

VF-C Honolulu M1 Release Planning

- 1 Overview
- 2 Scope
 - 2.1 What is this release trying to address?
 - 2.2 Use Cases
 - 2.3 Minimum Viable Product
 - 2.4 Functionalities
 - 2.4.1 Epics
 - 2.4.2 Stories
 - 2.5 Longer term roadmap
- 3 Release Deliverables
- 4 Sub-Components
- 5 Architecture
 - 5.1 High level architecture diagram
 - 5.2 Platform Maturity
 - 5.3 API Incoming Dependencies
 - 5.4 API Outgoing Dependencies
 - 5.5 Third Party Products Dependencies
- 6 Testing and Integration Plans
- 7 Gaps
- 8 Known Defects and Issues
- 9 Risks
- 10 Resources
- 11 Release Milestone
- 12 Team Internal Milestone
- 13 Documentation, Training
- 14 Other Information
 - 14.1 Vendor Neutral
 - 14.2 Free and Open Source Software

Overview

Project Name	Enter the name of the project
Target Release Name	Honolulu
Project Lifecycle State	Mature
Participating Company	China Mobile ,ZTE, Huawei, Nokia, Intel

Scope

What is this release trying to address?

Describe the problem being solved by this release

1. Improve platform maturity (TSC Muse have)
2. Fix the bugs left over from the previous version

Use Cases

Describe the use case this release is targeted for (better if reference to customer use case).

No new use case support planned for Honolulu release.

Minimum Viable Product

Describe the MVP for this release.

VF-C will include the necessary sub-components supporting the primary objectives: meeting platform maturity goals and supporting the use cases.

LCM(instantiate/terminate/heal/scaling) for NS and Vendor VNFs

FCAPS for vendor VNFs

LCM(instantiate/terminate) for open source VNFs








Minimum VF-C components supporting above functionalities:

NSLCM/GVNFM/Workflow/vendor VNF driver

Functionalities

List the functionalities that this release is committing to deliver by providing a link to JIRA Epics and Stories. In the JIRA Priority field, specify the priority (either High, Medium, Low). The priority will be used in case de-scoping is required. Don't assign High priority to all functionalities.

Epics

Key	Summary	T	Created	Updated	Due	Assignee	Reporter	P	Status	Resolution
VFC-1838	Release Candidate 2 Integration and Test		Apr 09, 2021	May 07, 2021	Apr 22, 2021	Unassigned	None	==	CLOSED	Done
VFC-1832	Release Candidate 1 Integration and Test		Mar 17, 2021	Apr 15, 2021	Mar 25, 2021	Unassigned	None	==	CLOSED	Done
VFC-1818	Release Candidate 0 Integration and Test		Feb 23, 2021	Apr 15, 2021	Mar 11, 2021	Unassigned	None	==	CLOSED	Done
VFC-1791	Feature Freeze		Feb 01, 2021	Mar 16, 2021	Feb 25, 2021	Unassigned	None	==	CLOSED	Done
VFC-1778	Specification Freeze		Dec 23, 2020	Feb 05, 2021	Jan 21, 2021	Unassigned	None	==	CLOSED	Done
VFC-1773	Release Planning		Dec 12, 2020	Feb 05, 2021	Jan 11, 2021	Unassigned	None	==	CLOSED	Done
VFC-1769	This epic covers the work to satisfy TSC Must-Have Items for Honolulu Release		Nov 25, 2020	Mar 22, 2021		Unassigned	None	==	CLOSED	Done

[7 issues](#)

Stories

Key	Summary	T	Created	Updated	Due	Assignee	Reporter	P	Status	Resolution
VFC-1805	Remove dependency on onaplogging		Feb 03, 2021	Mar 03, 2021		Unassigned	None	==	CLOSED	Done
VFC-1787	Update VFC document		Jan 26, 2021	Mar 22, 2021		Unassigned	None	==	CLOSED	Done
VFC-1770	Remove components that are no longer used or maintained		Nov 27, 2020	Feb 09, 2021		Unassigned	None	==	CLOSED	Done

[3 issues](#)

Longer term roadmap

Indicate at a high level the longer term roadmap. This is to put things into the big perspective.

Release Deliverables

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

Deliverable Name	Deliverable Description
Source Code	Source code for all VF-C components
Maven Artifacts	Maven Artifacts for all VF-C components
Docker Containers	Docker container associated with VF-C components
Documentation	VF-C detailed documentation

Sub-Components

List all sub-components part of this release.
Activities related to sub-components must be in sync with the overall release.

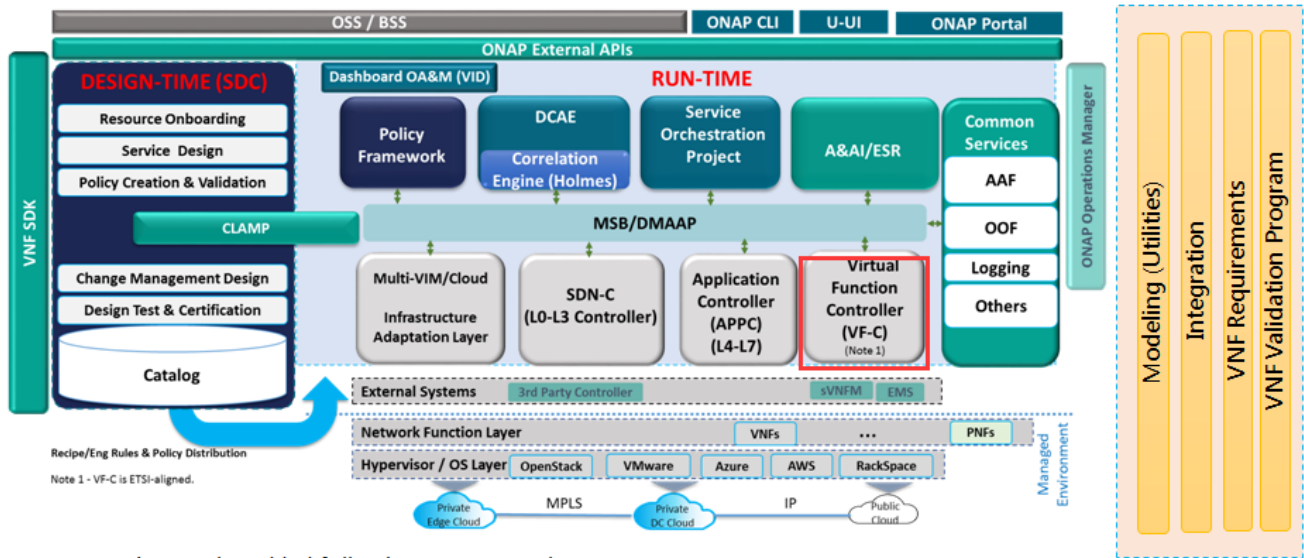
Please see the INFO.yaml files associated with each repo as the authoritative sources of information. <https://gerrit.onap.org/r/admin/repos/q/filter:vf-c>

Architecture

High level architecture diagram

At that stage within the Release, the team is expected to provide more Architecture details describing how the functional modules are interacting.

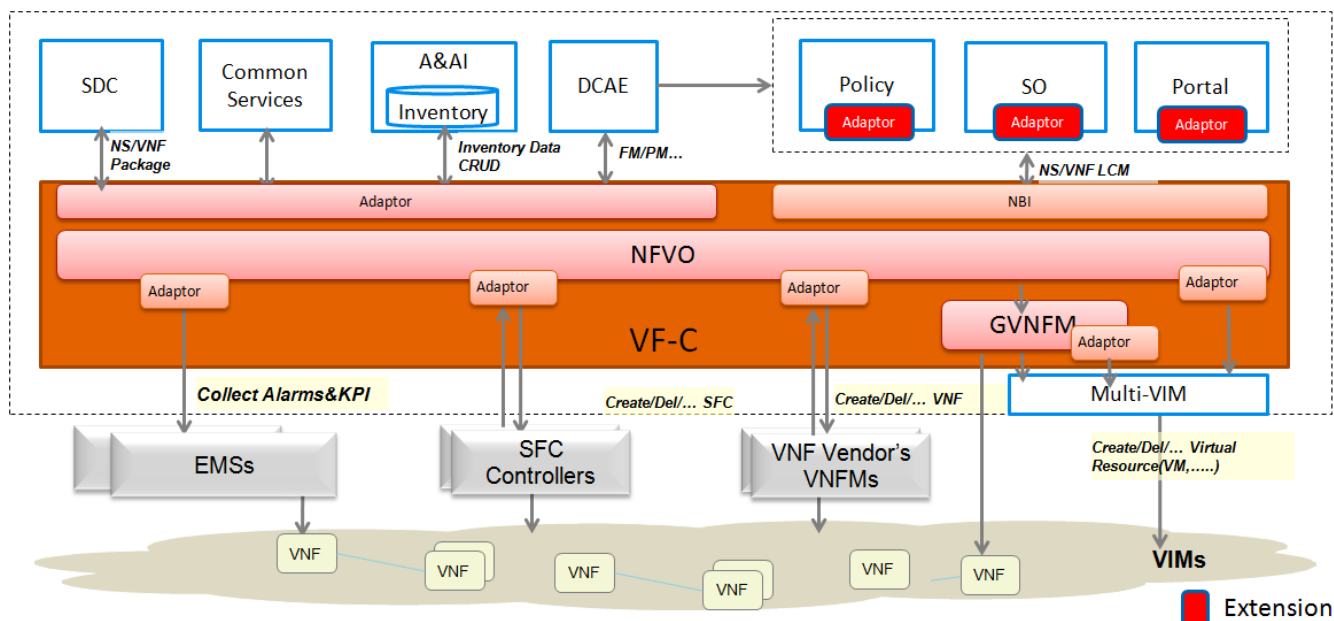
Indicate where your project fit within the [ONAP Architecture diagram](#).



New projects to be added following TSC approval

Block and sequence diagrams showing relation within the project as well as relation with external components are expected.

Anyone reading this section should have a good understanding of all the interacting modules.



Platform Maturity

Please fill out the centralized wiki page: [Honolulu Release Platform Maturity](#)

• API Incoming Dependencies

List the API this project is expecting from other projects.

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 API Freeze date](#).

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

API Name	API Description	API Definition Date	API Delivery date	API Definition link (i.e.swagger)
Parser API(Modeling)	API for parsing TOSCA file			Etsicatalog API Document
Micro-services bus API	API for registration and use of micro-services bus			Microservice Bus API Documentation
Multi-vim API	API to allocate resource to VIM			MultiCloud API
DCAE VES collector SB API	API to push vnf fcaps data to VES collector			DCAE API Document
A&AI API	API to store inventory			AAI REST API Documentation
OOF API	API to chose VIM which is used to deploy VNF			OOF API Document

• API Outgoing Dependencies

API this project is delivering to other projects.

API Name	API Description	API Definition Date	API Delivery date	API Definition link (i.e.swagger)
NSLCM API Definition	Network services lifecycle management APIs			https://docs.onap.org/projects/onap-vfc-nfvo-lcm/en/latest/platform/APIs/NSLCM_API/index.html
VNFM Driver API Definition	VNFM Driver component northbound APIs			https://docs.onap.org/projects/onap-vfc-nfvo-lcm/en/latest/platform/APIs/VNFMDriver_API/index.html
VNF LCM API Definition	provide VNF lifecycle management APIs			https://docs.onap.org/projects/onap-vfc-nfvo-lcm/en/latest/platform/APIs/VNFLCM_API/index.html

• Third Party Products Dependencies

Third Party Products mean products that are mandatory to provide services for your components. Development of new functionality in third party product may or not be expected.

List the Third Party Products (OpenStack, ODL, RabbitMQ, ElasticSearch, Crystal Reports, ...).

Name	Description	Version
Django	https://www.djangoproject.com/	2.1.10
djangoestframework	https://www.django-rest-framework.org/	3.10.3

In case there are specific dependencies (Centos 7 vs Ubuntu 16. Etc.) list them as well.

• Testing and Integration Plans

Provide a description of the testing activities (unit test, functional test, automation,...) that will be performed by the team within the scope of this release.

Describe the plan to integrate and test the release deliverables within the overall ONAP system.

Confirm that resources have been allocated to perform such activities.

For Unit Test, all components are required to maintain 55% codecoverage at the mininum.

<https://sonarcloud.io/organizations/onap/projects?search=vfc>

Functional test plan

[VF-C Honolulu Functional Test Plan](#)

• 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 a future Release.

List identified release gaps (if any), and its impact.

Gaps identified	Impact
To fill out	To fill out

• Known Defects and Issues

N/A

• 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).

Please update any risk on the centralized wiki page - [Honolulu Risks](#)

• Resources

Please see the INFO.yaml files associated with each repo as the authoritative sources of information. <https://gerrit.onap.org/r/admin/repos/q/filter:vfc>

• Release Milestone

The milestones are defined at the [Release Planning: Honolulu](#) and all the supporting project agreed to comply with these dates.

• Team Internal Milestone

This section is optional and may be used to document internal milestones within a project team or multiple project teams. For instance, in the case the team has made agreement with other team to deliver some artifacts on a certain date that are not in the release milestone, it is erecommended to provide these agreements and dates in this section.

It is not expected to have a detailed project plan.

Date	Project	Deliverable
To fill out	To fill out	To fill out

• Documentation, Training

Please update the following centralized wiki: [Honolulu Documentation](#)

That includes

- Team contributions to the specific document related to the project (Config guide, installation guide...).
- Team contributions to the overall Release Documentation and training asset
- 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

The Documentation project will provide the Documentation Tool Chain to edit, configure, store and publish all Documentation asset.

Other Information

• 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 ONAP 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 ONAP 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 project table available at [Project FOSS](#).

Charter Compliance

The project team comply with the [ONAP Charter](#).