Open Baton
The Open Source Network Function Virtualization Orchestrator (NFVO)

Giuseppe Carella, giuseppe.carella@fokus.fraunhofer.de
Internet: http://openbaton.org
Contact: info@openbaton.org
@openbaton
What is Open Baton?

Open Baton is an Open Source implementation of the ETSI MANO specification

- Open Baton aims to foster, within the NFV framework, the integration between the:
  - Virtual Network Function providers
  - Cloud Infrastructure providers
- Functionality:
  - Installation, deployment and configuration of network services
  - Runs on top of multi-site OpenStack
  - Provides independent infrastructure slices
  - Support for generic or specific VNF management
  - Runtime operations: fault management, autoscaling, etc.
  - A large amount of virtualization use cases e.g. core networks, M2M and Multimedia communication
- Designed for answering R&D requirements
  - Easy to configure and to deploy
  - Providing a centralized view of the testbed

- Available on github: [https://github.com/openbaton](https://github.com/openbaton)
What Open Baton stands for

- Built from scratch following the ETSI MANO specification.
  - The NFVO uses the ETSI NFV data model internally for the definition of the Network Service and Virtual Network Descriptors.

- Allows interoperability
  - Being interoperable is one of the challenges brought by the fragmented ecosystem in the management and orchestration area. It requires a lot of work to make two different vendors solution working together → need of a single vendor-independent platform

- Easily extensible
  - Based on a message bus architecture it provides several mechanisms for being extended with new functionalities and integrated in existing platforms
The reference implementation of the ETSI NFV MANO specification

- Open Baton is based on the ETSI NFV MANO v1.1.1 (2014-12) (*) specification. It provides
  - A **NFV Orchestrator** managing the lifecycle of Network Service Descriptors (NSD) and interfacing with one or more VNF Manager(s) (VNFM)
  - A **generic VNFM**, which can be easily extended for supporting different type of VNFs
  - An **Autoscaling Engine** which can be used for automatic runtime management of the VNFs
  - A **Fault Management System** for automatic management of faults
  - A **set of libraries** which could be used for building your own VNFMs (vnfm-sdk)
  - A **dashboard** for easily managing all the VNFs
- It integrates with OpenStack as main VIM implementation

(*) [http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf](http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf)
Each component run as a single Spring boot application

RabbitMQ used as Message Queue implementation

JClouds for the VIM Driver
Extensibility

Three main mechanisms

1. Plugins based on Remote Procedure Calls (RPC)
   - VIM Drivers
   - Monitoring Drivers

2. VNFMs pluggability
   - Messages using AMQP
   - REST interface between NFVO-VNFM

3. Events
   - NFVO generates events for every lifecycle event
OpenBaton is the missing piece of the larger virtualization ecosystem

- OpenBaton was designed to interact with multiple VIMs
  - Currently OpenStack is supported

- OpenBaton extends the basic orchestration towards network functions management
  - Includes a generic VNFM and a generic EMS
  - Can interoperate with other VNFMs

- Enables the deployment of multiple customized network slices

- OpenBaton environment includes multiple data centers
  - Allocating resources on top of multiple OpenStack installations
OpenBaton - Orchestrator

The orchestrator implements the key functionalities of the MANO architecture

- Currently uses the OpenStack as first integrated NFV PoP VIM
- Maintains an overview on the infrastructure, supporting dynamic registration of NFV PoPs
- Receives virtual network function packages from the different users including VNF images and virtual network functions descriptors (VNFDs)
- Deploys on-demand the VNFs on top of an infrastructure consisting of multiple data center instances (NFV PoPs)
- Deploys in parallel multiple slices one for each tenant, consisting of one or multiple VNFs

© Fraunhofer FOKUS
The Generic VNFM (together with the Generic EMS) has the following NFV functionality:

- Request to the NFVO the allocation of specific resources
  - for the virtual network instance

- Can request from the NFVO the:
  - instantiation,
  - modification,
  - starting and stopping
  of the virtual services (or directly to the VIM)

- Instructs the generic OpenBaton EMS to save and to execute specific configuration scripts within the virtual machine instances
Integrate Your Own VNF Using The Generic VNFM

Make use of the EMS and Generic VNFM for integrating your own VNF

Do you have a VNF, and would like to orchestrate it using OpenBaton without implementing your own VNFM?

− All you need to do is to implement your installation scripts and build your VNF Package:
  − Configure the OpenBaton EMS
  − Configure the Generic VNFM
  − Provide the Virtual Network Function Descriptor (VNFD)
  − Provide the virtual machine images
OpenBaton provides a set of libraries for integrating new network services

Do you have a VNF, and would like to orchestrate it using OpenBaton?

- All you need to do is to implement your installation scripts and a VNFM. The NFVO supports two different mechanisms for interacting with your VNFM.
  - We provide a Java SDK (soon to be extended with a python version) which can help you on building your VNFM
Albeit, OpenBaton offers a generic VNFM which can be easily extended for the different services, it includes also a set of mechanisms which enable the support for external VNFM:

- The interface between VNFM and NFVO provides mechanisms for instantiating, modifying, terminating a VNF
  - A granting mechanism allows the VNFM to either directly instantiate the virtual resources or request their instantiation to the NFVO

- The NFVO provides two different mechanisms for interacting with VNFM:
  - Publish/Subscribe mechanism
    - Over a JMS message queue
  - RESTful API
Soon to come a VNF catalogues where you can choose existing open source VNFs for interoperability testing

In case you are developing a specific VNF and would like to integrate in a Network Service
- Make use of the open catalogue for downloading and using existing VNFs
- Create your own instance independent of the catalogue
- Adapt your own VNF to the existing VNFs
- Contribute to the community with your own VNFs.
OpenBaton includes a user-friendly dashboard which enables the management of the complete environment

- The control of the infrastructure
  - enabling the easy understanding and modification of the dynamic registered NFV PoPs;

- The management of the deployed network services
  - their creation (install, deploy, configure)
  - the overview of the deployed services.
OpenStack as the standard de-facto implementation of the VIM:

- Creation of virtual networks based on the requirements provided by the VNFDs
- Creation of virtual machines which are used for hosting the VNFCs
- NFVO uses the quota information provided by VIM for reserving the resources required by each NS
Monitoring integration

- Using Zabbix as monitoring system for monitoring:
  - NFVI components
  - VNF components

- The plugin offers an interface compliant with the draft IFA005_Or-Vi_ref_point_Spec [1] for retrieving monitoring information

[1] https://docbox.etsi.org/isg/nfv/open/Drafts/IFA005_Or-Vi_ref_point_Spec/
- The Autoscaling Engine dynamically scales a NSR based on policies contained in the NSD.

- It interacts with the monitoring system for retrieving real time data.
It registers to the event INSTANTIATION_FINISHED generated by the NFVO at the end of the NSR deployment

- It retrieves the policy contained in the NSD

- It starts a task checking whether the conditions are met or not

- It uses different algorithms for optimising the instantiation of new VNFCs

- It can integrate with different monitoring systems using the plugin mechanism
Advanced scaling out mechanism

The IDLE groups contain VNF instances instantiated but not configured.

VNF instances are already ACTIVE and under control of the autoscaling engine.

The diagram illustrates the components of the MANO (Multi-Access Edge Computing) architecture, including:

- **NFVO** (Network Function Virtualization Orchestration)
- **AE System** (Application Environment System)
- **Message Queue**
- **VIM** (Virtual Infrastructure Manager)
- **Generic VNFM** (Virtual Network Function Manager)
- **Open Baton**
- **Monitoring Driver**
- **VIM Driver**

The layers are divided into:

- **Virtualisation layer**
- **Physical layer**
- **Monitoring layer** (ZABBIX)

The interaction between these components is depicted through arrows, showing the flow of information and control within the MANO framework.
The Fault Management System detects faults occurring at the NFVI/VNF level and executes recovery actions for recovering the Network Service.
Fault Management System

- It registers to the event INSTANTIATION_FINISHED generated by the NFVO at the end of the NSR deployment

- It retrieves the policy contained in the NSD

- It creates rules for detecting faults on the monitoring system and registers triggers

- Every time a trigger is received, it executes an action which can be (depending on the level of criticality):
  - Heal the VNF
  - Switch to standby the VNF

```
"fault_management_policy": [
  {
    "name": "web server not available", "isVNFAAlarm": true,
    "criteria": [
      {
        "parameter_ref": "net.tcp.listen[80]",
        "function": "last()",
        "vnfc_selector": "at_least_one",
        "comparison_operator": "=",
        "threshold": "0"
      }
    ],
    "period": 5,
    "severity": "CRITICAL"
  }
]
```
Network Slicing Engine (NSE)

- The NSE instantiates rules on physical networks for allocating dedicated bandwidth as per Network Service specific requirements.
Multi PoP NFVI – inter-datacenter networking

- The NSE provides an abstracted view of the inter-datacenter networking topology allowing the instantiation of guaranteed bandwidth levels on top of the physical network elements.
Using the docker image containing the EMS software, it is possible to instantiate a VNF on top of a docker container using the GenericVNFM and VNF Package approach.

The docker image can be stored in the glance repository and selected on the VNFD.

The compute node need to use the nova-docker driver.
Projects, Roles, Users,

- Possibility of defining different projects:
  - Separation of resources, very important for resource orchestrations
- Possibility of registering users and assign them different roles:
  - Admins, services, users, developers
- Possibility of registering users and assign them to different projects:
  - Assign user X to project Y
Soon to become open: VNF Marketplace
Continuous Integration

Travis-ci (public)
- Builds after commits
- Unit Tests

Jenkins (in our premises)
- Integration tests with complex scenarios
- Nightly builds
Apart from adapting to the newest version of the ETSI MANO specification, the following items are planned:

November 2014: OpenSDNCore Rel. 2
Support for multi-data center environments

October 2015: Release 1
- ETSI MANO Specification
- Generic VNFM/EMS

April 2016: Release 2
- Auto-scaling
- Fault management
- Monitoring integration
- TOSCA
- Support of Docker-based EMS

October 2016: Release 3
- Alignment with new ETSI MANO specification drafts/releases
- Basic integration with Machine Learning
- Integration with OPNFV
- Many new features

The timeline and the features are indicative and does not represent a contractual obligation (we are always at least a bit too optimistic)
For further information, technical questions, research information and project requests, contact us at info@openbaton.org

Website: http://openbaton.org
Download at: https://github.com/openbaton