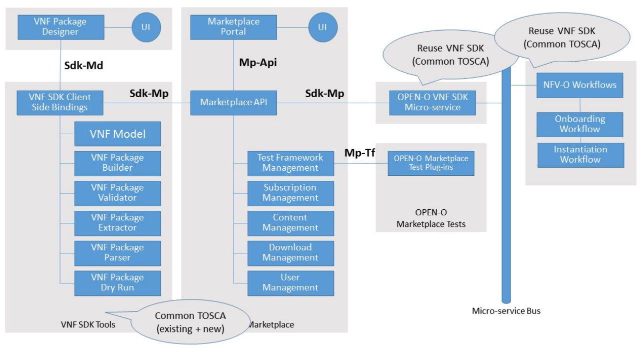
Suggestion

```
Artur Tyloch: For
functest we should
consider ABOT framework.
It is open source VNF
functional testing
framework maintained by
Rebaca https://jujucharm
s.com/u/debayan-ch/abot-
ims-basic/trusty/35
```

- Contributor/Committer:

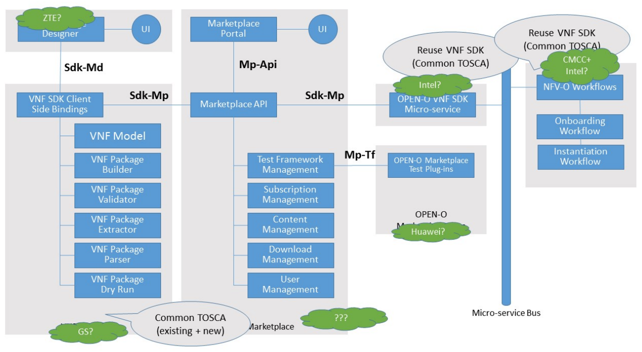
Marketplace

- Suggestion:
- Contributor/committer:



VNF SDK Contributions (PROPOSED)

The following diagram illustrates the proposal for contribution of the VNF SDK components.



Project Plan

Error rendering macro 'viewxls' : The viewfile macro is unable to locate the attachment "VNF-SDK Plan V0.1 .xlsx" on this page

OPEN-O VNF SDK Design(11.15)

Error rendering macro 'viewppt' : The viewfile macro is unable to locate the attachment "OPEN-O_VNF_SDK_Design(11.15).pptx" on this page

Mercury Plan 2016-12-14

Error rendering macro 'viewppt' : The viewfile macro is unable to locate the attachment "OPEN-O VNF-SDK Planning in Mercury Release(2106.12.15).pptx" on this page

VNF Design Tool 2016-1-22

Error rendering macro 'viewppt' : The viewfile macro is unable to locate the attachment "VNF SDK Design&Package Tool.pptx" on this page

Recent space activity

- **gao weitao**
  - Validate & Lifecycle Test updated Apr 28, 2017 • view change

- VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC1) created Apr 13, 2017
- VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC2) updated Apr 13, 2017 • view change
- VNF SDK Mercury Release Deliverables for Release Candidate Milestone (RC0) updated Apr 13, 2017 • view change

- **Gildas Lanilis**
  - VNF SDK Project Home Page updated Mar 17, 2017 • view change

VNF SDK Project Proposal

Project Description

The "VNF Supplier APIs and SDK" (short name: VNF SDK) project delivers code, documentation and best practices for the following functionality:

- Functionality for automated delivery and publication of VNF products with explicit VNFI and VIM dependencies in a supplier/provider neutral and standardized way
- Functionality to enable automated download and CI/CD of VNF products in a supplier/provider neutral and standardized way
- Hooks for automated KPI specification, telemetry generation and VNF performance testing
- Functionality to enable a rich OPEN-O community ecosystem of VNF product suppliers and operators, with high degree of interoperability

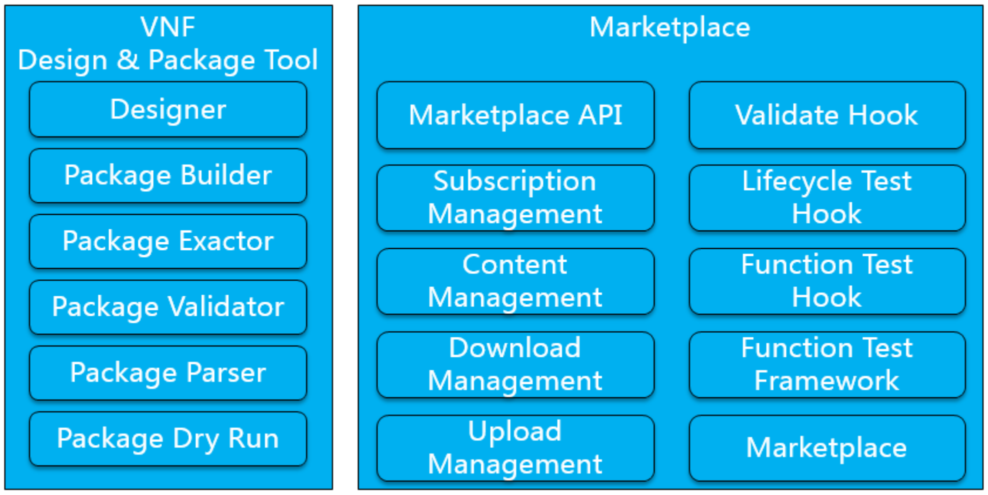
This functionality is intended for use by VNF product developers, VNF product DevOps teams, as well as operators of OPEN-O managed infrastructure. As a stretch goal, this project will deliver authoring tools to ease the creation of TOSCA blueprints by VNF developers.

Project Status

Approved on August 31, 2016

Architecture

ARC Overview



Flow Diagram

You can find flow diagram here.

Open-O-VNF-Supplier-v5

OPEN-O_VNF_Supplier_APIs_&_SDK_Project_Proposal(2016116)

References

VNF SDK References

VNF SDK References

VNF SDK Functional Components & Interfaces

The following diagram illustrates the VNF SDK functional components and their public interfaces. Additional information about each project can be found by following the links below.

Functional Components

VNF Product Model/Blueprint

VNF product model/blueprint provides a declarative way to define deployment, operational and functional attributes of a VNF product. The VNF product is defined in terms of deployment time requirements and dependencies and exposed telemetry indicator definitions.

The deployment time requirements and dependencies define any and all compute infrastructure needs of the VNF product, such as specific hardware architecture, on-chip features, instruction set availability and hypervisor capabilities.

The telemetry indicator definitions define a set of default indicators exposed by a given VNF product for use by monitoring and assurance tools. This list can be extended and customized once a given VNF product is on-boarded and instantiated at run-time.

The VNF product model is specified using the TOSCA NFV simple profile. It is persisted, along with the product executables and data, using TOSCA CSAR files.

VNF Product Package Designer

VNF Package Designer, provides VNF product DevOps engineers with a graphical tool to define the VNF product model and package content. It is made available as part of the VNF Supplier SDK tools.The package designer makes use of the VNF SDK command line interfaces (CLIs) and client-side API language bindings in order to define the model and the package content. As such, it is functionally equivalent to the VNF SDK tools.

VNF SDK Tools

VNF SDK tools provide VNF product DevOps engineers with command line tools and client side API language bindings to define the VNF product model and package content. The following tools are included...

- VNF Package Builder - creates a CSAR file based on inputs provided by the VNF product DevOps engineer
- VNF Package Validator - validates the content of the VNF packages to ensure that everything has been built correctly
- VNF Package Extractor - extracts VNF product model and executables from the CSAR file
- VNF Package Parser - translates VNF proeduct blueprint into a format consumable by OPEN-O components
- VNF Package Dry Run - performs a "dry run" install to ensure that the package can be deployed during instantiation

NF Marketplace

TBD

OPEN-O Marketplace Tests

TBD

OPEN-O VNF SDK Microservice

TBD

OPEN-O NFV-O workflow extensions (in collaboration with the NFV-O project)

TBD

Interfaces

Sdk-Md

Exposed by the VNF SDK as commands and client side language bindings. Enables use of VNF SDK functions from graphical user interfaces, shell scripts, programs and command line interfaces. Consumed by the VNF Package Designer.

Sdk-Mp

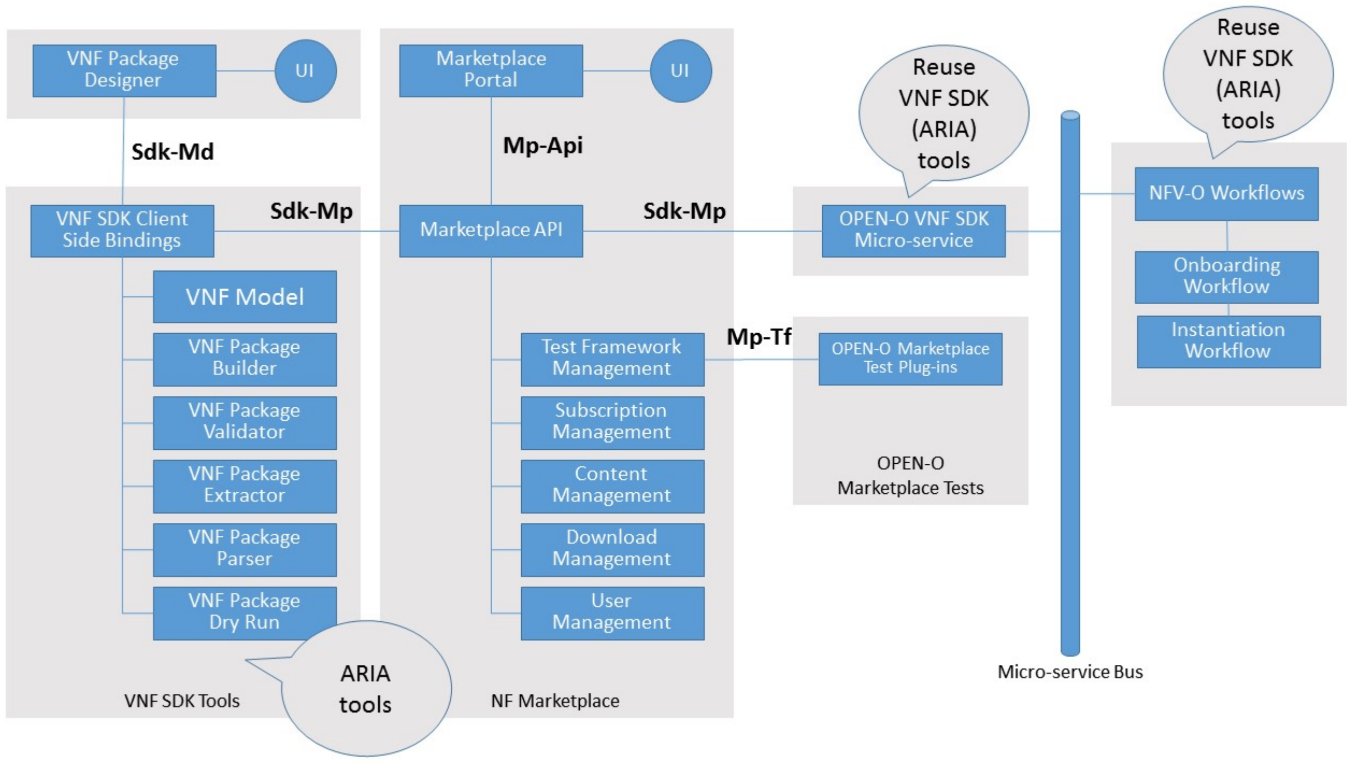
Exposed by the VNF SDK as commands and client side language bindings. Enables use of NF Marketplace functions from graphical user interfaces, shell scripts, programs and command line interfaces. Consumed by the VNF Package Designer and OPEN-O VNF SDK micro-service adapter.

Mp-Api

Exposed by the NF Marketplace as a programmatic API. Enables use of NF marketplace functions from command and graphical user interfaces. Consumed by the marketplace portal.

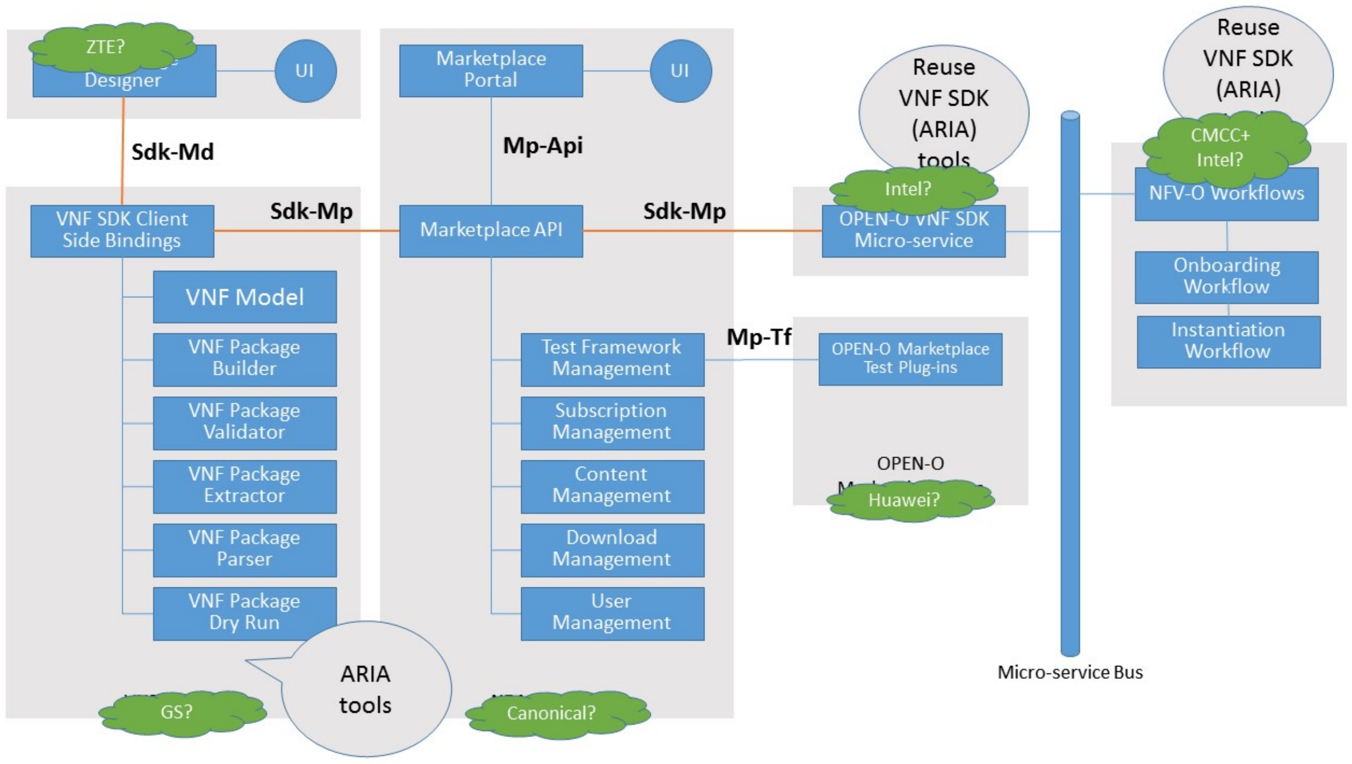
Mp-Tf

Exposed by the NF Marketplace as a programmatic API. Enables use of the NF Marketplace's PnP test framework for validation of uploaded VNF products. Consumed by the OPEN-O functional tests.



VNF SDK Contributions (PROPOSED)

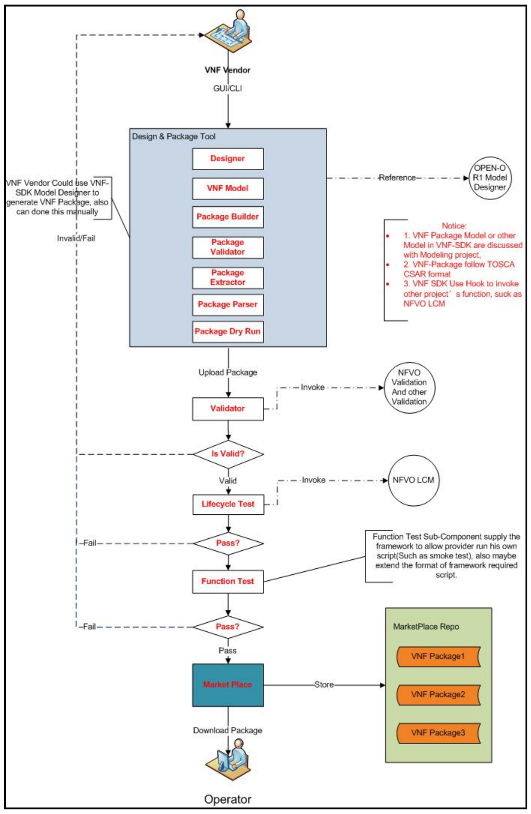
The following diagram illustrates the proposal for contribution of the VNF SDK components.



VNF SDK Sub Projects

VNF SDK Flow Diagram

This is draw by visio 2007, you can find the source file in attachment *.vsd.



VNF Vendor Could use VNF-SDK Model Designer to generate VNF Package, also can done this manually

Notice:
- 1. VNF Package Model or other Model in VNF-SDK are discussed with Modeling project,
- 2. VNF-Package follow TOSCA CSAR format
- 3. VNF SDK Use Hook to invoke other project's function, suck as NFVO LCM

Function Test Sub-Component supply the framework to allow provider run his own script(Such as smoke test), also maybe extend the format of framework required script.

Design&Package Tool

Project structure

- VNF Package Designer
- VNF SDK tools
    - VNF Model: VNF SDK Blueprint
    - VNF Package Builder
    - VNF Package Validator
    - VNF Package Extractor
    - VNF Package Parser
    - VNF Package Dry Run

VNF SDK tools

VNF Model

The model information can be found under the following link: VNF SDK Blueprint

VNF Package tools

Provided tools

VNF Package Builder - creates a CSAR file based on inputs provided by the VNF product DevOps engineer
VNF Package Validator - validates the content of the VNF packages to ensure that everything has been built correctly
VNF Package Extractor - extracts VNF product model and executables from the CSAR file
VNF Package Parser - translates VNF product blueprint into a format consumable by OPEN-O components
VNF Package Dry Run - performs a "dry run" install to ensure that the package can be deployed during instantiation

The tools are provided in a form of a shared library (Python module) that can be used in other projects. A CLI is also provided out-of-the box for DevOps to use the library with their scripts and autoamtion framework.

Repository

Name: vnf-sdk-design-pkg

Clone command: git clone https://gerrit.open-o.org/r/vnf-sdk-design-pkg

Installation

Python module with CLI is installed by Python pip command. It is possible to install into a virtual environment (virtualenv).

The following commands are executed in the cloned repository directory

1. `pip install -r requirements.txt`

   `Install all required dependencies`

2. `pip install .`

   `Install VNF SDK tools package`

Usage

- `Create CSAR by specifying a directory`

  `vnfsdk csar-create -d DESTINATION source entry`

- `Extract CSAR content`

  `vnfsdk csar-open -d DESTINATION source`

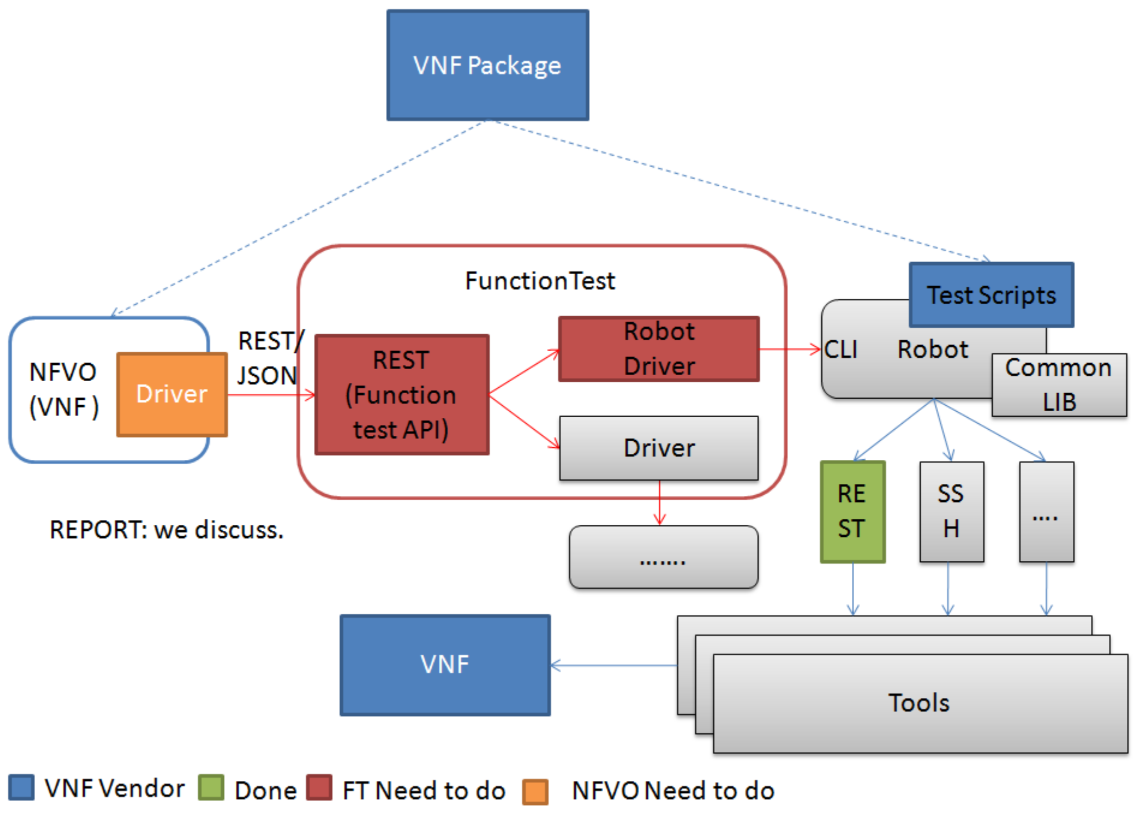
- `Validate CSAR content`

  `vnfsdk csar-validate source`

All commands have -h switch which displays help and description of all paramaters.

Function Test

Abstract Design



Function Test Requirements:

- Support to integrate with different kinds of framework.
- Generate test result with different format.
- Deploy test framework and execute VNF's test scripts automatically
- Collect and display test result.
- Create or extend Robot Framework's test library.

Function Test Interface

Execute Function Test

Interface Definition	Description
URI	/openoapi/vnfsdk/v1/functest
Operation Type	POST
Content-Type	multipart/form-data

Request Parameters:

Attribute	Qualifier	Cardinality	Content	Description
file	M	1	InputStream	The CSAR file stream

Response:

Attribute	Qualifier	Cardinality	Content	Description
functestId	M	1	String	Id of the function test

Get Function Test Result

Interface Definition	Description
URI	/openoapi/vnfsdk/v1/functest
Operation Type	GET
Content-Type	multipart/form-data

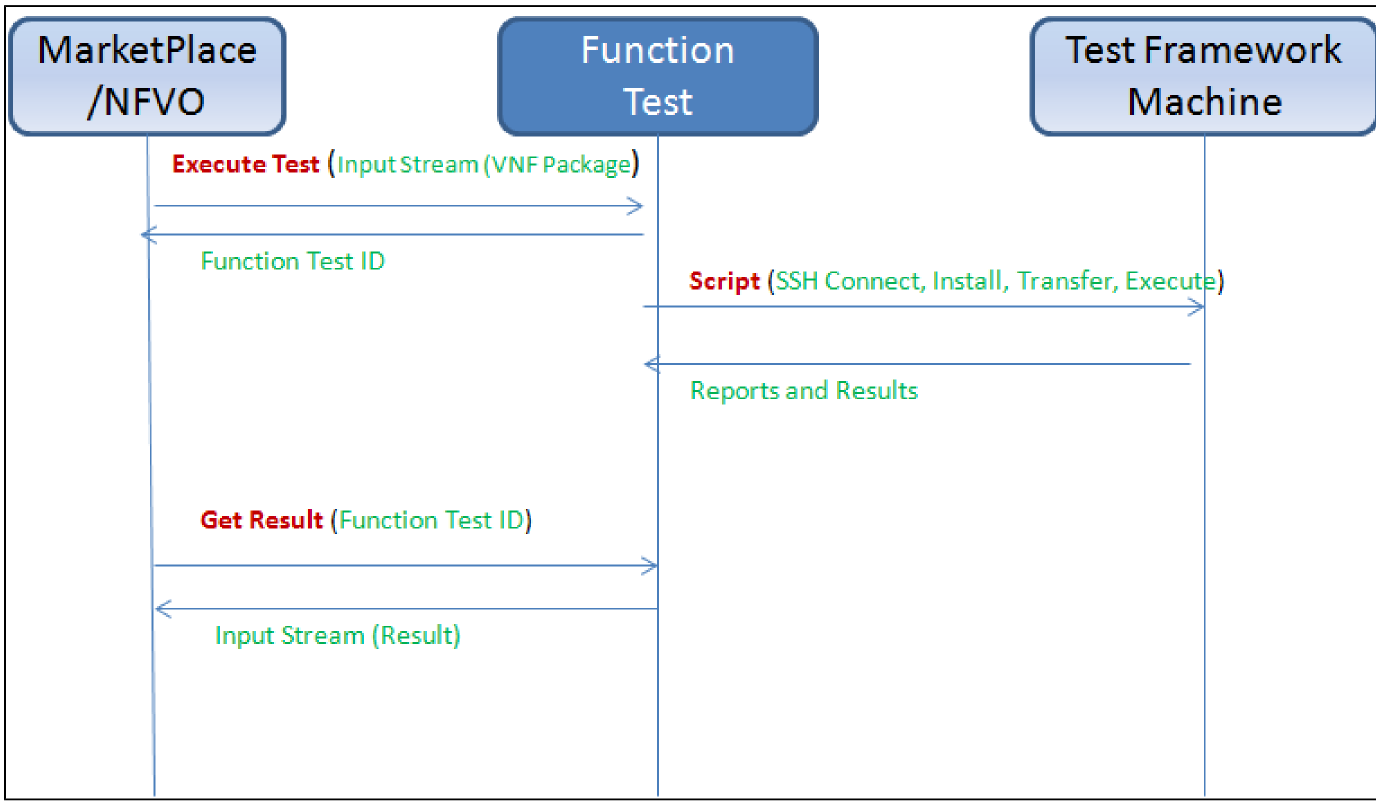
Request Parameters:

Attribute	Qualifier	Cardinality	Content	Description
functestId	M	1	String	Id of function test

Response:

Attribute	Qualifier	Cardinality	Content	Description
result	M	1	Stream	result folder is zipped and sent out as stream

Initial Design Flow


MarketPlace /NFVO
Function Test
Test Framework Machine
Execute Test (Input Stream (VNF Package)
Function Test ID
Script (SSH Connect, Install, Transfer, Execute)
Reports and Results
Get Result (Function Test ID)
Input Stream (Result)

Test Plan

Testing Plan:

1. Test Interface between Market Place and Function Test
2. Execute a simple test and check whether tests are executed in remote machine and results are transferred to local machine.
3. Results must be displayed in the portal

References:

Test Framework Selection

Automatic Test Framework Selection for Current Release.pptx

Enhanced REST Library for Robot framework

Enhanced REST Library for Robot.pptx

Suggestion
Amir Levy: Suggest to focus on func-test interfaces - and leave the actual func-test to VNF's that specialized Artur Tyloch: For functest we should consider ABOT framework. It is open source VNF functional testing framework maintained by Rebaca https://jujucharms.com/u/debayan-ch/abot-ims-basic/trusty/35

Market Place

Market Place API

Upload VNF Package

Interface Definition	Description
URI	/openoapi/marketplace/v1/vnf
Operation Type	POST
Content-Type	multipart/form-data

Request Parameters:

Attribute	Qualifier	Cardinality	Content	Description
file	M	1	InputStream	The CSAR file stream
file	M	1	FormDataContentDisposition	The detail of CSAR file

FormDataContentDisposition

Attribute	Content	Description
type	String	the disposition type. will be "form-data"
name	String	the control name
fileName	String	the file name
creationDate	Date	the creation date
modificationDate	Date	the modification date

readDate	Date	the read date
size	String	the size
parameters	Map<String,String>	the parameters

Response:

Attribute	Qualifier	Cardinality	Content	Description
csarId	M	1	String	The CSAR identifier

Query VNF infomation by csarId

Interface Definition	Description
URI	/openoapi/marketplace/v1/csars/{csarId}
Operation Type	GET

Request Parameters:

Attribute	Qualifier	Cardinality	Content	Description
csarId	M	1	String	The id of CSAR package

Response:
CSARPackage

Sample:

```
{
       "csarId":"78ede6f3-66cc-46ab-b748-38a6c010d272",
       "name":"NanocellGateway",
       "provider":"ZTE",
       "version":"V1.0",
       "createTime":"2016-06-29 03:33:15",
       "modifyTime":"2016-06-29 09:33:15",
       "size":"0.93M",
       "downloadUri":"http://msb_ip:msb_port/files/marketplace-http/CSAR/ZTE/NanocellGW/v1.0/NanocellGatewa
y.csar",
       "type":"CSAR"
   }
```

Download VNF package files download URI

Interface Definition	Description
URI	/openoapi/marketplace/v1/csars/{csarId}/files?relativePath=xxx
Operation Type	GET

Request Parameters:

Attribute	Qualifier	Cardinality	Content	Description
csarId	M	1	String	The id of CSAR
relativePath	M	1	String	The file relative path in CSAR package

Sample:
/openoapi/marketplace/v1/csars/78ede6f3-66cc-46ab-b748-38a6c010d272/files?relativePath="/images/segw.img"

Response:

Attribute	Qualifier	Cardinality	Content	Description
downloadUri	M	1	String	The download URI of file

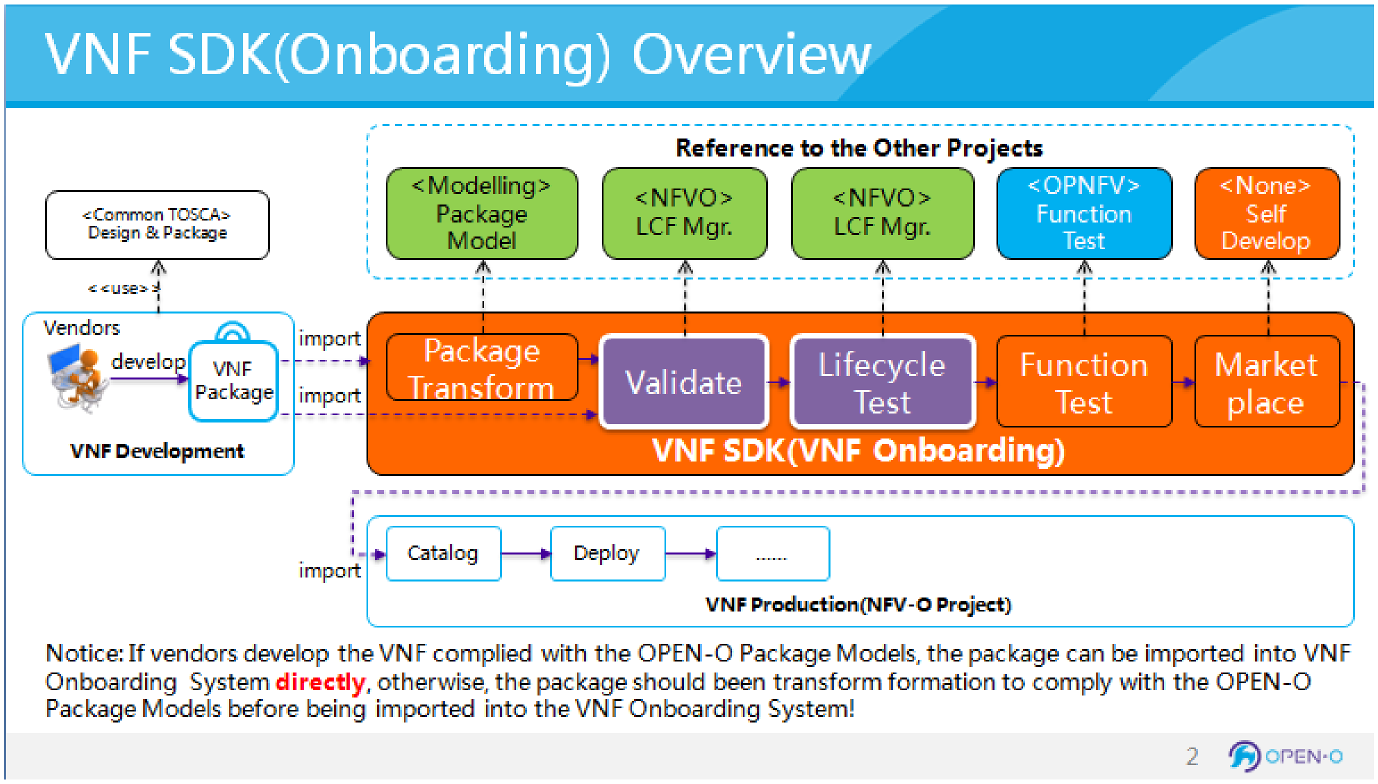
Sample:

```
{
       "downloadUri": "http://msb_ip:msb_port/files/marketplace-http/NSAR/ZTE/NanocellGW/v1.0/NanocellGW/images/segw.img",
       "localPath": "D:\CSAR\ZTE\NanocellGW\v1.0\NanocellGW\images\segw.img"
   }
```

Validate & Lifecycle Test

Background

Validate & Lifecycle Test as one of the parts of VNF SDK project in OPEN-O, there are two main functions in this part, one is validating the package from vendors, and the other is doing the lifecycle test for the package (e.g. instantiation, termination, scaling, etc.). The scope of Validate & Lifecycle Test component is shown in the figure with purple blocks.



VNF SDK(Onboarding) Overview
Reference to the Other Projects
<Modelling> Package Model
<NFVO> LCF Mgr.
<NFVO> LCF Mgr.
<OPNFV> Function Test
<None> Self Develop
<Common TOSCA> Design & Package
<<use>>
Vendors
develop
VNF Package
VNF Development
import
import
Package Transform
Validate
Lifecycle Test
Function Test
Market place
VNF SDK(VNF Onboarding)
import
Catalog
Deploy
......
VNF Production(NFV-O Project)
Notice: If vendors develop the VNF complied with the OPEN-O Package Models, the package can be imported into VNF Onboarding System directly, otherwise, the package should been transform formation to comply with the OPEN-O Package Models before being imported into the VNF Onboarding System!
2
OPEN-O

Overall Design

Validate

In Mercury Release, by reusing the onboarding operation for VNF package in NFVO, the package from vendors can be validated. There are three main functions listed below.

- Format checking
- Integrity checking
- Tamper proofing

Lifecycle Test

In Mercury Release, by reusing the instantiation and termination operation for VNF in NFVO, the lifecycle test for the package can be done. There are two main functions listed below.

- The instantiation of VNF
- The termination of VNF

In other words, in mercury release, NFVO will provide the functional entity for the validation module, and the operational entity for the lifecycle test module.

Interface Design (draft - need to be confirmed)

Refer to NS LCM API Definition of NFVO.

1. here VNF-SDK use NFVO onboarding interface
2. NFVO onboarding interface will cover package Validation and Lifecycle test functionality. VNF-SDK need package the package(or Package Id) to NFVO do onboarding operation

VNF Package LifecycleTest and Validate

Interface Definition	Description	Direction
URI	/openoapi/vnfsdk/v1/vnfpackage	MarketPlace->Lifecycle->NFVO
Method	POST	

Request:

Parameter	Qualifier	Cardinality	Content	Description
Package Id	M	1	String	VNF package id in Marketplace

Response:

Parameter	Qualifier	Cardinality	Content	Description
jobId	M	1	Identifier	The identifier of the VNF package operation occurrence job.

VNF Package Delete

Interface Definition	Description	Direction
URI	/openoapi/vnfsdk/v1/vnfpackage/{csarId}	MarketPlace->Lifecycle->NFVO
Method	DELETE	

Request:

none

Response:

Parameter	Qualifier	Cardinality	Content	Description
jobId	M	1	Identifier	The identifier of the VNF package operation occurrence job.

Update Validation/Lifecycle Test interface

Interface Definition	Description	Direction

URL	/openoapi/vnfsdk-lifecycletest/v1/vnfpackage/updatestatus	NFVO->Lifecycle->Marketplace
Method	POST	

Request:

Parameter	Qualifier	Cardinality	Content	Description
jobId	M	1	Identifier	The identifier of the VNF package operation occurrence job.
Validate_status	M	1	String	Indicate Validation success or not Should be:"Pass" or "Fail"
Lifecycle_status	M	1	String	Indicate Lifecycle Test success or not Should be:"Pass" or "Fail"
Vnf_info	O	1	String	The information about the Package deployed, only valid when Lifecycle_status is "Pass"
-Vms	M	1~n	String	The Virtual Machine information inside the vnf
--ip	M	1	String	The management ip of VM
--username	M	1	String	The username that can access the VM
--Password	M	1	String	The Password that can access the VM

```
{
 "jobId": "1",
 "validate_status": "Pass",
 "lifecycle_status": "Pass",
 vnf_info: {
  "vms": [{
   "ip": "1.1.1.1",
   "username": "open-o",
   "password": "*******"
  },
  {
   "ip": "1.1.1.2",
   "username": "open-o-2",
   "password": "*******"
  }]
 }
}
```

Response:

None

VNF Instantiation

As previous discussion, the VNF Package Onboarding interface includes the process of onboarding and the instantiation of VNF.

VNF Termination

Whether the VNF Package Delete interface including the termination of VNF or not needs to be confirmed.

Validate

NFVO onboarding interface will cover validate functionality

Resource

Lifecycle Resource
Contributor/Committer:

- Tal Liron - LFID tal.liron - GigaSpaces
- Avia Efrat - LFID avia - GigaSpaces
- Ran Ziv - LFID RanZiv - GigaSpaces
- Krzysztof Frukacz - LFID - kfrukacz-gs
- Alex Vul - LFID avul - Intel
- Marcus Williams - LFID mgkwill - Intel

Lifecycle Test Contributor/Committer:

- Maxim Orlov - LFID mxmrlv - GigaSpaces
- Alex Vul - LFID avul - Intel
- Marcus Williams - LFID mgkwill - Intel

VNF SDK Testing

- VNF SDK Function Test CSIT use cases
- VNF SDK Marketplace CSIT use cases
- VNF SDK package tool CSIT use cases
- VNF SDK Validation / Lifecycle Test CSIT use cases

VNF SDK Function Test CSIT use cases

Test Plan

Testing Plan:

1. Use Robot Framework scripts to test each REST Interface.
2. All Test cases must be running in Jenkins.

Sample Test Script Link:

Sample CSIT cases

Use Case 1 – Done (Running in Jenkins)

Basic Health Check for the service including MSB registration

Use Case 2 - Done (Running in Jenkins)

Upload the CSAR package to VNF Function test with **Firewall VNF** test cases. Result should return the ***funcTestID.***

Use Case 3 - Done (Running in Jenkins)

Get the function test result using ***funcTestID***. Result should return result zip file in Stream format.

VNF Vendor Script

Firewall function test use case.

a. Ping remote machine
b. Connect to remote machine
c. execute firewall command to stop accepting icmp packets from test machine
d. Ping remote machine and it should fail

e. execute firewall command to remove the above firewall command to block icmp ping.
f. Ping remote machine and it should pass now.

VNF SDK Marketplace CSIT use cases

Use Case 1 - Done (Running

in Jenkins)

query and publish CSAR list in UI

Use Case 2 - Done (Running in Jenkins)

upload CSAR to marketplace repository which includes validation, life cycle test and functional test

Use Case 3 - Done

download CSAR from marketplace repository

VNF SDK package tool CSIT use cases

Use Case 1

Package a properly structured files into CSAR

Preconditions

A directory that contains a proper YAML file. Additional files may be provided

Tests steps

1. call csar-create procedure to create a temporary CSAR file from input directory
2. check for errors
3. check if the file is not empty

Expected result

A non-empty CSAR file is created

Use Case 2

Validate a properly structured CSAR

Preconditions

A reference CSAR file provided for the test case

Tests steps

1. call csar-validate procedure using the provided CSAR file
2. check for errors

Expected result

Operation ends with no errors

Use Case 3

Extract proper CSAR and confirm consistency with original files

Preconditions

A reference CSAR file and reference files to compare with the extracted ones

Tests steps

1. extract the reference CSAR file to a temporary directory
2. compare the extracted files with the reference ones (they should be identical)

Expected result

Extracted files are identical as the reference files

Use Case 4

Package invalid structure into CSAR - Attempt should end with error infomration

Preconditions

A malformed CSAR file is provided

Tests steps

1. attempt to extract the malformed CSAR file

Expected result

Operation ends with error

VNF SDK Validation / Lifecycle Test CSIT use cases

Use Case 1 - Done (Running in Jenkins)

Call VNF Package Onboarding interface of NFVO to validate the VNF package.

Use Case 2 - Done (Running in Jenkins)

Call Instantiate VNF interface of NFVO to instatiate VNF.

Use Case 3 - Done

Call Terminate VNF interface of NFVO to terminate VNF.