Part1 VF-C Casablanca Highlights
VF-C Highlights

- **NS Orchestration Supports PNF**
  - NSLCM supports NSD, composed of VNF, PNF, and VL
  - Catalog supports SOL001 PNFD and NSD DM

- **SOL003 Alignment**
  - GVNFNM support SOL003 API
  - Catalog support SOL003 related API

- **Standalone DB Microservice**

- **Hardware Platform Awareness (HPA) Support**
  - Integrate with OOF, and VF-C sending HPA homing requests to OOF and OOF return homing decision to VF-C.
  - VF-C can parse R2+ TOSCA Model which includes HPA feature.

Note1: components listed here don’t contain all VF-C components, only used to highlight Casablanca work.
Mixed Orchestration of VNF and PNF

NS Orchestration Supports PNF

- NSLCM supports NSD, composed of VNF, PNF, and VL
- Catalog supports PNFD and updates NSD DM parse
**Mixed Orchestration of VNF and PNF**

5G use case SO – VFC – EMS workflow

A: Model design & dispatch

B: EMS deploys PNF

C: PNF register to EM
EMS register to VFC and put into A&AI

D: UUI config PNFD parameter

E: UUI invoke 5G NS via VFC

VFC decomposes the NS
VFC invokes VNFM to create VNF
VFC invokes EM to init config PNF
VFC invokes Multivim to create VL

F: PNF report alarm/performance to ONAP
Split existing DB in each component into one standalone DB microservice.
VF-C DB Summary

```
-D MYSQL_ADDR
-docker run
-vfc-nslcm docker
-Django
-vfc-catalog docker
-Django
-vfc-vnflcm docker
-Django

settings.py

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': DB_NAME,
        'HOST': DB_IP,
        'PORT': DB_PORT,
        'USER': DB_USER,
        'PASSWORD': DB_PASSWD,
    },
}

redisco.connection_setup(host=REDIS_HOST, port=REDIS_PORT, password=REDIS_PASSWD, db=0)

```

start redis-server
start mysql service
create database

docker entrypoint
Homing Decision Between VF-C and OOF

1. VFC sends out homing request to OOF(OSDF) containing resource info
2. OOF(OSDF) pulls all the related homing constraints from Policy
3. OOF(HAS) check AAI database to pull region(flavor) information
4. OOF(HAS) communicate with Multi-cloud to check cloud capacity (vims which fulfill the requirements)
5. OOF(OSDF) returns homing allocation solution to VFC
SOL003 Alignment

GVNFM Driver

Catalog

GVNFM

NFVO

NS Package (NSD)

VNF Package ( VNFD)

VIM

Simple VNF

Opensource VNF

Opensource VNF

VNF Package

Catalog support SOL003 related API, including Upload/Download VNF Package content Create/Delete/Get VNF Package

• GVNFM support SOL003 API, including Create/Instantiate/Operate/Terminate/Delete/Query VNF Instance VNF LCM Operation subscriptions and notification.
NSLCM Supporting via UUI

VNF/NS package Management

NSLCM Management
Part2 VF-C Dublin Planning
ETSI Alignment Requirements

1. NFVO directly exposes SOL005 interface to OSS/BSS
2. NFVO is connected to SVNFM or GVNFM through SOL003 interface.
VF-C workflow has different options, the build-in workflow has higher execution efficiency and VF-C also provides workflow microservice that can integrate with different workflow engines. This design allows VF-C to be decoupled from specific workflow engines and more flexible.

The built-in workflow is executed by default in current VF-C code.

In Dublin release, we would like to perform different workflow branches through flexible configuration. And increase the capabilities that workflow microservices can provide, such as monitoring ...
DB HA Support

Continue to separate the database
• independent redis
• mysql migrate to the mariadb Galera Cluster

Mysql Port 3306
Redis Port 6379

Leverage OOM shared MariaDB Galera Cluster Charts
vCPE deployment and termination is managed by ONAP VF-C
1. A&AI will extend service instance to support NS instance information in Dublin release
2. VF-C need to sync the NS information to A&AI service Instance

<table>
<thead>
<tr>
<th>VF-C Inventory</th>
<th>A&amp;AI Inventory</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS</td>
<td>Service</td>
</tr>
<tr>
<td>VNF</td>
<td>Generic-VNF</td>
</tr>
<tr>
<td>Vm</td>
<td>vServer</td>
</tr>
<tr>
<td>Network</td>
<td>L3-network</td>
</tr>
<tr>
<td>......</td>
<td>........</td>
</tr>
</tbody>
</table>
VF-C is participating related analysis and discussion about generic Tosca parser.

<table>
<thead>
<tr>
<th>Project</th>
<th>Implementation language</th>
<th>Parser used</th>
</tr>
</thead>
<tbody>
<tr>
<td>SO</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>VF-C</td>
<td>Python</td>
<td>NFV Tosca Parser</td>
</tr>
<tr>
<td>UUI</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>VNFSDK</td>
<td>Java/Python</td>
<td>NFV Tosca Parser/SDC Tosca Parser</td>
</tr>
<tr>
<td>Policy</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>A&amp;AI</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>SDNC</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>VID</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>SDC</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>CLAMP</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
<tr>
<td>APPC</td>
<td>Java</td>
<td>SDC Tosca Parser</td>
</tr>
</tbody>
</table>

Will continue to participate the follow discussion:
• Parser API requirements collection
• Parser API Design
Current CI/CD with Daily Summary include functional tests + health checks, the functional tests such as:

a. ONAP CI.Vnf-Orchestration.Instantiate Virtual DNS
b. ONAP CI.Vnf-Orchestration.Instantiate Virtual Firewall CL

All current functional tests only cover heat-based VNF deployment, will plan to add tosca-based VNF deployment functional tests.
Thanks