Federation, programmability and security in future NFV/SDN infrastructures

Roberto Minerva, EIT Digital, Italy
Giuseppe Carella, TUB, Germany
Simone Pizzimenti, Reply Communication Valley, Italy

About the speakers

Roberto Minerva holds a Ph.D in Computer Science and Telecommunications from Telecom Sud Paris, France, and a Master Degree in Computer Science from Bari University, Italy. He was the Chairman of the IEEE IoT Initiative, an effort to nurture a technical community and to foster research in IoT. Roberto has been for several years in TIMLab, involved in activities on SDN/NFV, 5G, Big Data, architectures for IoT. Now he is the EIT Digital Technical Leader for the H2020 project SOftFIRE, a research engineer in Paris Sud Telecom and the Chief Technologist in Bitify.it, a startup aiming to drive the digitalization of businesses in several industries. He is authors of several papers published in international conferences, books and magazines.

About the speakers

M. Sc. Giuseppe Carella is a senior researcher at the computer sciences and electrical engineering faculty of Technical University of Berlin, Institute for Telecommunication Systems, as well as senior solution architect at the Fraunhofer FOKUS institute. He received his M.Sc. in Engineering of Computer Science from the Alma Mater Studiorum University of Bologna in 2011. During his research work he focused on finding mechanisms and solutions for adopting Cloud Computing technologies in Next Generation Networks. In 2012, he realized one of the first proof of concepts for elastically scaling virtualized network functions. Since 2015 he leads the Open Baton open source initiative, and he is responsible for several European and Industrial project collaborations.

About the speakers

M. Sc. Simone Pizzimenti is an IT Security Consultant within Communication Valley Reply. He holds a Computer Science Master degree from the Milan University (summa cum Laude). He is currently working on the SoftFIRE project on the Security as a Service concept, aiming at integrating security virtualized features within a ETSI MANO architecture.

Agenda

- Part 1: Software Defined Network and Network Function Virtualization
- Part 2: Open Source In the Telco Domain
- Part 3: From the Design to the Implementation: an example on a federated testbed

Part 1: Software Defined Network and Network Function Virtualization

Introduction and Objectives

Objective of the Tutorial

- To provide a real example of how important technologies like NFV and SDN can be integrated and used in complex infrastructures
- A guide to use and program the infrastructure

- We do this by means of the SoftFIRE Federated Testbed

SoftFIRE in a Nutshell

- EIT Digital
- ADS
- Deutsche Telekom
- Fraunhofer FOKUS
- Ericsson
- Reply
- Technical University of Berlin
- University of Surrey

- 27 months
- 4 M budget
- Half of that for OPEN CALLS and external participation

Wave 1
- Interoperability Test
- First Open Call more than 20 applicants and 6 winners)

Wave 2
- Major architectural reshape
- Second Hackathon
- Second Open Call (13 exps)

Wave 3
- Third Open Call (12 experiments)
- Challenge (more than 15 applicants)
- Innovation Hackathon
- IETF Hackathon participation

Next Events
- Challenge at FEC3 in conjunction with FED4FIRE project in Paris March 15th
- IETF Hackathon (under discussion) in London 17-18 March
- Innovation Hackathon to be held in Rome 18-19 April in conjunction with Ericsson

Some Technical goals

- The creation of a Federated Platform that can foster the studies towards 5G
- A better integration between NFV and SDN
- The integration of Security with platform development
- The definition of KPIs and the initial proposition of best practices
  - It is not only the technologies it is also what you do to support their usage and how to measure their value

Defining Software Defined Networks (SDN) and Network Function Virtualisation (NFV)

Benefits for the Operators

The Rise of Softwarization

SOFTFIRE

Key drivers towards softwarization

Commoditization of HW,
i.e., general purpose HW is becoming more and more powerful and cheap. Cloud computing evolving towards the a Fog of very powerful terminals (smartphones)

Virtualization,
i.e., the capability to execute functions and services on virtual computational environments

Autonomics and Self-Organization
i.e., the ability of large system to adaptively and autonomously optimize their behavior

Commoditization of communications,
i.e., the ubiquitous availability of communications means

Availability of Application Programming Interfaces
for several resources and functionalities (pertaining to the Comm, Stor, Proc, Sens/Acting realms)

Open Source,
i.e., the ability to model resources and functions by means of software communities that share results and tools

Big data,
i.e. the capability to collect data in real time that describe a phenomenon associated with a resource or a person (or groups of them)

Softwarization instantiations

Softwarization of the Telcos
- Software Defined Networks (SDN)
- Network Functions Virtualization (NFV)
- Integration of SDN, NFV with Cloud

Emergence of new Services paradigms and Biz Models
- Servitization: Anything as a Service (e.g., IoT, IwT)
- Pervasive sensoring and actuation

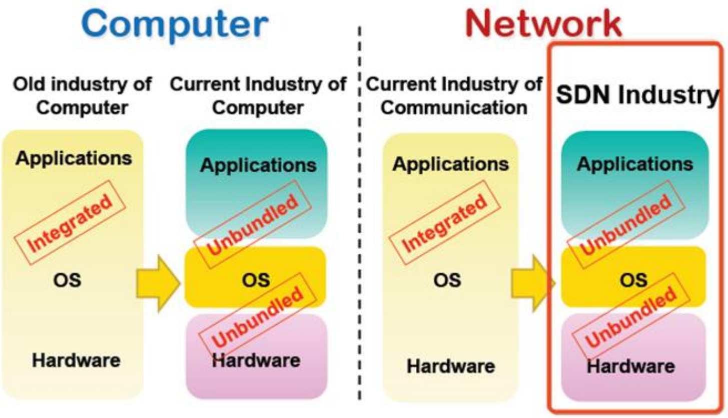
Virtual Continuum
- Creating new Virtual Worlds bridging the Physical
- WorldMetaverse: Integrating of the Physical and Virtual Worlds
- MicroManufacturing: 3D Printers

Big Data
- Real Time Data management
- The Bank of User Data
- Electronic Money

Edge as Point of Intelligence Accumulation
- Smart Terminals
- Different connectivity options
- Smart environment
- ....

Processing, Storage and Communication resources will be interchangeable. Their composition will allow to provide high quality services, while virtualization and autonomics will allow for system optimization (aggregating resources where they are needed the most)

Two Disruptive Factors in ICT Industry



- More and more functions from HW to SW
  - General Purpose HW is more and more usable also in mission critical systems
  - Think to WebCompany Data Centers
- Extensive Virtualization of Systems
  - From virtualization of Operating Systems to virtualization of entire Networks (e.g., Peer to Peer Networks)
- This leads to:
  - Strong separation of sw solutions from hw ones (disruption of the current ecosystem of Vendorship similar to what happened in computer industry)
  - Need to Master the Software (Programmability will became the differentiator for many companies)

Softwarization: Two possible Strategies for Telcos

- **Evolutionary**: for the development of current networks
  - Seamless integration, compatibility with legacy,…
  - Solutions from traditional Vendors (or some Start-ups) …
  - Costs Reductions (CAPEX, OPEX), probably
  - Competition
- **Revolutionary**: for the deployment of new (low costs) networks for new service
  - Disruptive low cost architectures using standard h/w
  - Open Source s/w
  - Low investments and costs
  - Open Innovation

Virtualization and Softwarization

- Mastering of software will be a differentiator also for communications services
- The ability to control simultaneously storage, processing, communications (and sensing) will be a strategic advantage
- The ability to integrate different environments will play a major role in service differentiation
- Behind the C – S front end, there are fully distributed systems with increasing complexity
- Security of the environment will be a major issue

How to bring the SW advantages within the Network

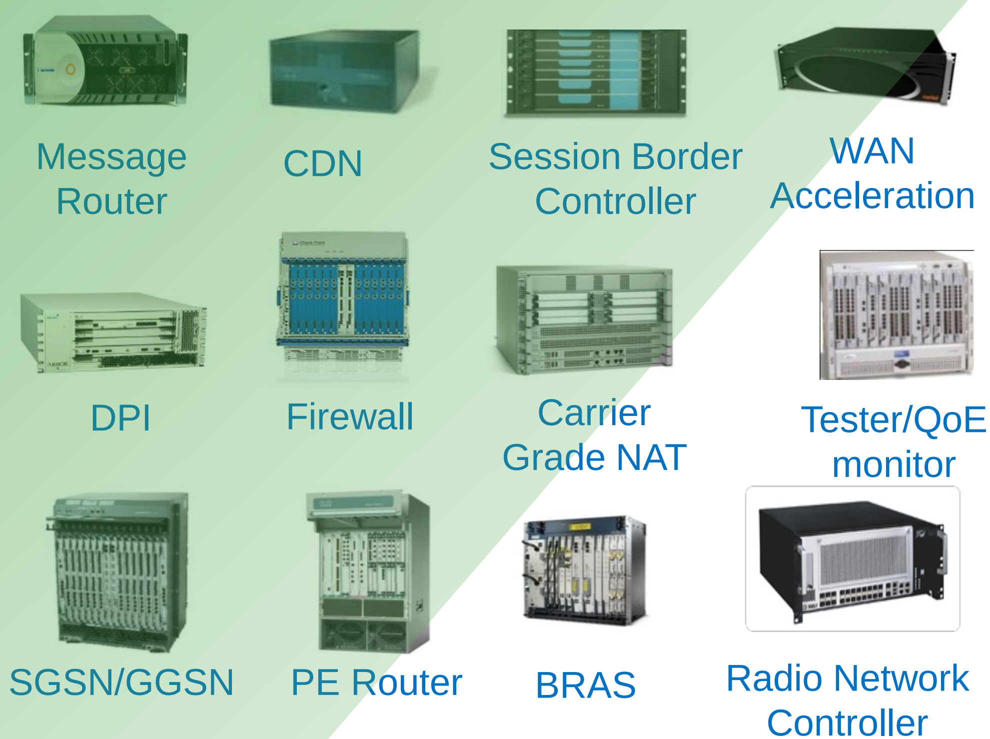
- Telcos are great consumers of sw
- They have already in place large sw platforms
- There is an opportunity to change the network infrastructure and bring in the flexibility of sw
- At what price?
  - Modify the attitude (sw vs. project management)
  - Move from a to a software company
  - Experiment and fail

Network Virtualization

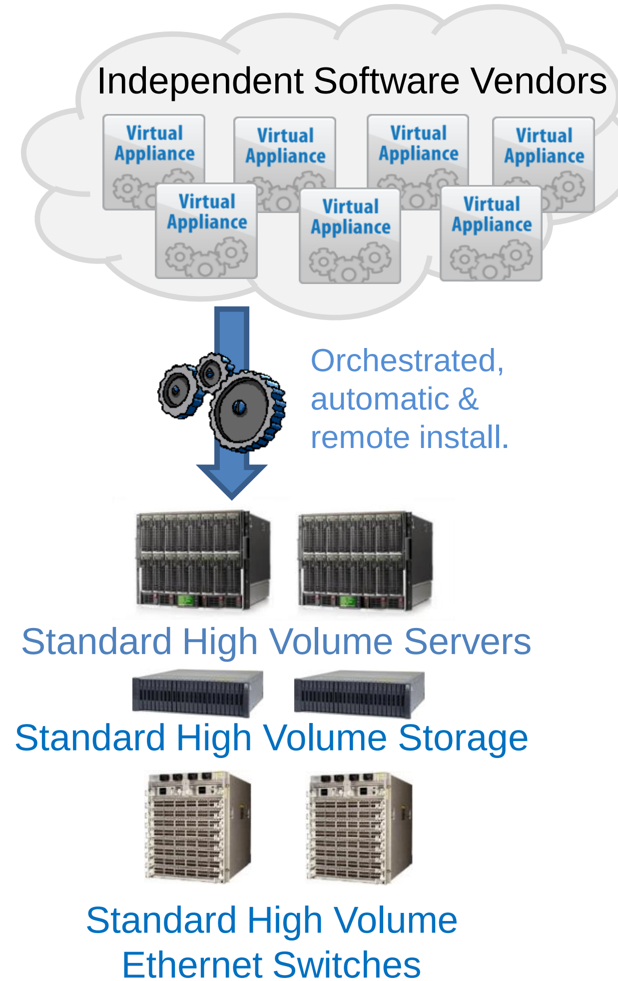
- Virtualization:
  - – The ability to run multiple operating systems on a single physical system and share the underlying hardware resources*
  - VMware white paper, Virtualization Overview
- A network wide virtualization (using the same paradigm used for IT resources) would allow:
  - To optimize the use of physical resources
  - To integrated deeply IT and Net resources in virtual networks tailored to apps requirements
  - To operate independent virtual networks "dedicated" to different Users and migrate them if when necessary

Network Functions Virtualization

Classical Network Appliance Approach



- Fragmented non-commodity hardware.
- Physical install per appliance per site.
- Hardware development large barrier to entry for new vendors, constraining innovation & competition.



Network Virtualisation Approach

Source: White Paper of NFV Operators' group

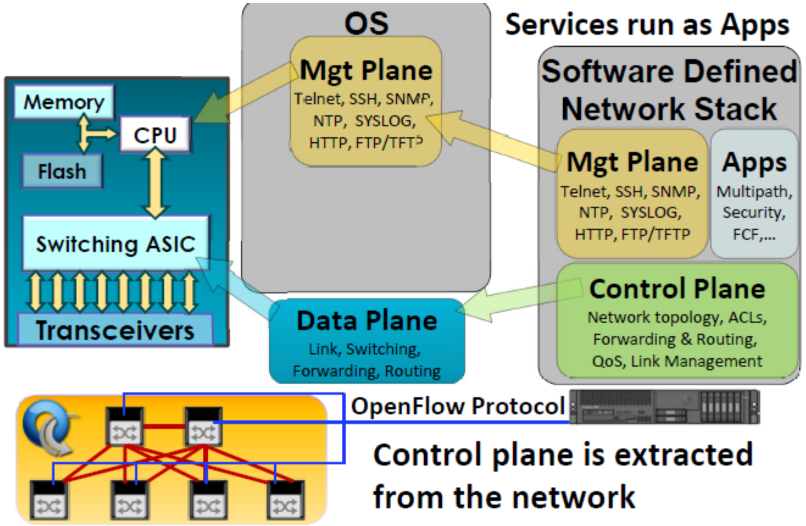
Virtualization

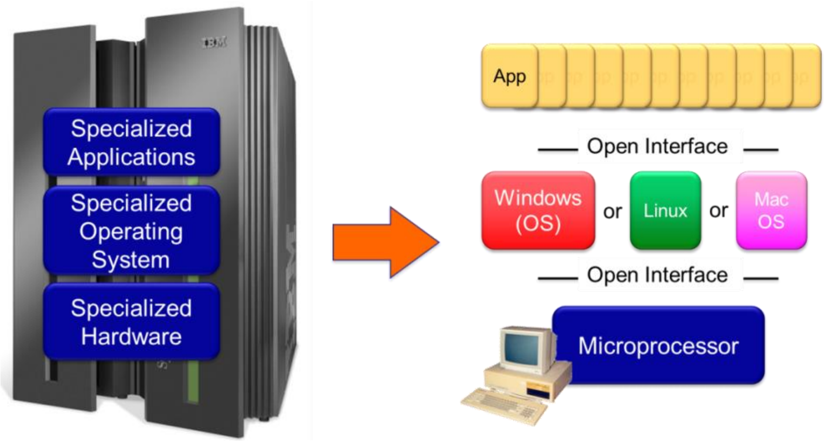
- Better reuse of HW
- Possibility to migrate old "functions" in a newer infrastructure
- Possibility to segment different customers and serve them better

- More software skills are needed
- Virtualization may slow down the "functions"

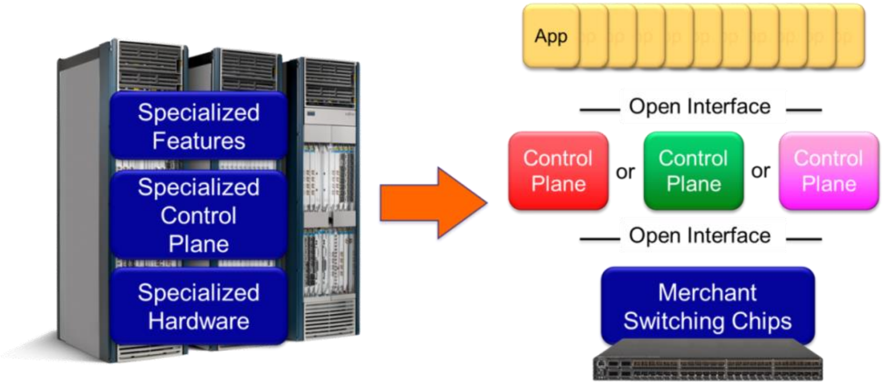
SDN: decoupling H/W from S/W

- In SDN, control and data planes are decoupled, so network control (NetOS) and states are logically centralized, and the underlying network infrastructure is abstracted from the applications.





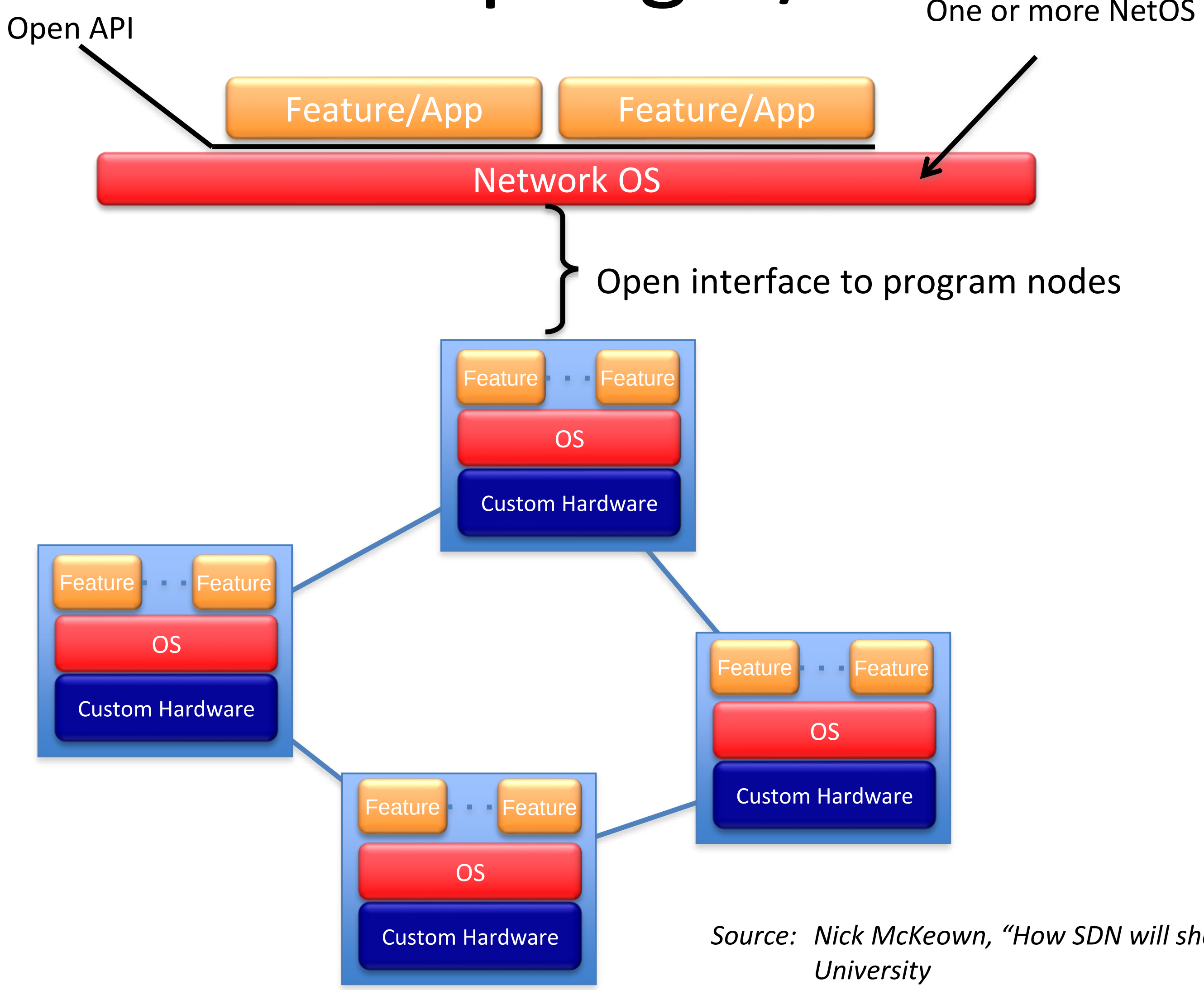
Evolution of PC



Evolution of SDN nodes

Source: Nick McKeown, "How SDN will shape networking", Stanford University

SDN: decoupling H/W from S/W

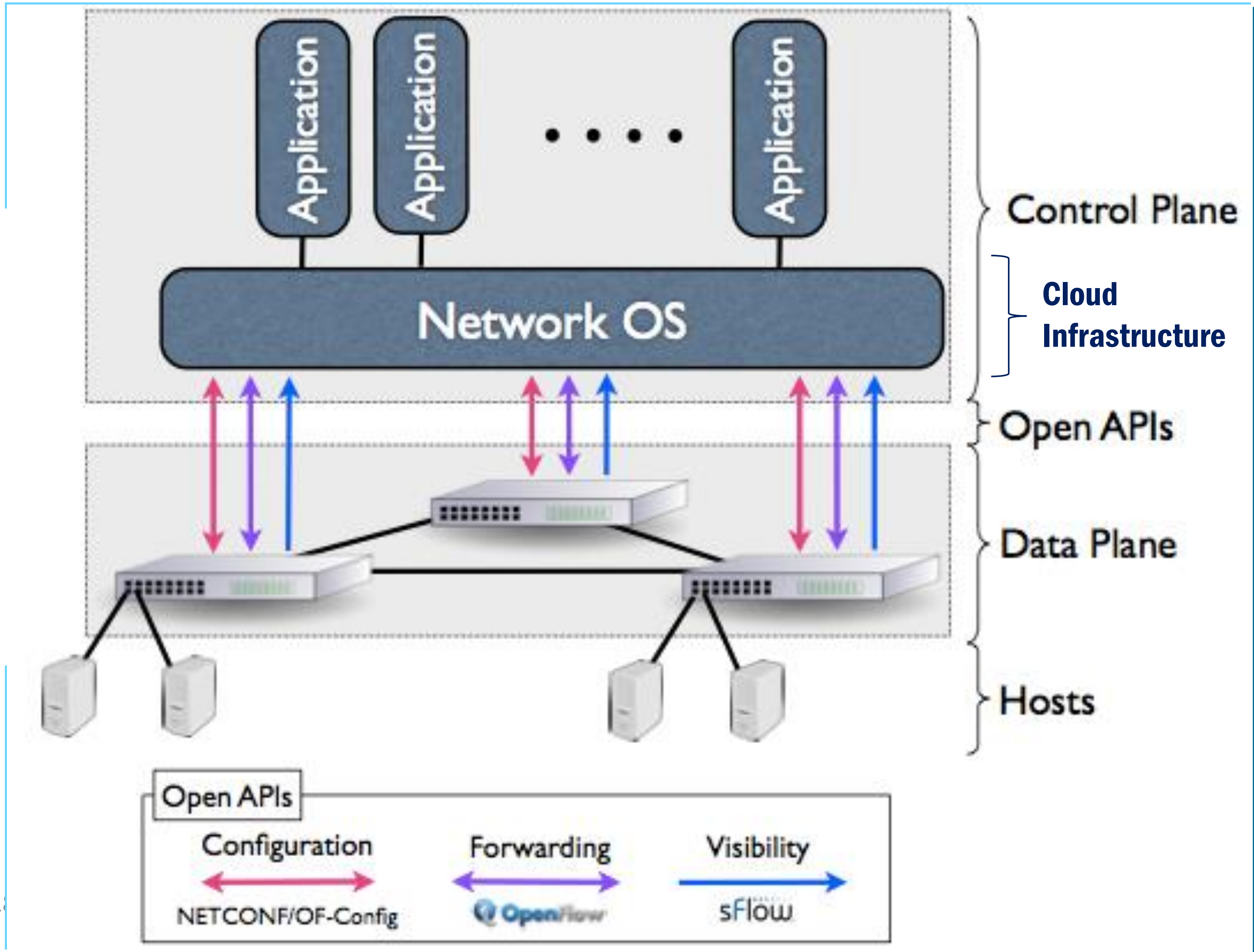


Source: Nick McKeown, "How SDN will shape networking", Stanford University

Software Defined Networks

- SDN fully decouples network control plane (a clean-slate approach)
- SDN offers programmable interfaces (API) to the network (i.e., “Control” is programmable)

Source: http://blog.sflow.com/2012/05/software-defined-networking.html



Current "Pros and Cons" of SDN

Pros

- Decoupling H/W from S/W (same evolution of PC)
- Network OS: logically centralized control plane
- Network programmability (API)
- Opportunity of complementing with network virtualization
- Multi-tenancy, MVNO
- H/W consolidation
- Reducing time to market
- Saving Capex and Opex

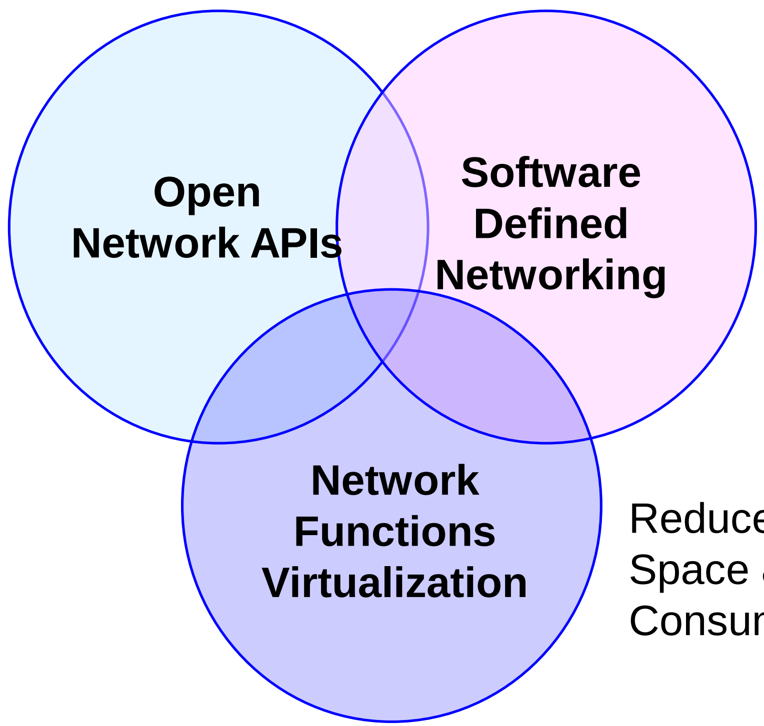
Cons

- Scalability and performance ? (h/w speedup required in core nodes)
- Consistency of network states (data) when logically centralising the control ?
- Today just focusing on Network
- Signaling overhead ?
- Availability, Complexity, Stability ?
- Security
- Interoperability

NFV + SDN

- NFV and SDN are highly complementary
- Both topics are mutually beneficial but not dependent on each other

Creates competitive supply of innovative applications by third parties



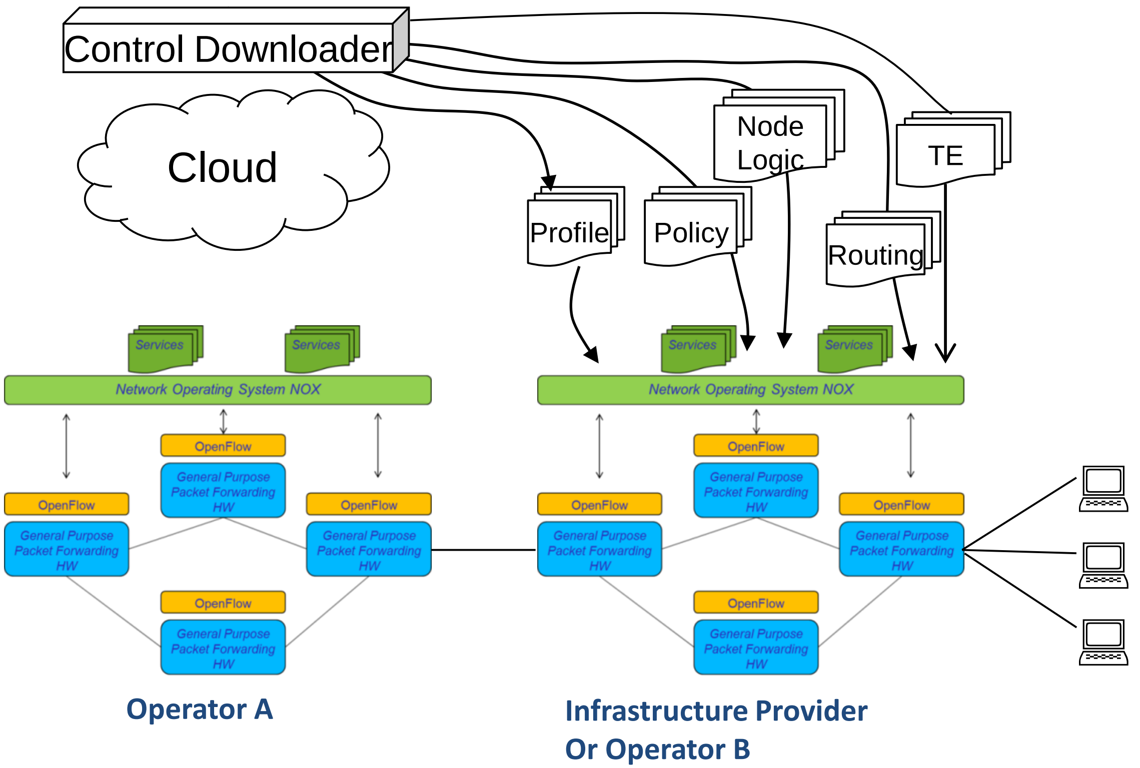
Creates network abstractions to enable faster innovation

Reduces CAPEX, OPEX, Space & Power Consumption

Adapted from White Paper of NFV Operators' group

SDN + NFV: a disruptive example, the Network Control Upload

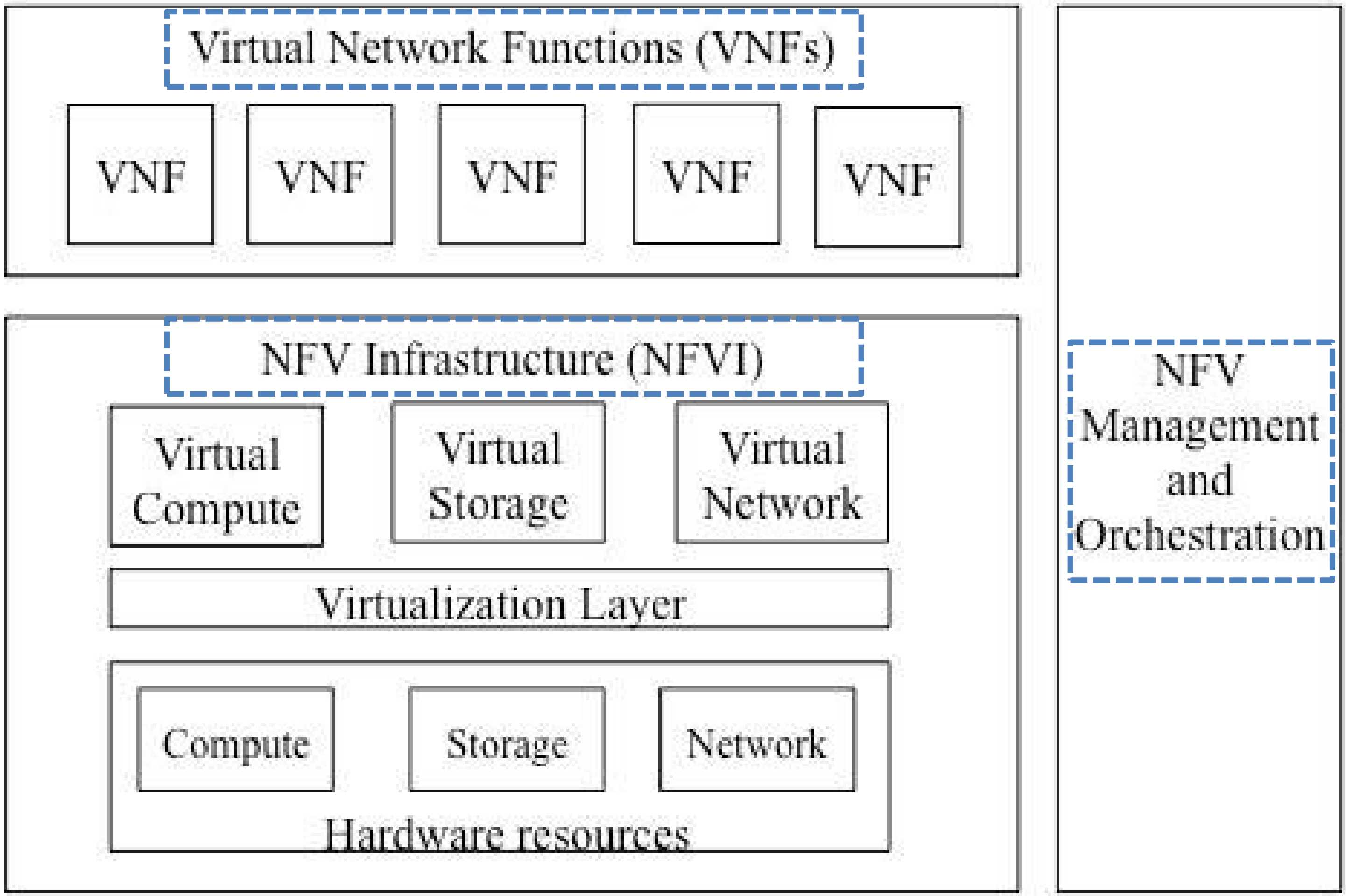
- **Downloading Control S/W from Operator A to another infrastructure Provider (beyond roaming)**
- **Each Operator (through agreement) could upload control nodes in other networks for better serving its customers**



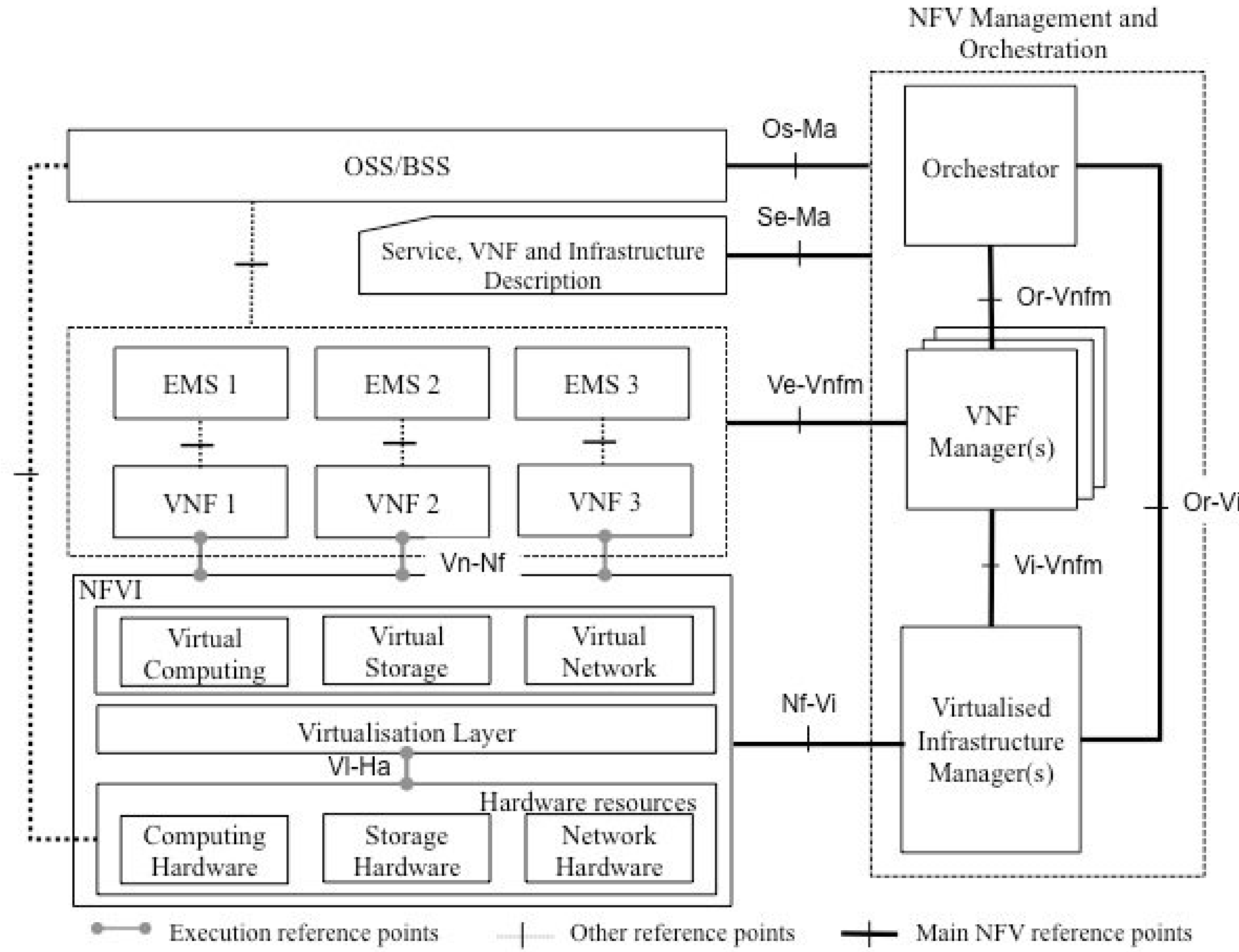
Entering new markets with low investments

A bit more of Architecture

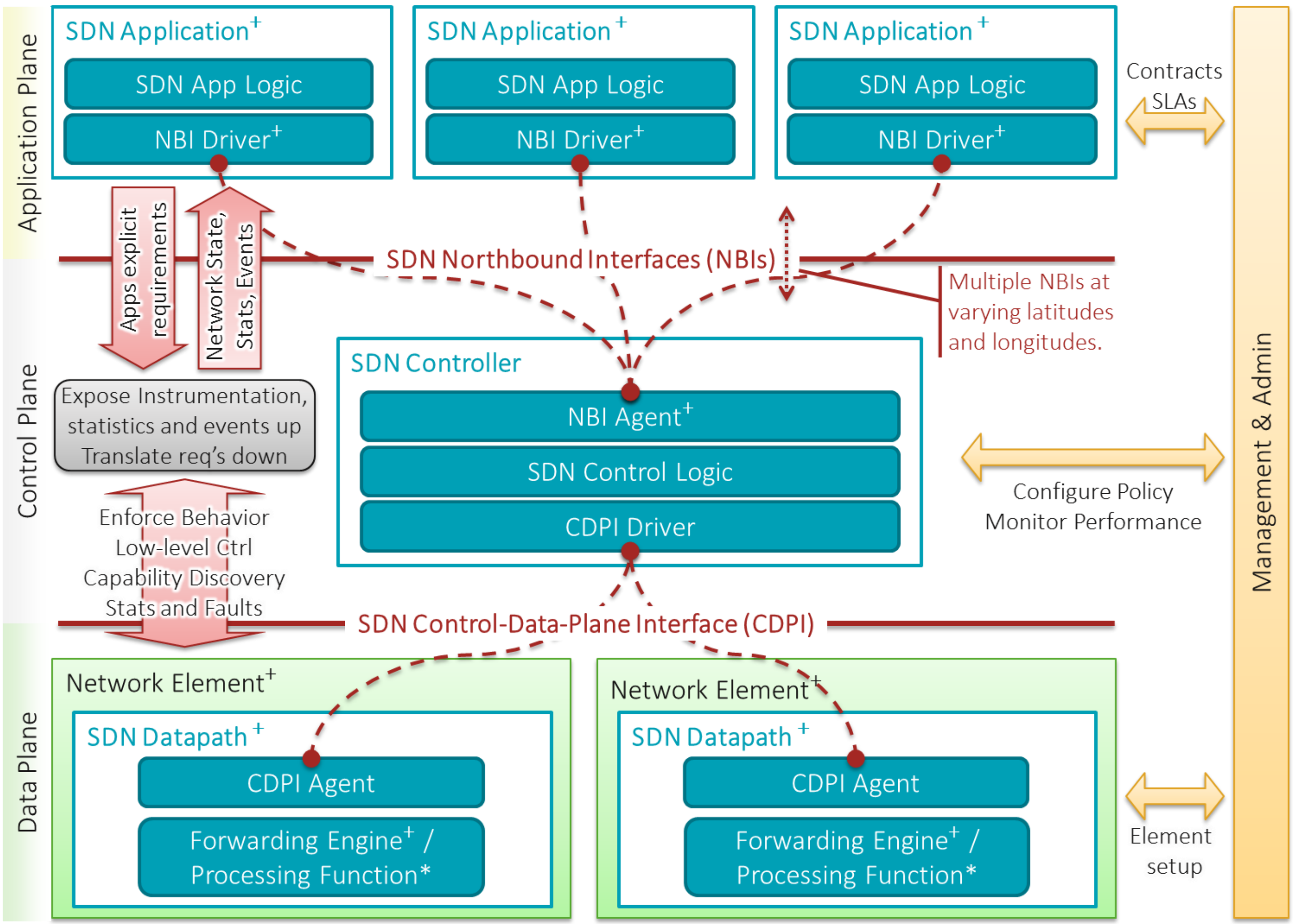
ETSI Functional Blocks



ETSI architectural framework, 2013



Open Networking Foundation (ONF) - SDN Architecture Overview

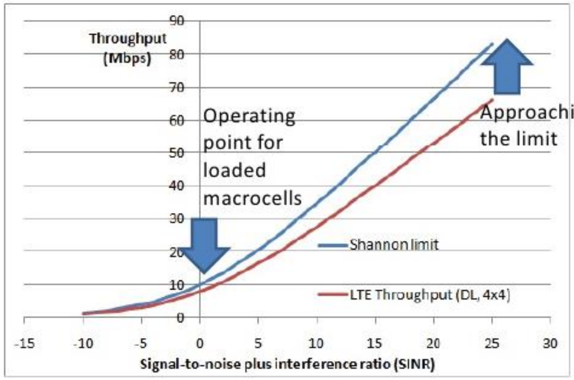


+ indicates one or more instances | * indicates zero or more instances

Some hints about 5 G

Three Features in 5G

- Next-generation system performance is close to the Shannon bound



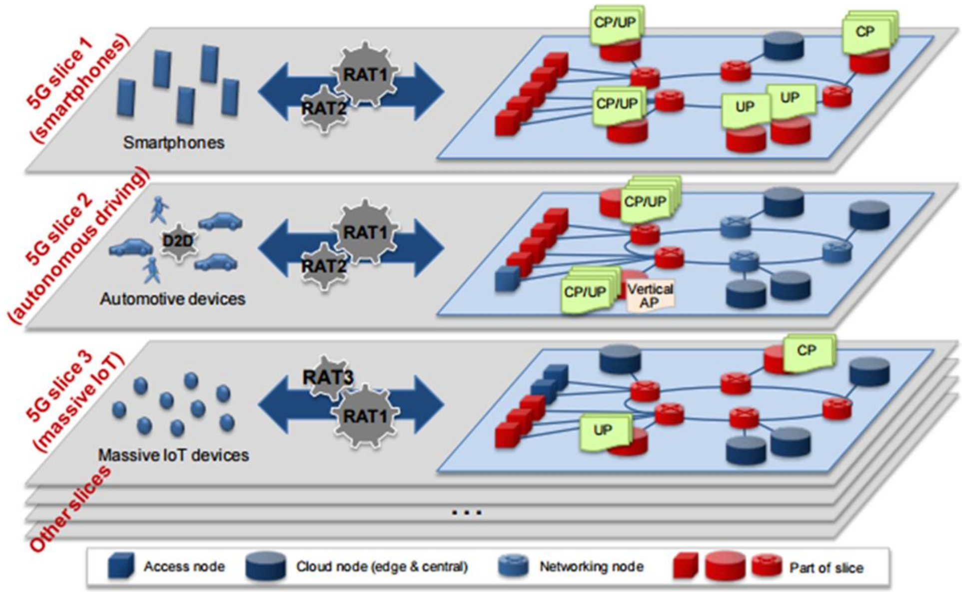
- Need *more cells* and *tighter interference control* to continue to increase capacity

http://www.slideshare.net/zahidtg/thinking-networks-by-prof-simon-saunders

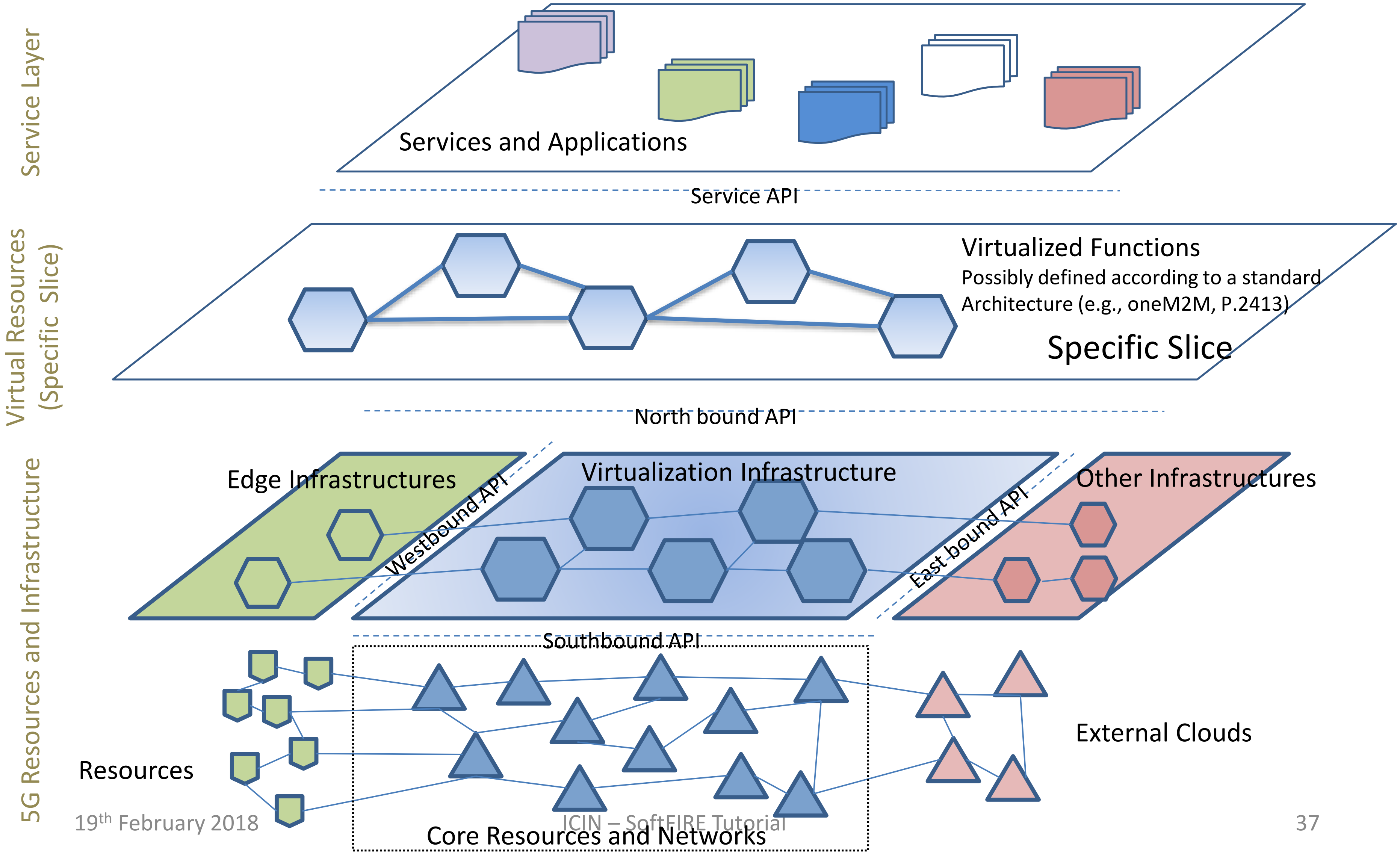
- Ultradense Network
  - Many Antennas
  - Interworking with legacy
  - Heterogeneous access
- Softwarization of the Network
  - By means of Software Dedined Networking (SDN) and Network Function Virtualization (NFV)
    - Reuse of existing architectures or new approaches?
    - IMS or evolution to other software architectures?
- EDGE and FOG computing
  - Moving the intelligence close to the terminals

5G Slice concept

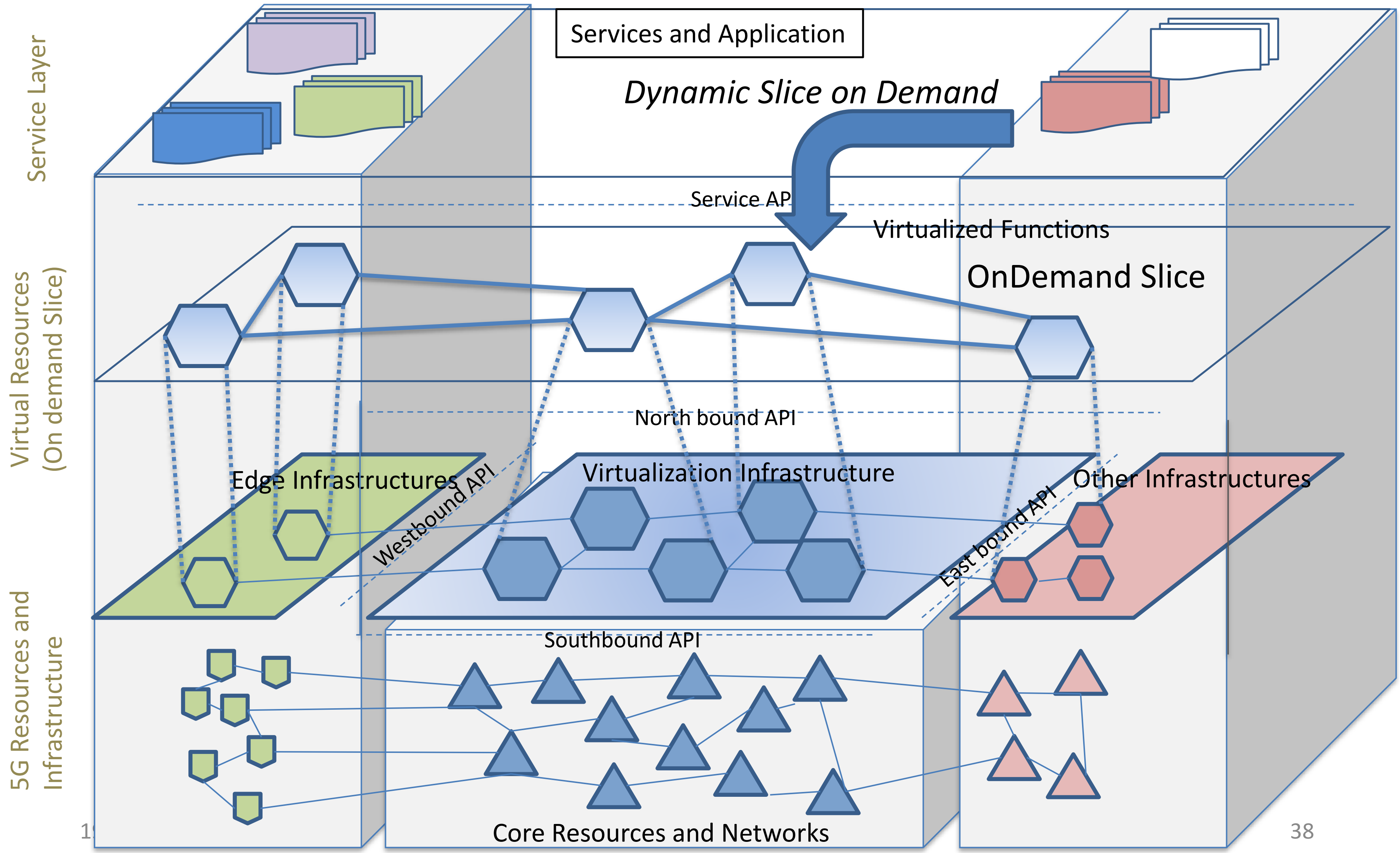
- A instantiation of virtual environments associated to allocation of access resources allows the dynamic creation of slices of functionalities.
- They can support specific devices, or Vertical applications with stringent requirements
- They can offer the possibility of providing a Network on demand capability to be used by service providers to tailor the network capabilities to their needs



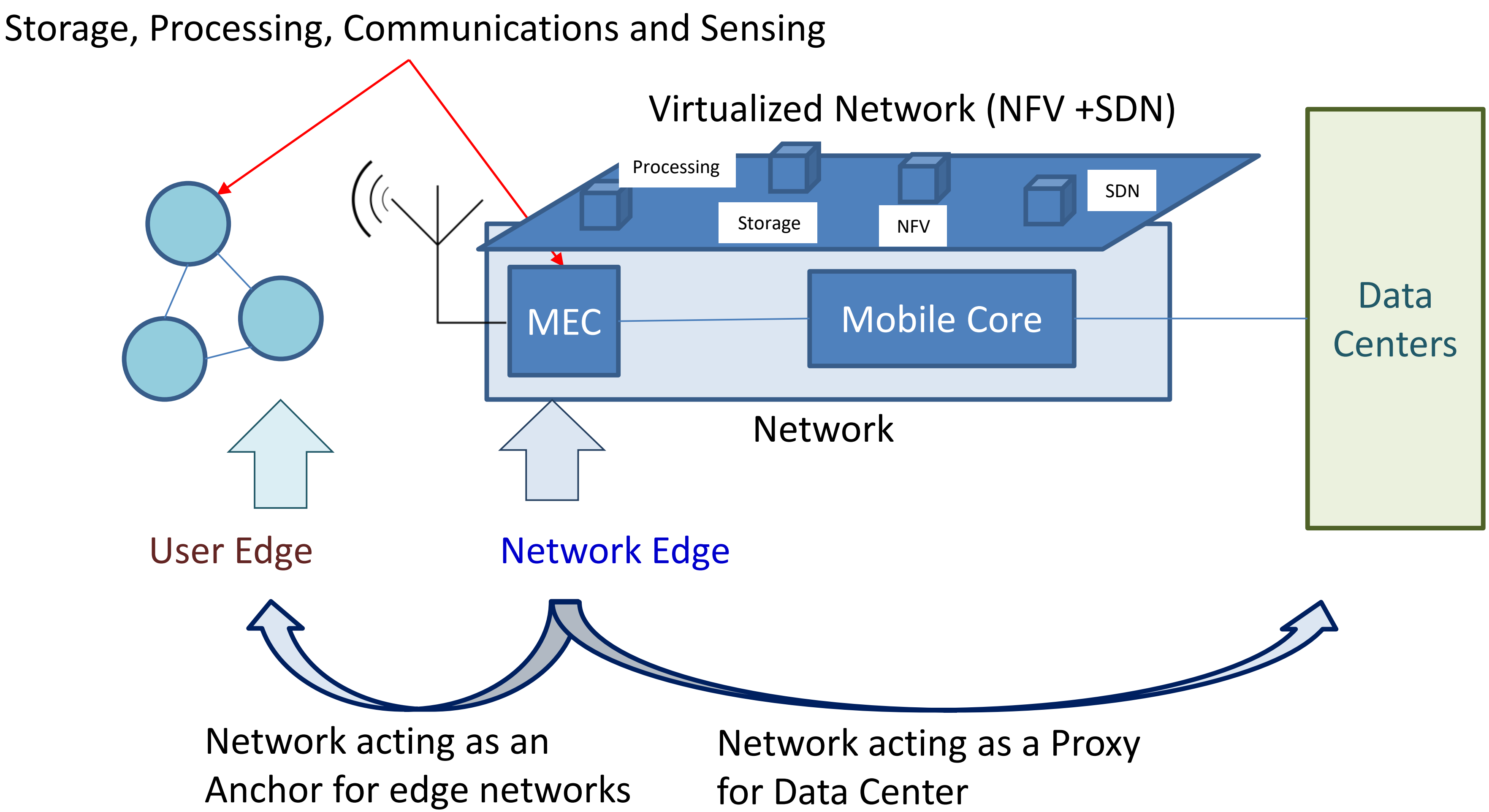
5G Slicing (Telcos view)



5G Slicing in reality?



Implementation at the Mobile Edge



New Business Model: Servitization of the Network

- The network can be highly virtualized and programmed
- The resources and the entire slices (environments) can be created and update dynamically
- Networks can be tailored to specific Service providers needs
- Communications can be complemented/substituted by storage and processing and sensing
- Big data flows can be dynamically analyzed
- New Biz models should be studied and mathematically analyzed
- This creates the possibility to offer the Network As A Service

The SoftFIRE platform (initial)

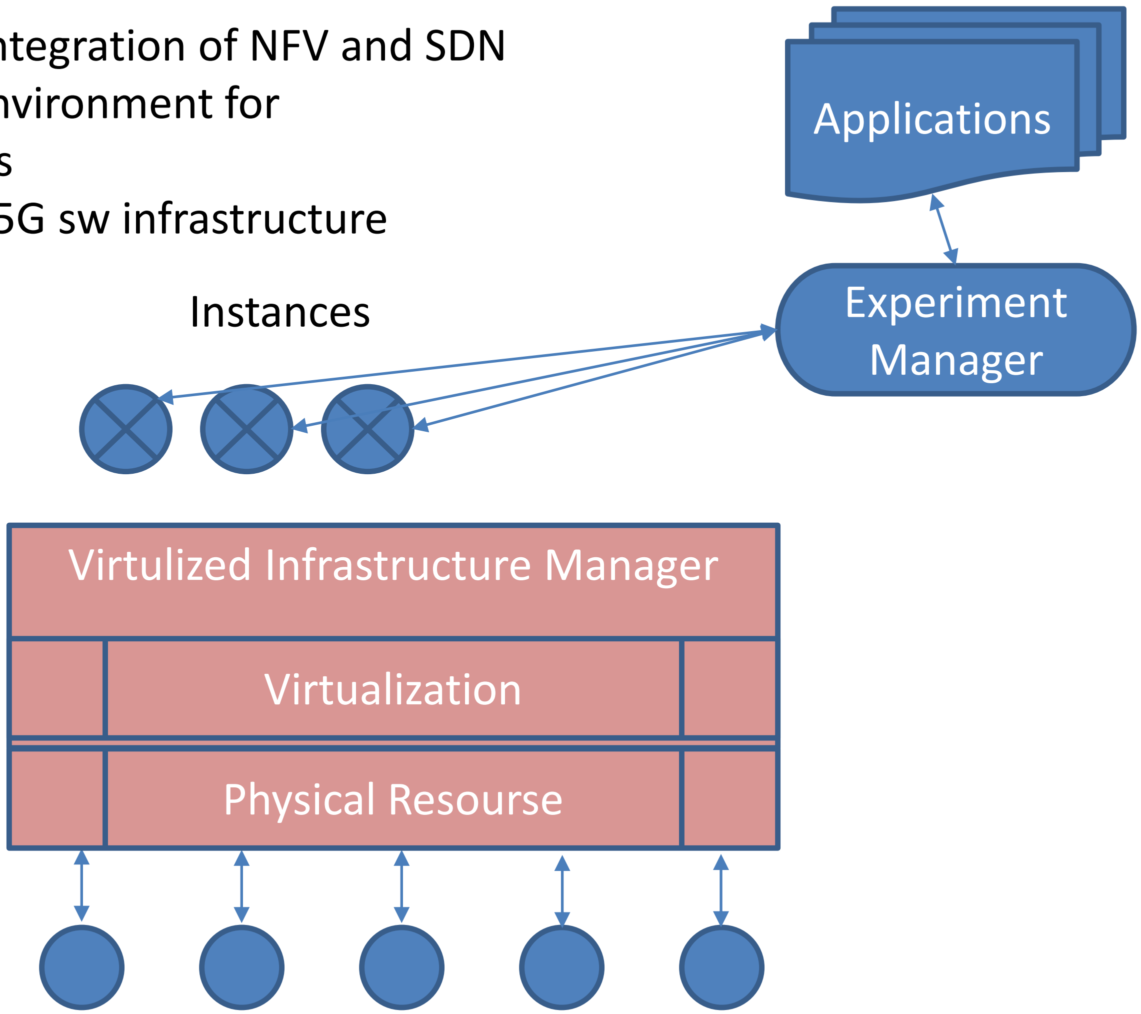
SoftFIRE Approach

- Bottom up
  - Starting from very different testbeds (in scope and technologies), the project has pursued a strong integration of capabilities, functionalities and communications
- Top Down
  - Starting from a requirements and standardization effort undertaken by several partners of the projects, the team has pursued the creation and building of an open programmable and secure middleware infrastructure
- Openness
  - The SoftFIRE project has pursued the provision of an open infrastructure

The SoftFIRE software

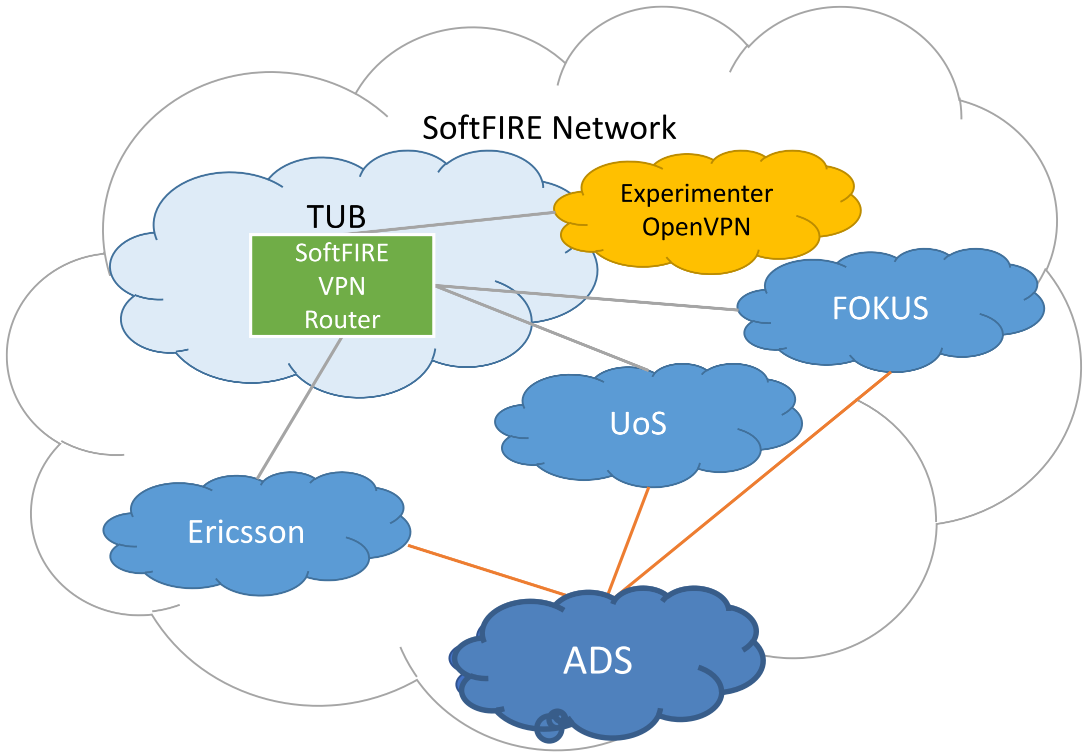
SoftFIRE aims at:

- Supporting the integration of NFV and SDN
- Providing a sw environment for experimentations
- Anticipating the 5G sw infrastructure



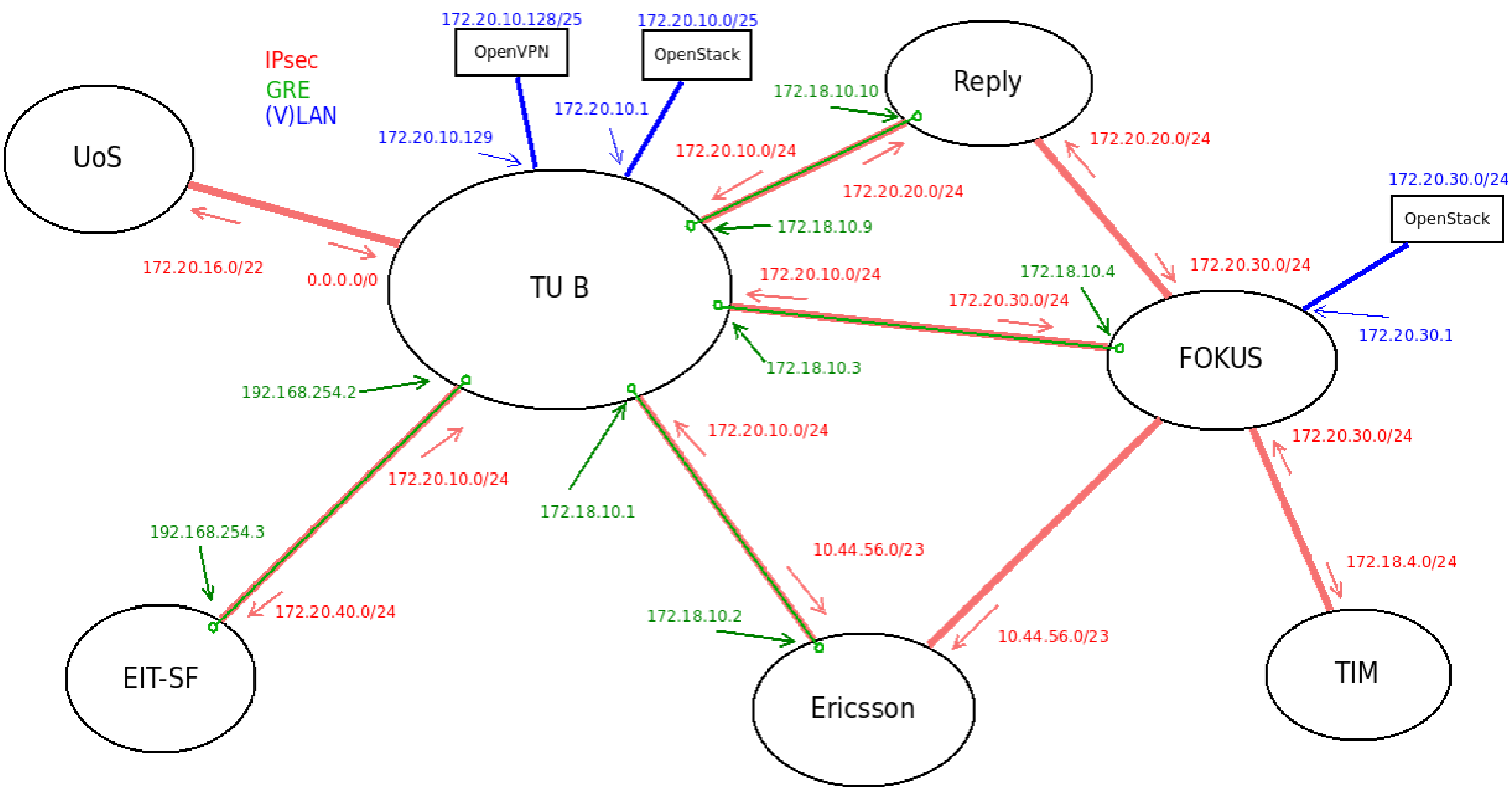
Federation

- Overview



Interconnectivity (example)

Multiple layer of Tunnels and encryption



A few considerations about interoperability, programmability and security

Interoperability

- Very hard to maintain a Federated Testbed
- Interoperability has issues everywhere:
  - OpenVPN
  - OpenStack
  - …
- Different Functionalities and approaches
  - SDN flavours
- Impact on orchestration! and management
- A discrepancy between NFV and SDN
  - NFV is more consolidated and standardized
  - SDN is more industry focused (For a and Initiatives)

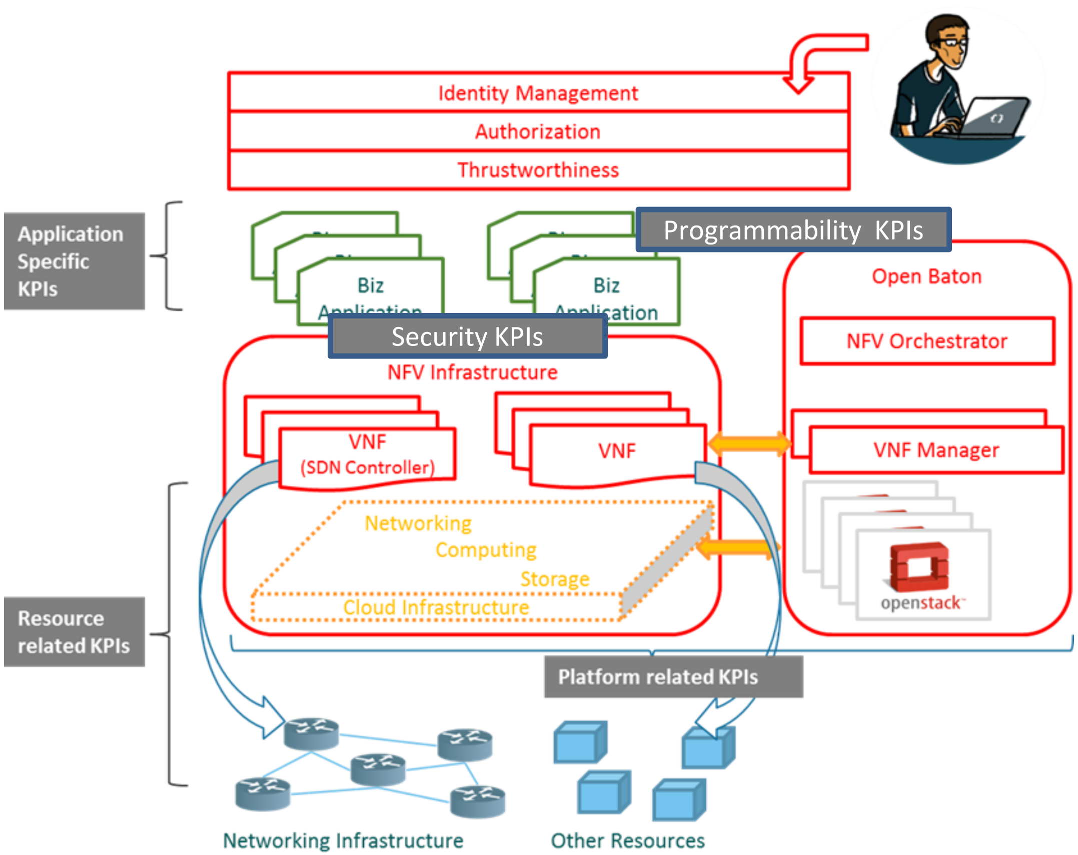
Programmability

- Multi dimensional approach
  - Functional (what I want to do)
  - Infrastructural (where I want to do)
- Plenty of Interfaces and APIs
  - Programming is not always easy
  - Many bugs are a combination of small issues
- Richness of functions and extensibility

Security

- Difficult to provide security functions in very heterogeneous platforms
- Importance of guidelines within the single IT department
- Approach chosen: to provide a set of security functions to programmers and users of the platform.

Monitoring of the platform and KPIs



So far KPIs are related to infrastructure. There is a strong need to define and build new KPIs that cover the entire life cycle of application on top of a NFV/SDN platfor.
SoftFIRE has compile a list of KPIs. They are available in Del D3.1

- Infrastructure (including resources with special focus on SDN related ones)
- Platform services
- Self-Organized Networking (SON) features. Note that this is essential for next generation mobile networking environments, i.e. 5G.
- The other two KPI groups are as follows:
- Programmability
- Security.

Part 2: Open Source In the Telco Domain

Open Source In the Telco Domain

Open Source collaboration as alternative to SDOs

Rapid innovation, interoperability, customizability, flexibility, and freedom as key factors for success

Standards vs de-facto standards:

- Risk of divergence between standards and open source reference implementations

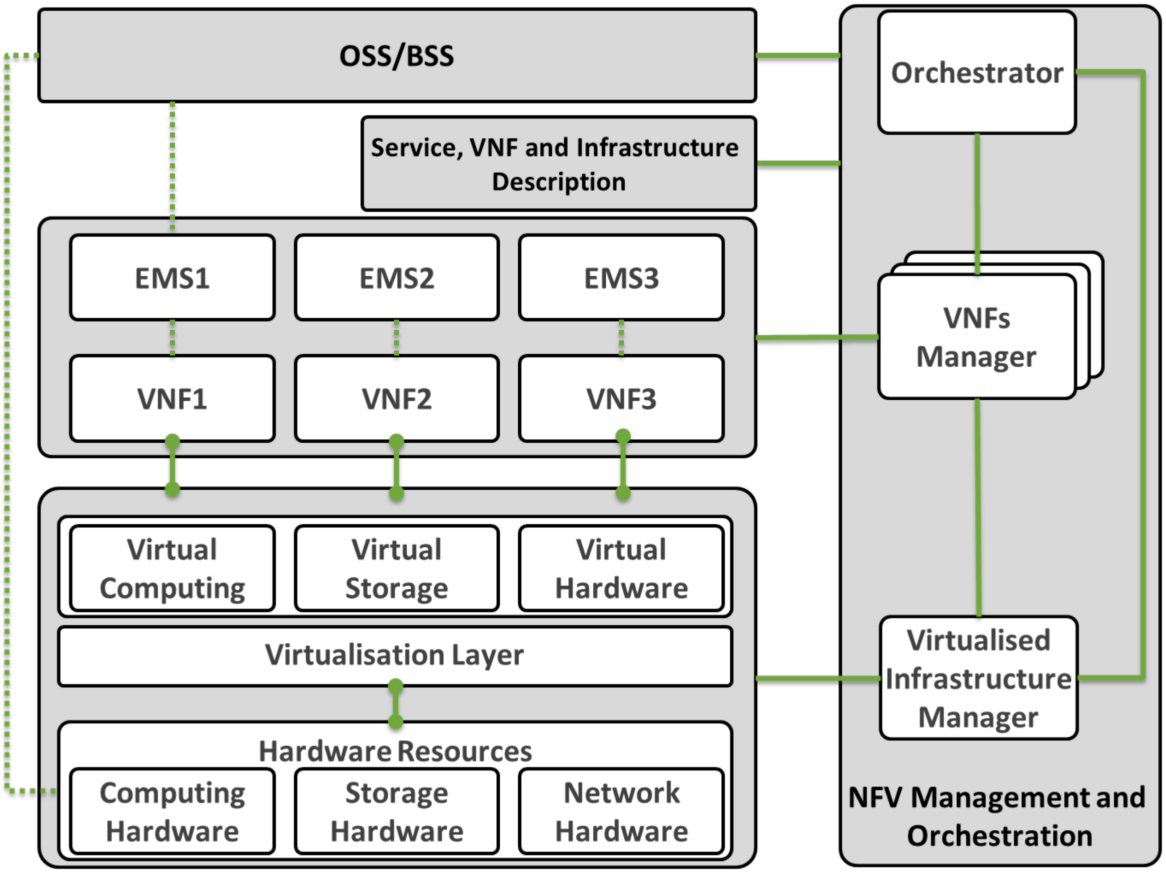
Diversification:

- It is definitively important to determine the boundaries for the standardization work
- Same for the open source: extensibility and customization are key

…and many more

“if god gave us the source code, we would change the world”

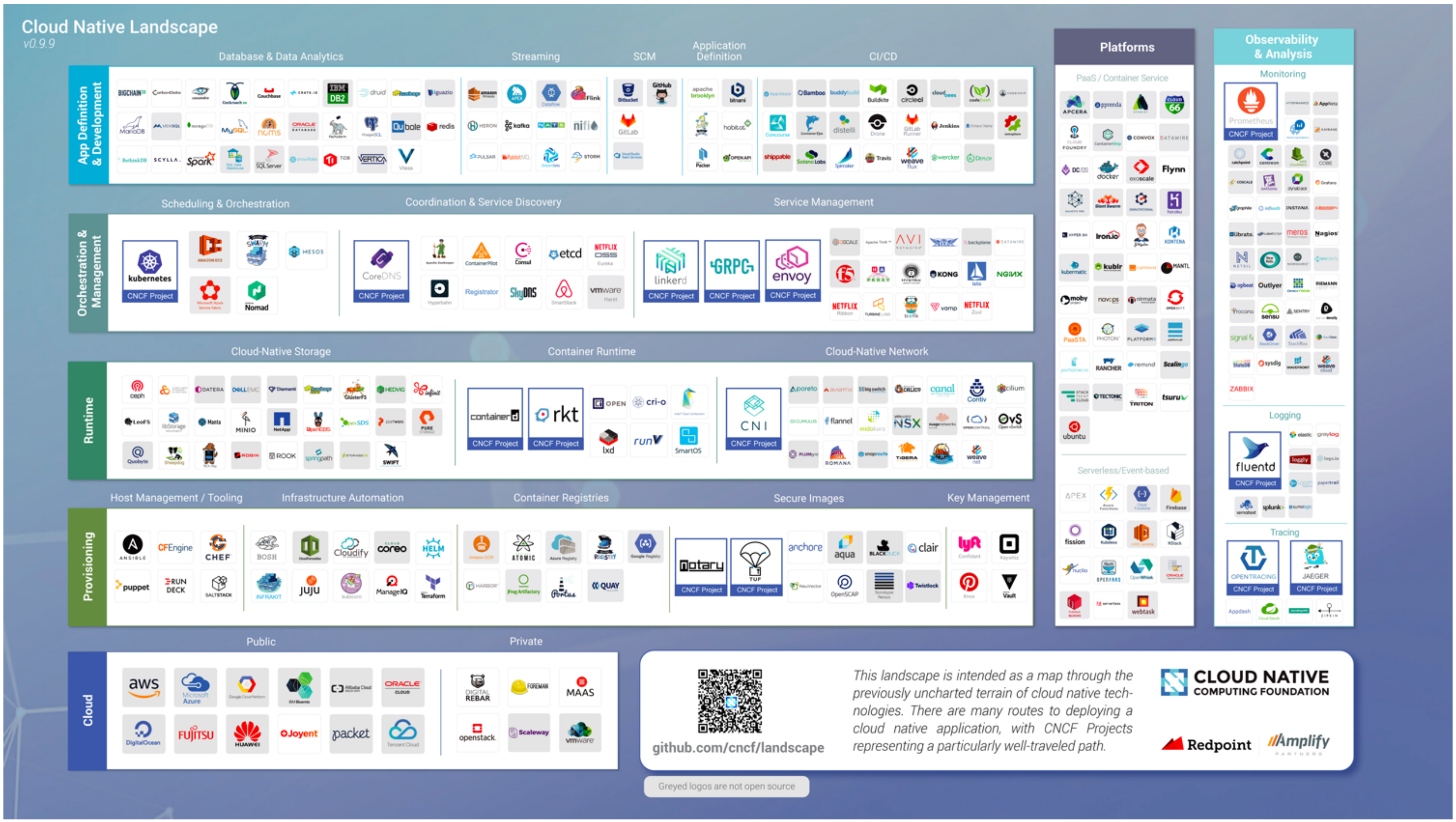
The NFV Ecosystem



tacker

53

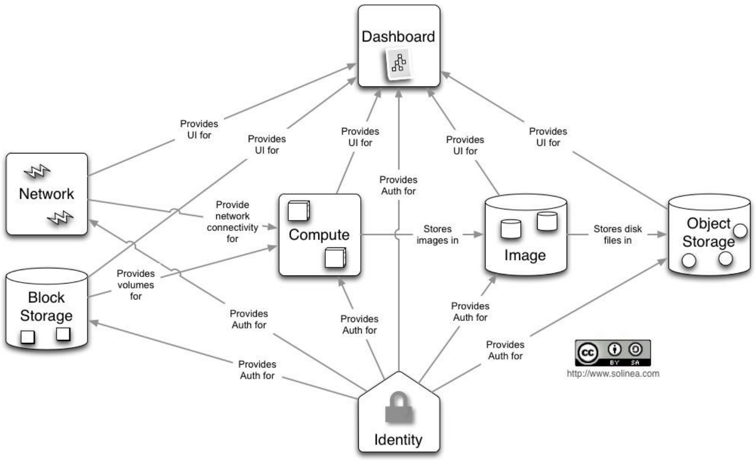
The Cloud Native Landscape



OpenStack – the standard de-facto VIM implementation

OpenStack is the most used world-wide cloud manager:

- It is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter.
- Managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface
- It is composed of 6 key core services (nova, glance, neutron, etc.) and 13 (and even more) optional services communicating over a message bus based on a microservices architecture



More info at: https://www.openstack.org/

OpenStack – several distributions

Development Environments:

- Single host installation: devstack, packstack

Production environments:

- Multi host installation including high availability: mirantis, juju+maas, packstack

Many closed source distributions: HPE, Oracle, IBM, Rackspace, etc.

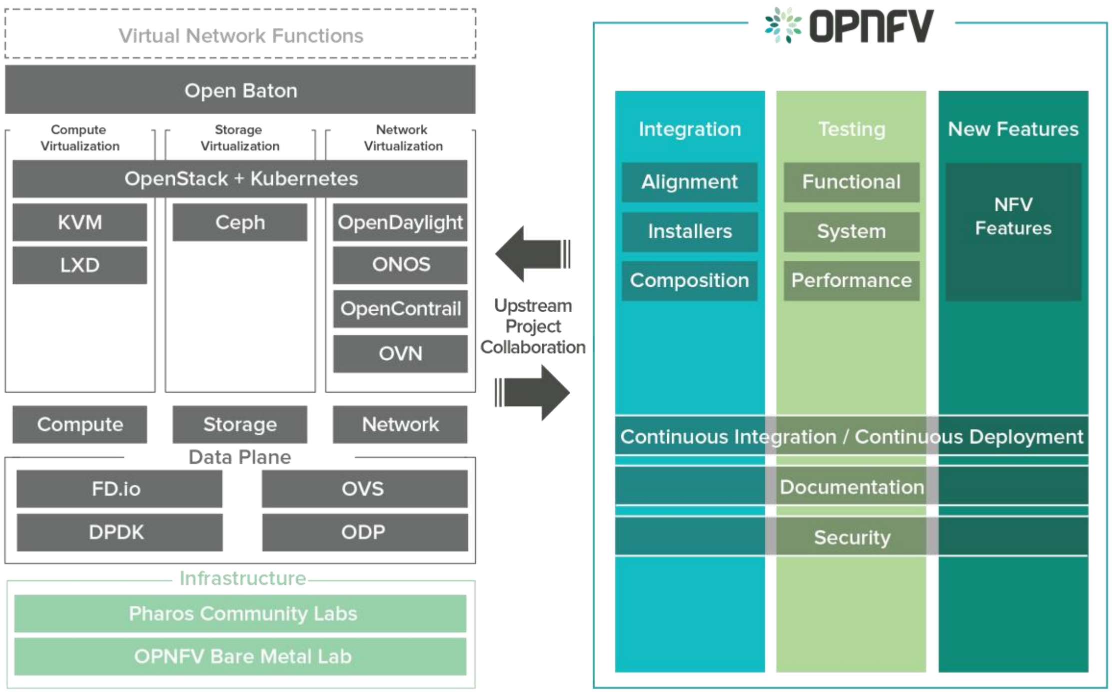
…and many more...

Open Platform for NFV (OPNFV)

Project launched in October 2014

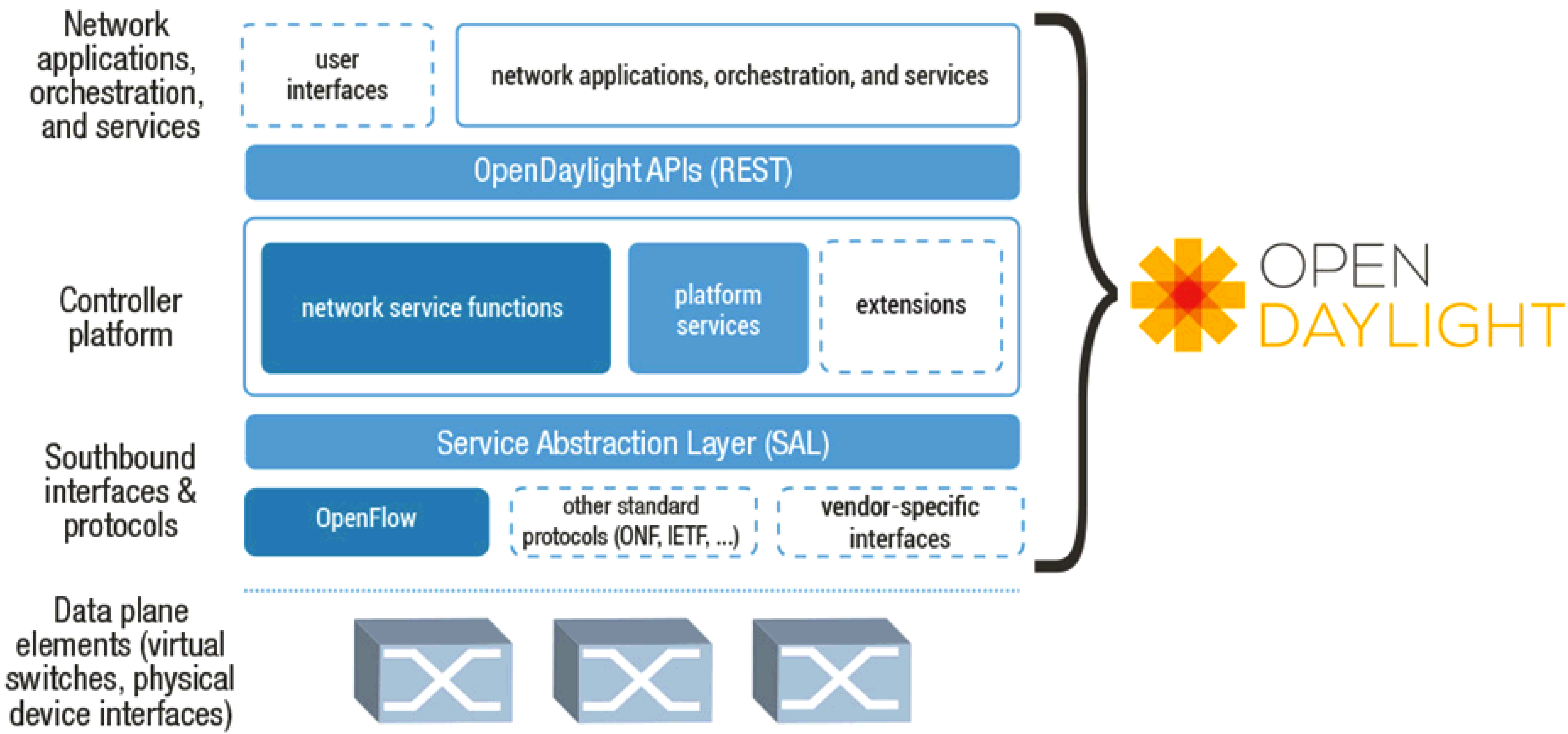
Main Objectives:
- Realize an open source platform for promoting interoperable NFV solutions
- Extend the list of features that are provided by OpenStack for supporting NFV requirements

Euphrates (rel.5) released in October 2017



More info at: https://www.opnfv.org/

OpenDayLight Architecture

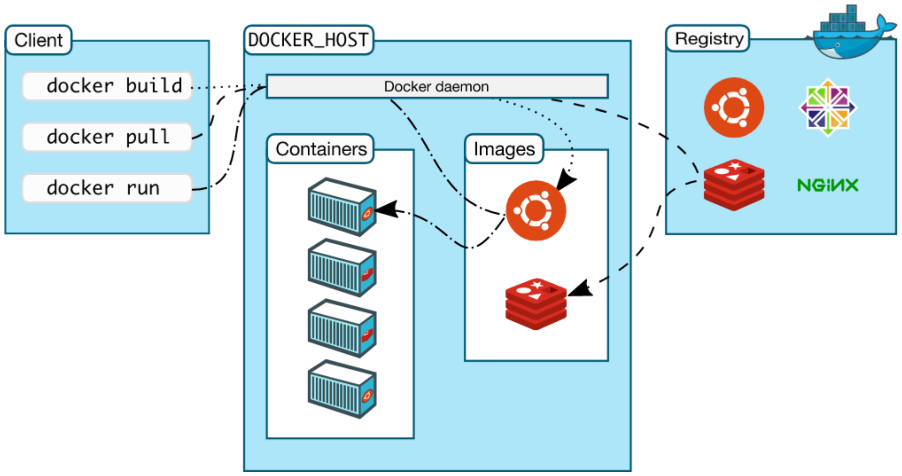


Docker

Project started in March 2013 with the objective of “build once…(finally) run anywhere”

It provides a clean way for porting applications between different environments

Isolation and automation as the most important features



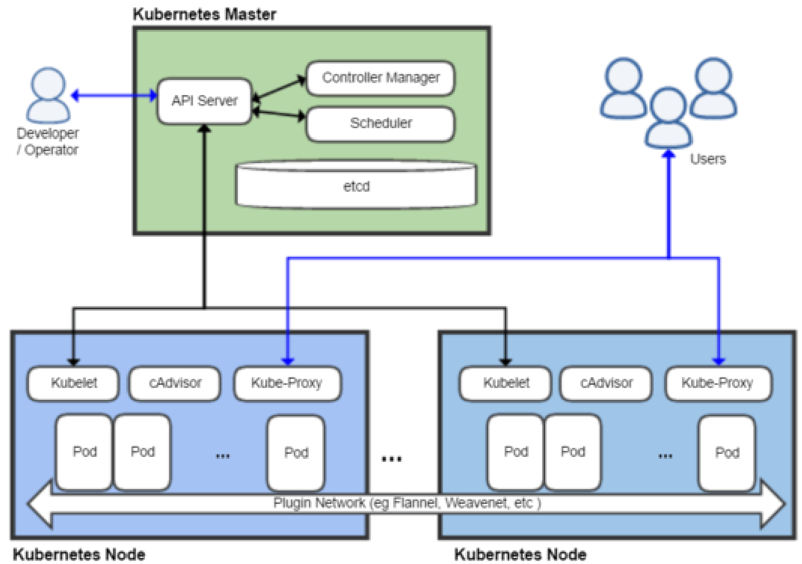
More info: https://docs.docker.com/engine/docker-overview/

Kubernetes

- Originated by Google in 2014 and donated to the Cloud Native Computing Foundation

- It builds on top of docker, exposing a set of primitives providing techniques for deploying, maintaining and scaling applications

- Based on recent surveys Kubernetes is the leading container orchestration technology

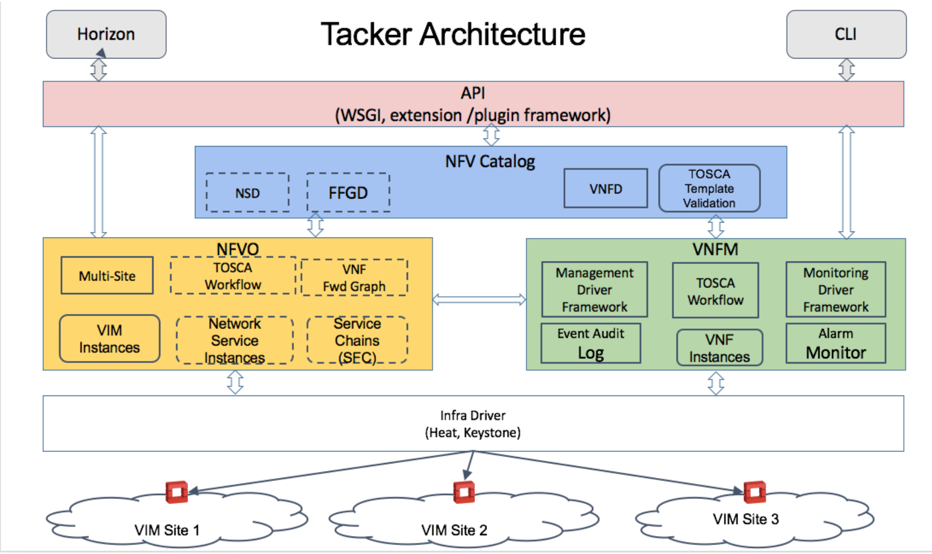


Source: https://en.wikipedia.org/wiki/Kubernetes

Survey link: https://www.cncf.io/blog/2017/06/28/survey-shows-kubernetes-leading-orchestration-platform/

OpenStack Tacker

- Initially driven by Cisco, Brocade and Intel started during the Juno release
- Initially intended as VNFM; extended its scope towards NFV Orchestration
- Transforms TOSCA templates into Heat templates



More info at: https://wiki.openstack.org/wiki/Tacker

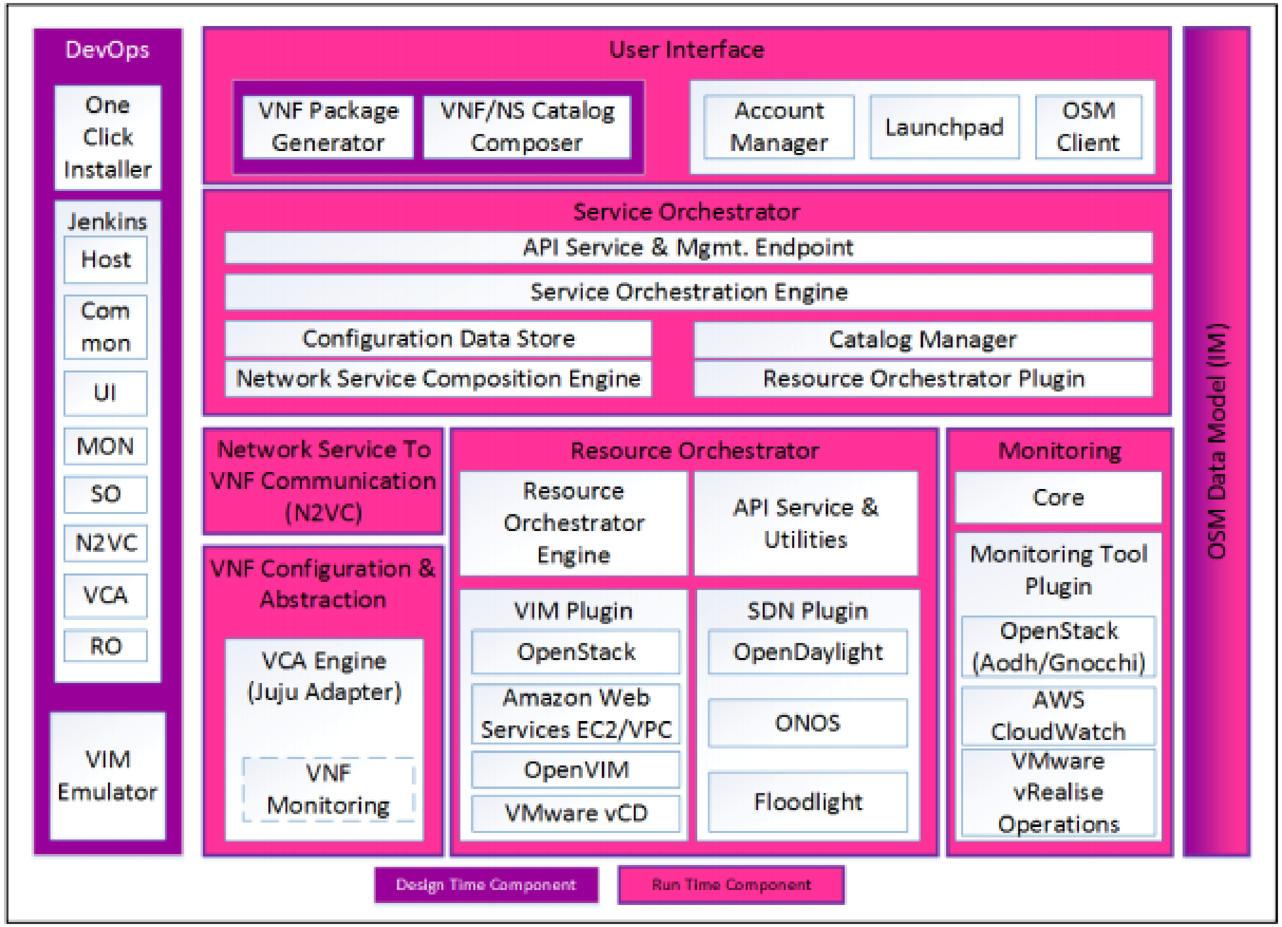
Open Source MANO

- Open Source MANO is an ETSI Initiative led by Telefonica and launched by a small group of Operators and Vendors with the objective of building an open source MANO platform:
- Release 0 launched at the end of May 2016
- Release 3 ongoing
- Based on three other major upstream projects:
  - OpenMANO
  - Juju
  - Riftware

- As of today, around 90 companies have joined the initiative.

More info at: http://osm.etsi.org/

Open Source MANO REL.3 Architecture

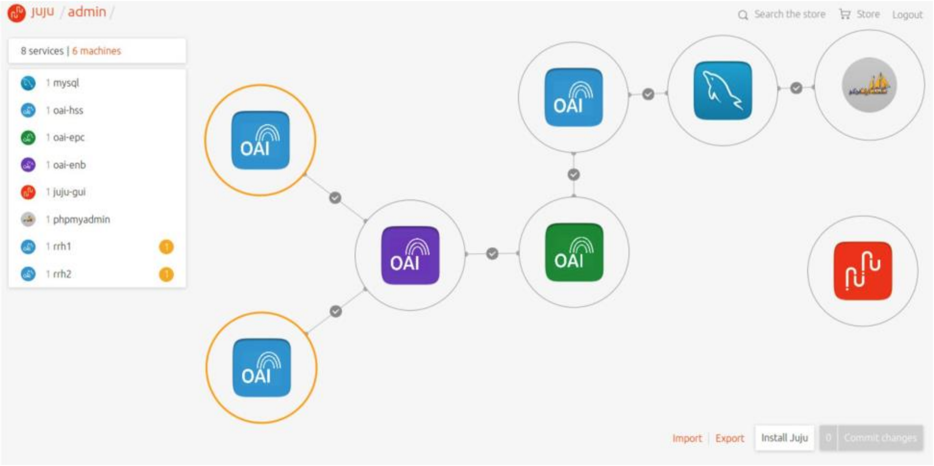


Source: https://osm.etsi.org/images/OSM-Whitepaper-TechContent-ReleaseTHREE-FINAL.PDF

Juju as Generic VNFM

- Juju is a Service Orchestrator maintained by Canonical:
  - It provides a single orchestration element able to compose services
- Concept of charms as set of scripts implementing the different steps in the service lifecycle
- Single environment with multiple providers supported (Amazon, OpenStack, etc.)

- In the NFV context Juju is typically associated to the role of a Generic VNFM

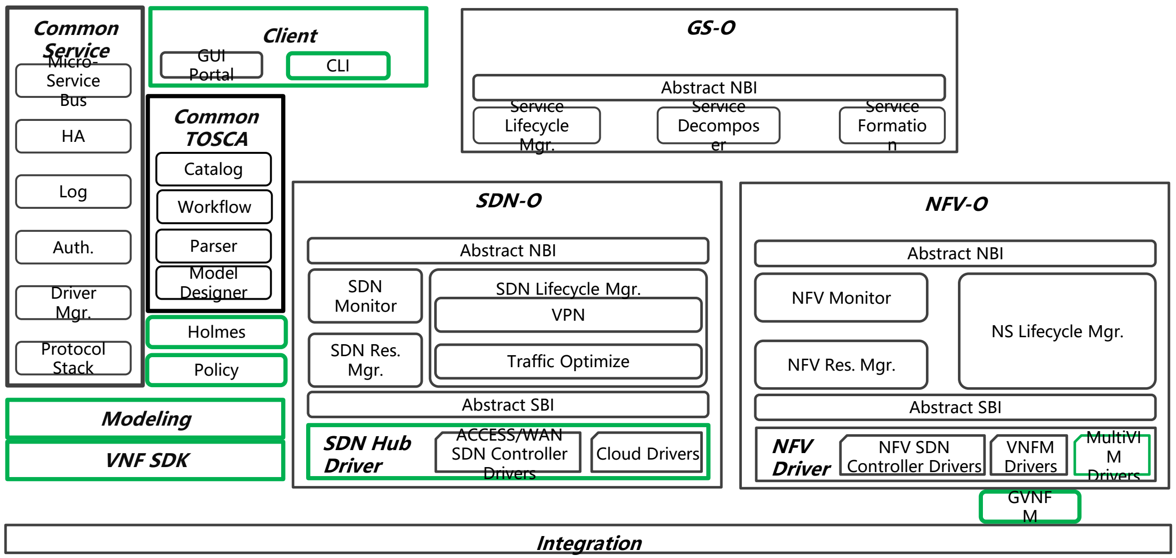


More info at: https://jujucharms.com/

Open-O

Open-O is a project launched by the Linux Foundation in June 2016

- Open-O plans to include elements defined by ETSI NFV such as NFVO, and integration with EMS, VNFM and VIM, including OpenStack
- Open-O, like OSM, introduces a 'Service Orchestrator' component on top of the NFVO resource orchestration function

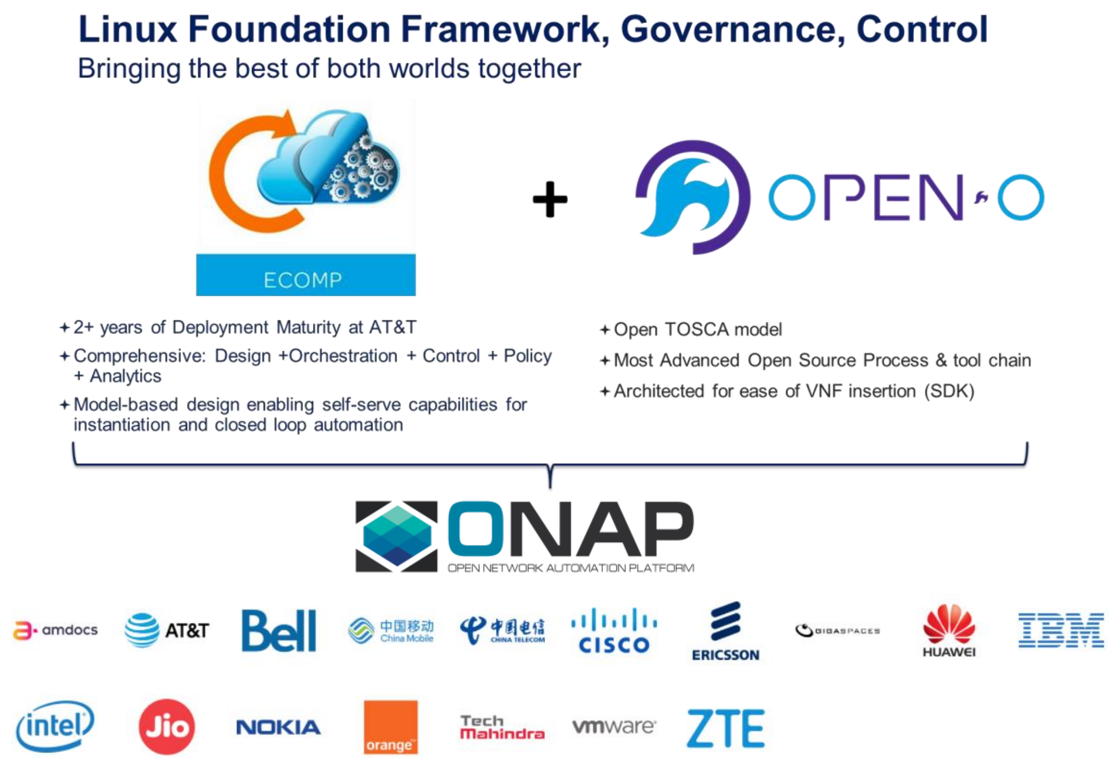


More info at: https://www.open-o.org/

Source: https://wiki.open-o.org/display/AR/Architecture+Home?preview=/65822/4982924/OPEN-O%20MercuryArchitecture.pdf

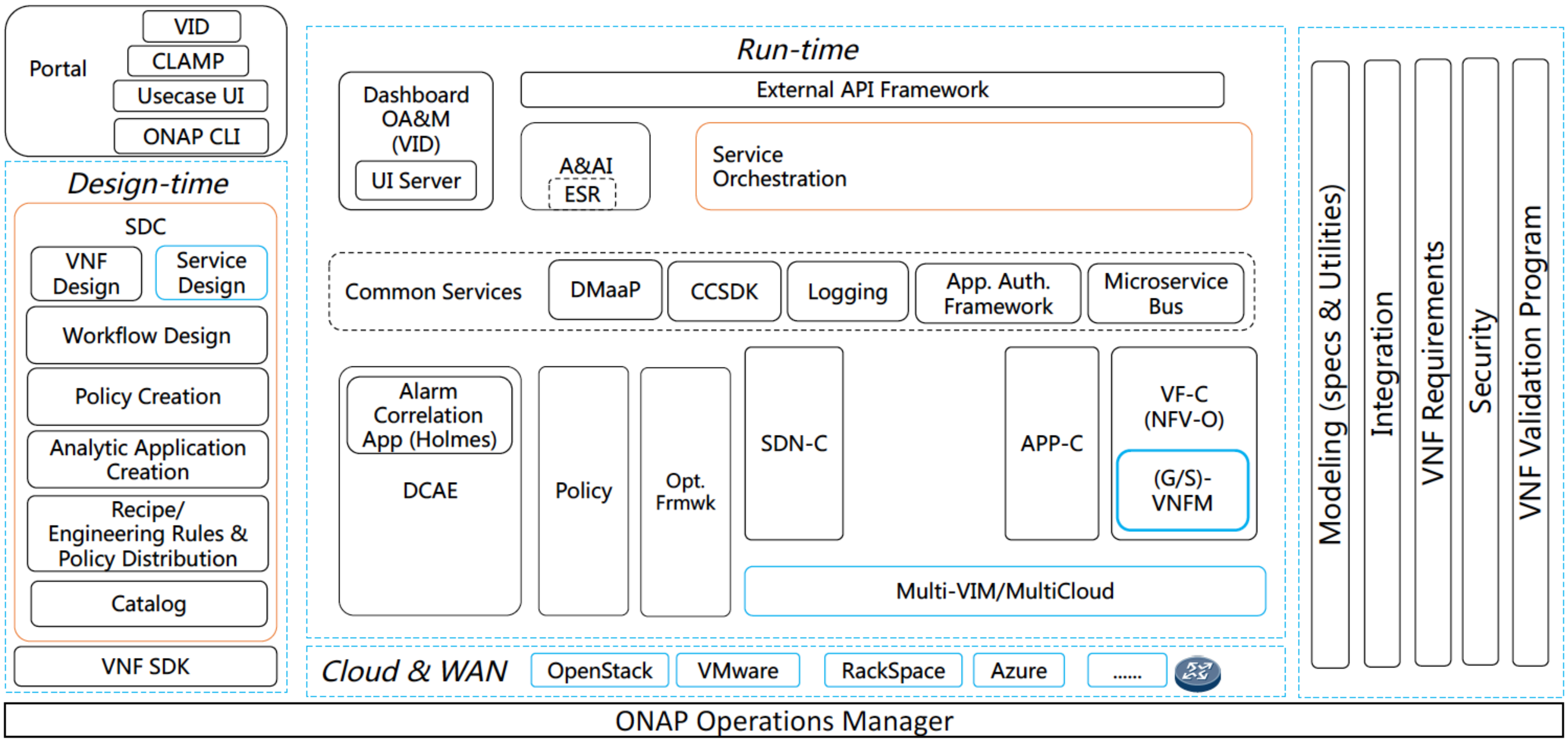
Open Network automation platform (ONAP)

- ONAP project launched in February 2017 under the Linux Foundation
- It represents a merge between Open-O and Open ECOMP (AT&T)
- As of September 2017 it counts around 50 members



More info at: https://www.onap.org/

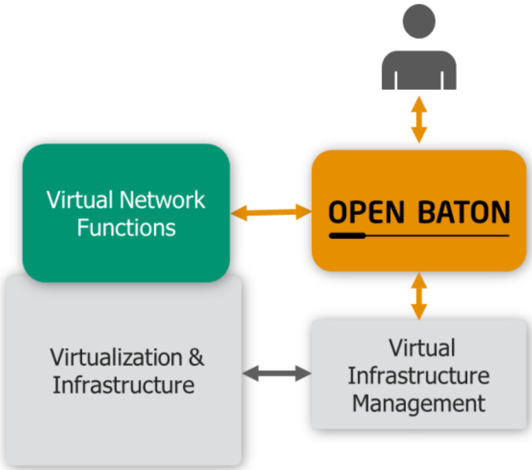
ONAP Architecture



Open Baton

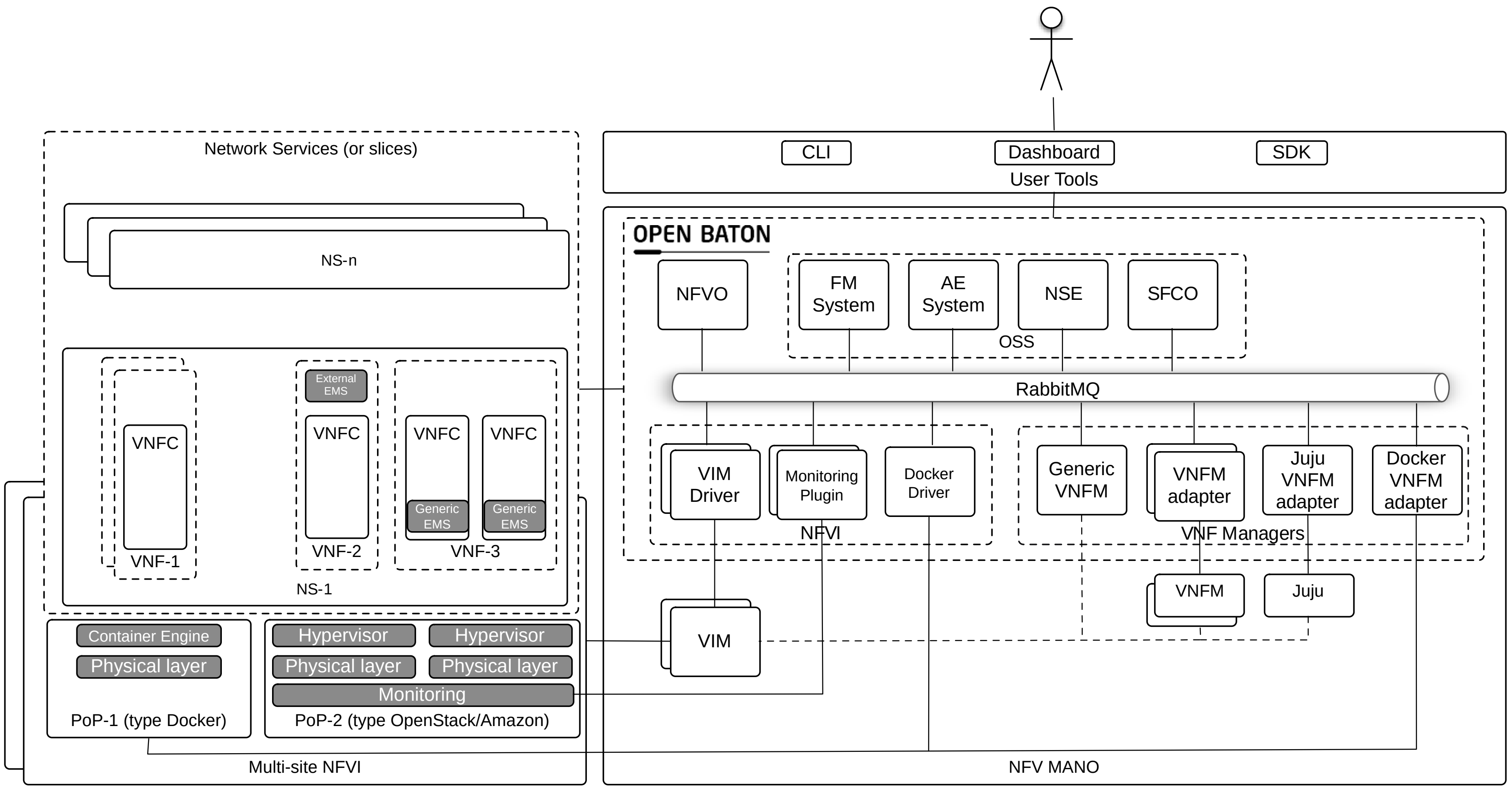
- Functionalities:
  - Installation, deployment and configuration of network services
  - Management of a multi-site NFV Infrastructures
  - Supports independent infrastructure slices
  - Includes service orchestration
  - Generic or specific VNF management
  - Runtime operations: fault management, autoscaling, etc.
  - Supports the execution of several use cases e.g. vEPC, vIMS, M2M and Multimedia communication

- Designed for answering Industry requirements
  - Easy to configure and to deploy
  - Providing a centralized view of the NFV Infrastructure



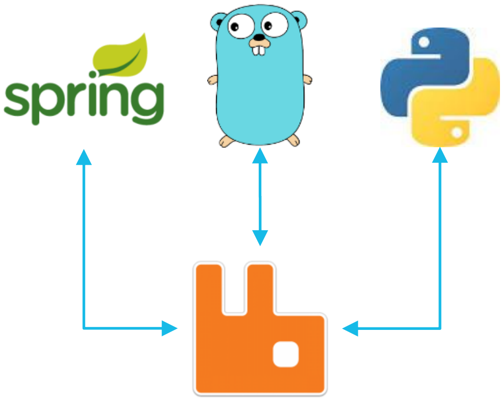
Available on github: https://github.com/openbaton

Open Baton Rel.5 Architecture



Open Baton in a nutshell

- Public GitHub repository: https://github.com/openbaton
  - Around 42 projects (to date)
- Public website: http://openbaton.github.io/
- Public documentation: http://openbaton.github.io/documentation/
- 2 Major releases per year
- RabbitMQ as message bus implementation
- Spring as framework and gradle as building tool for most of the Java components
- Three different set of SDKs languages: Java, Python and Go
- Dashboard implemented using AngularJS and Boostrap
- Installation with a single bootstrap command on ubuntu/debian
- Docker based environment for (almost) any kind of Operating System



MANO Comparison

- The table lists the main functionalities required for a MANO project:

- NFV Orchestration
- Service Orchestration
- Generic VNFM
- Multisite

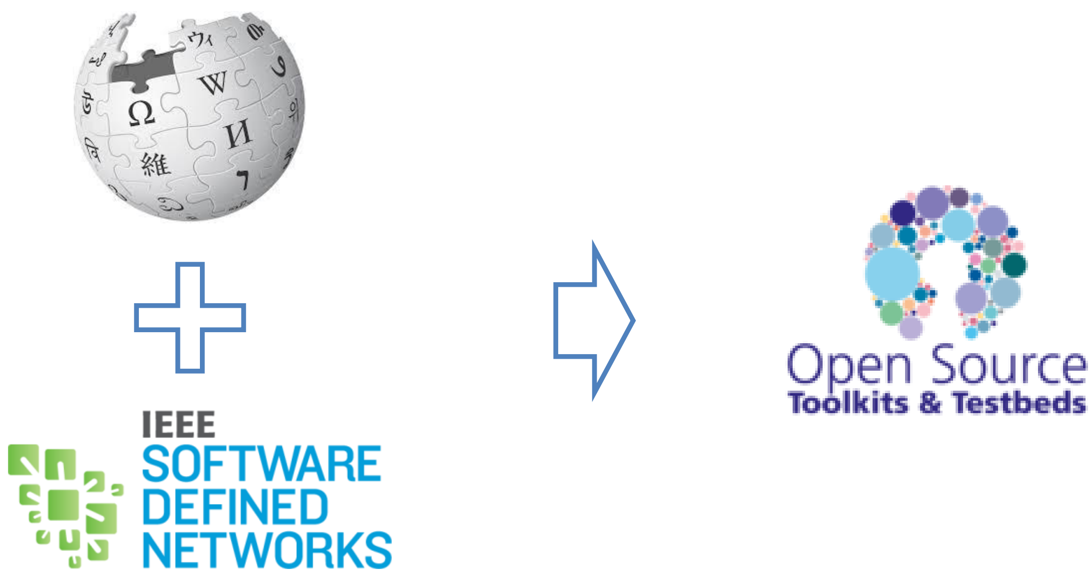
- It shows that most of the project already (or at least plan to) support those functionalities

	Tacker	ETSI OSM	Open Baton	Open-o
Community Governance	Y	Y	-	Y
Apache 2.0 license	Y	Y	Y	Y
Release	multiple	Rel 0	1st version	Not yet
NFVO	Y	Y	Y	Y
NFVO split: NSO/RO	-	Y	-	-
Generic VNFM	Y	Y	Y	Y
Specific VNFM support	Y	Y	Y	Y
OpenStack	Y	Y	Y	Y
Multiple OpenStack	Y	Y	Y	Y
Other VIM	Y	Y	Y	Y
TOSCA	Y	-	-	Y
Yang	-	-	-	Y
Dashboard	-	-	Y	Portal (?)
Service Orchestrator	-	Y	-	Y

Source: http://sdn.ieee.org/newsletter/july-2016/opensource-mano

IEEE SDN Catalogue

- An Open Catalogue of Open Source Toolkits and Testbeds



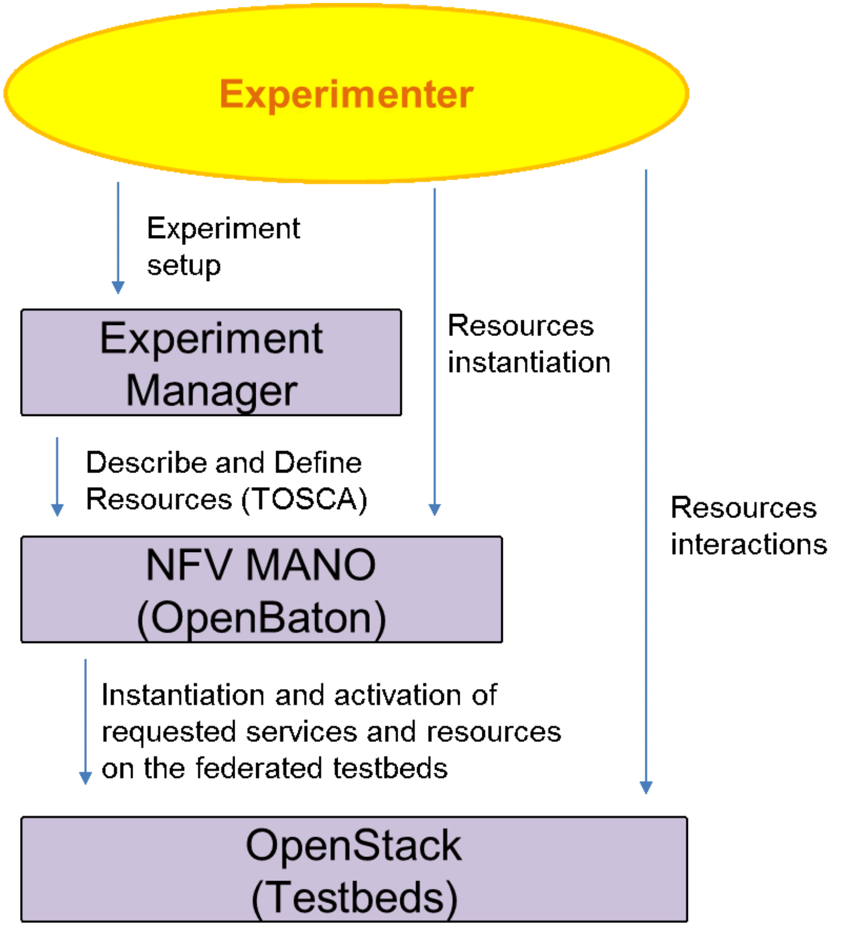
More info at: http://www.sdn-os-toolkits.org/

SoftFIRE High Level Architecture

It includes SoftFIRE software tools used by experimenters to interact with the Virtualization infrastructure

It exposes NFV Management and Orchestration (MANO) functionalities to the upper levels

It represents the collection of multiple interconnected testbeds and the OpenStack environment



The SoftFIRE Middleware

Software Middleware completely built from scratch (implemented in Python):

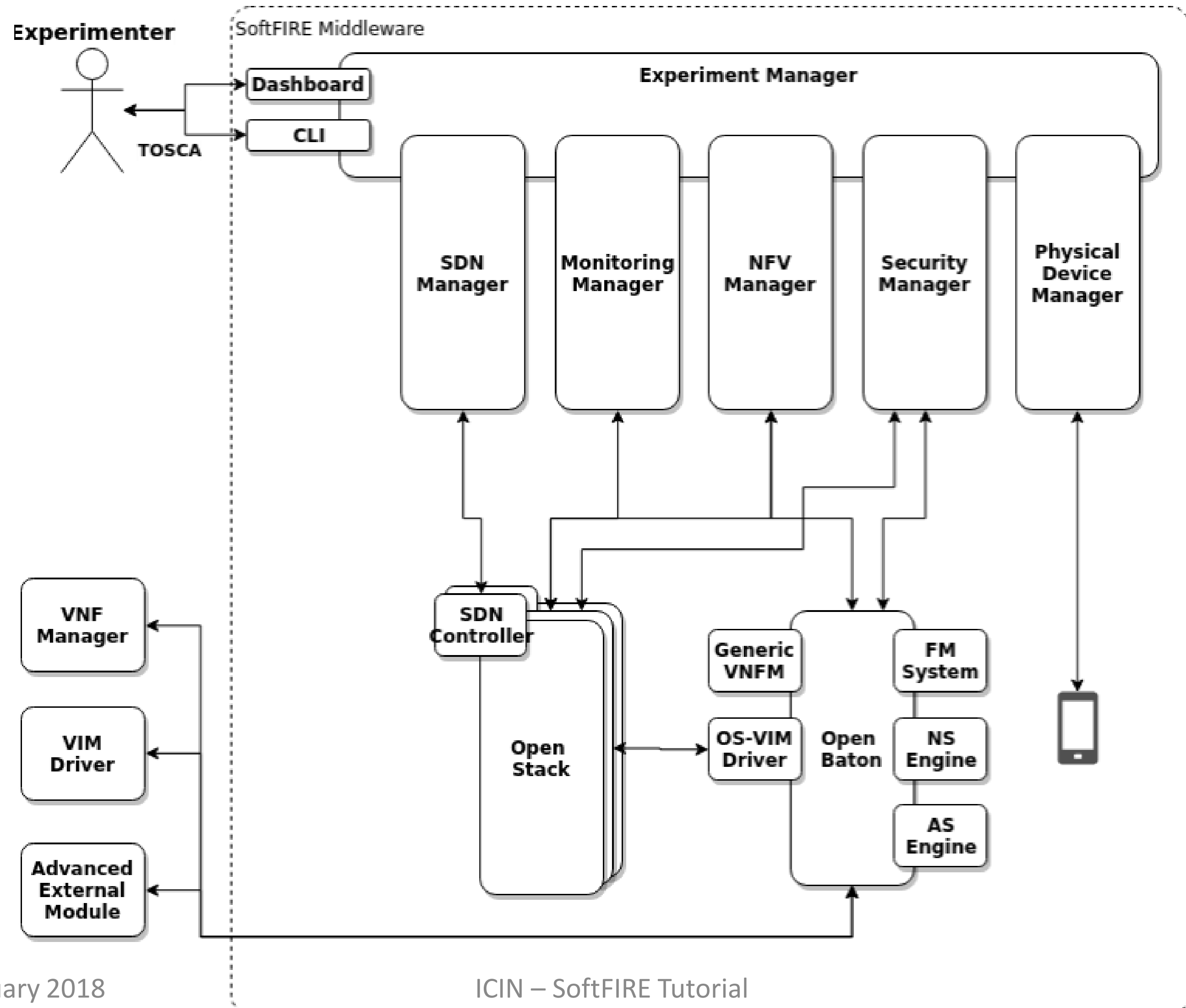
- Microservices-oriented architecture. The major component, the Experiment Manager acts as a broker between N different management entities in charge of specific parts of an experiment
  - Google RPC (gRPC) as messaging broker between the experiment manager and

Experiment Manager only access point to SoftFIRE for experimenters

- exposes TOSCA APIs
- delegates specific operations to the sub managers
- implements authentication and authorization
- generates certificate for SoftFIRE VPN for experimenters
- provides specific resource reservation
- lists all resources available
- deploys any kind of resources through the specific manager

https://github.com/softfire-eu/experiment-manager

The SoftFIRE Middleware Architecture



The SoftFIRE Middleware Architecture

Software Middleware completely built from scratch (implemented in Python):

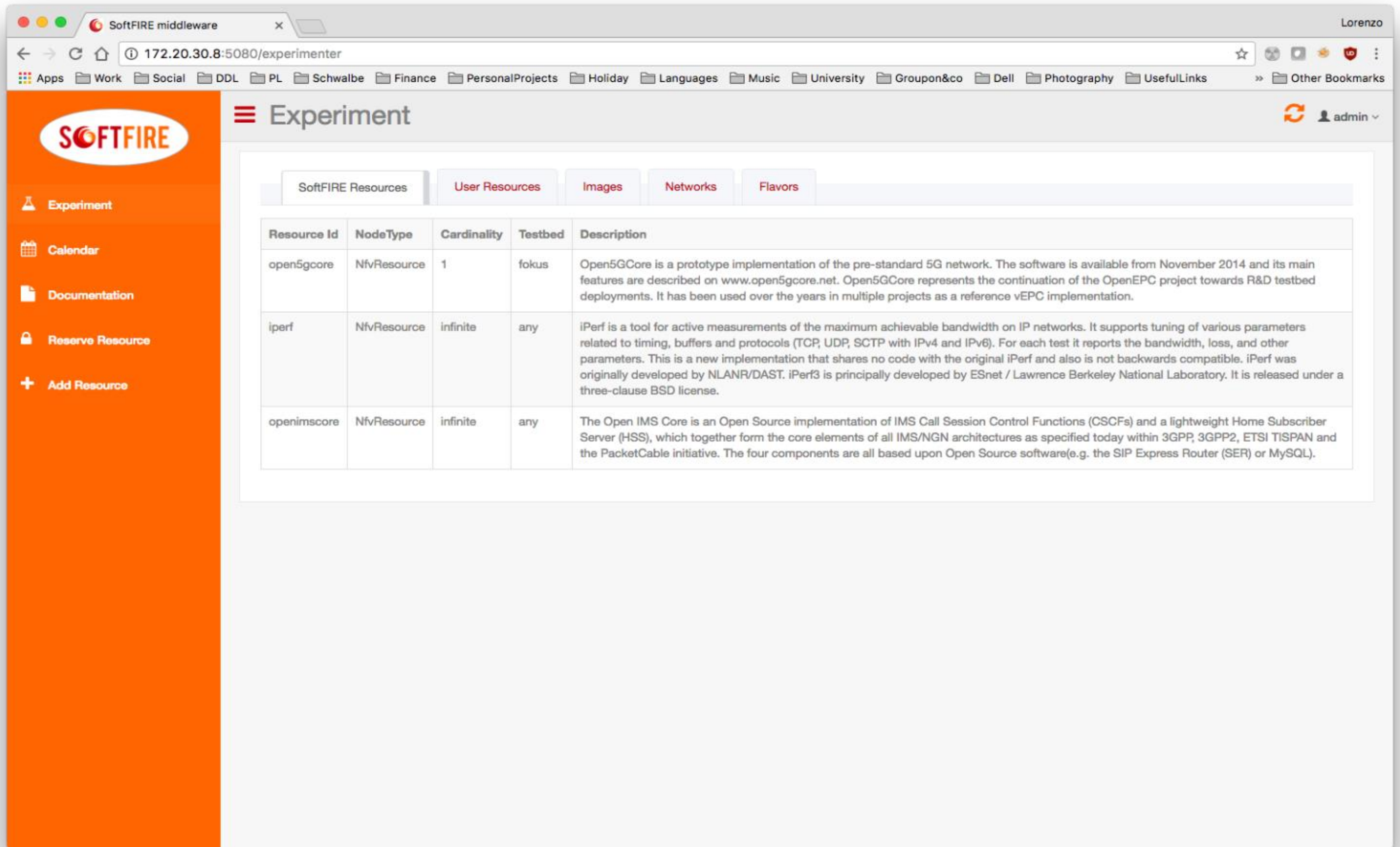
- Microservices-oriented architecture. The major component, the Experiment Manager acts as a broker between N different management entities in charge of specific parts of an experiment
  - Google RPC (gRPC) as messaging broker between the experiment manager and

Experiment Manager only access point to SoftFIRE for experimenters

- exposes TOSCA APIs
- delegates specific operations to the *sub* managers
- implements authentication and authorization
- generates certificate for SoftFIRE VPN for experimenters
- provides specific resource reservation
- lists all resources available
- deploys any kind of resources through the specific manager

https://github.com/softfire-eu/experiment-manager

The SoftFIRE Middleware Dashbaord

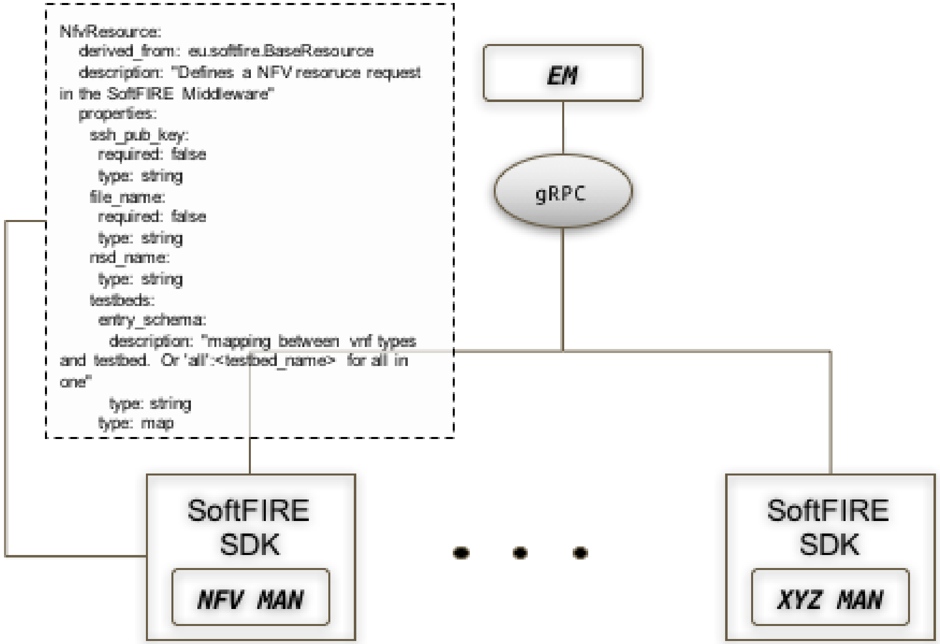


https://github.com/softfire-eu/views

Extensible and Customizable

Each Sub Manager implements a SDK and provides to the Experiment Manager (EM) the list of the kind of resources it is able to manage

- each resource is defined in a publicly available TOSCA definition file
- thus the EM can do the first TOSCA syntax validation
- the sub manager does the functional validation
- the EM sends the provide_resource message to the sub manager containing the resource to deploy and the sub manager executes specific logic based on the resource kind

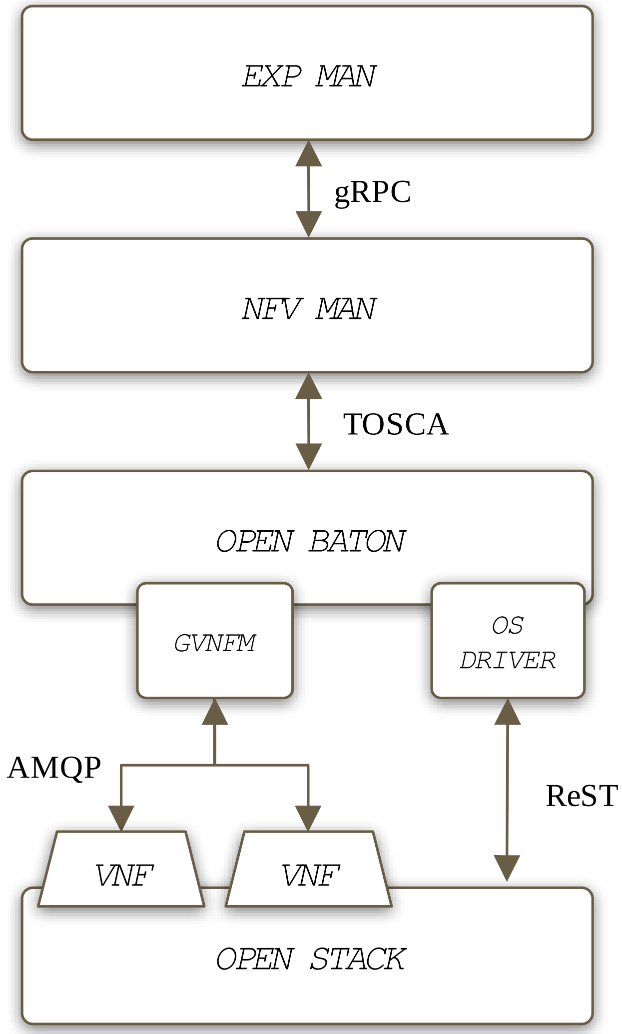


https://github.com/softfire-eu/softfire-sdk

NFV Manager

The NFV Manager integrates the NFV MANO system (Open Baton) within the system:

- In case of an experiment deployment it uploads a NSD to the NFVO and triggers the deployment.
- When the experiment is deleted it removes the NSR and NSD from the NFVO.
- The NFV Manager sends NSR status updates periodically to the EM.

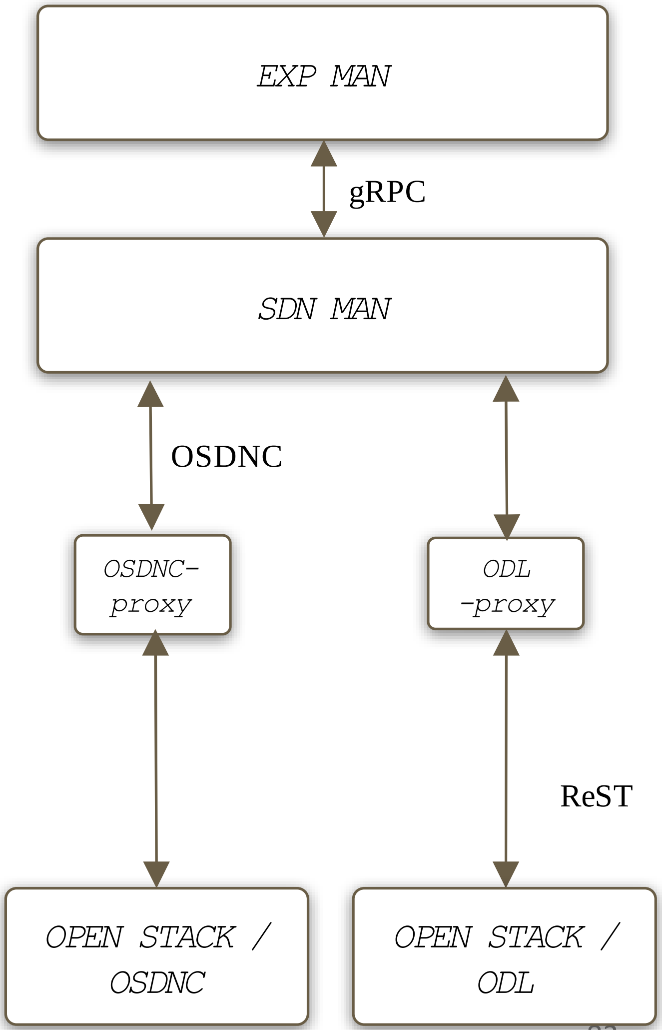


https://github.com/softfire-eu/nfv-manager

SDN Manager

The SDN Manager integrates SDN resources (i.e. the controller):

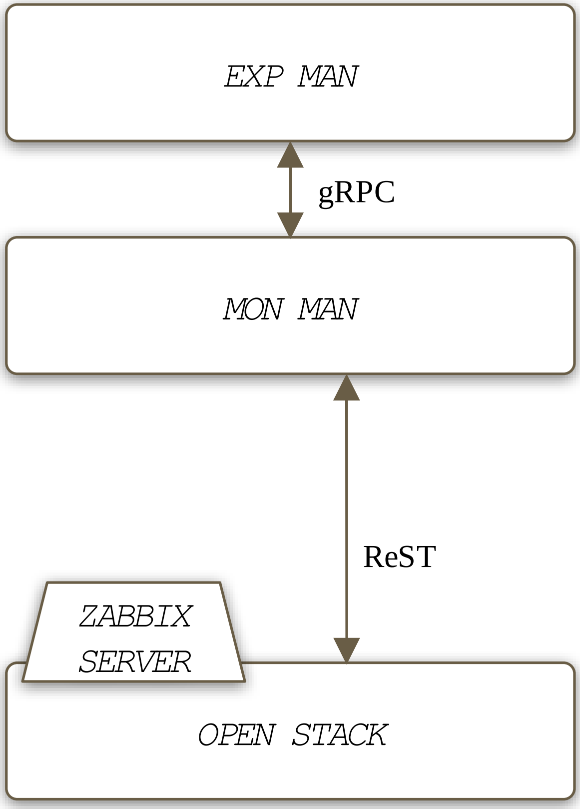
- Maintains a mapping between existing SDN resources and executing experiments
- Interact with the proxy to instantiate new tenants per experimenters



Monitoring Manager

The Monitoring Manager provides the capability to request to the SoftFIRE platform,

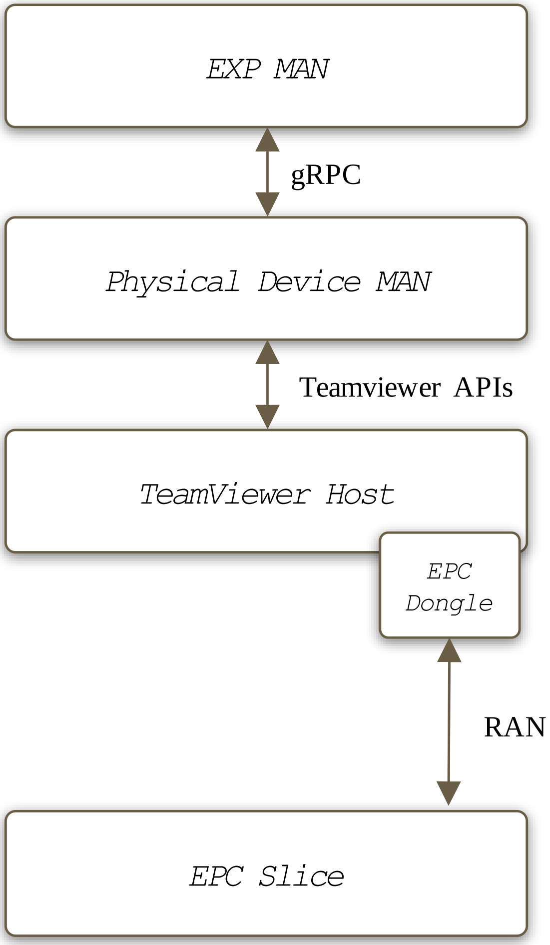
- The installation of a pre-configured Zabbix Server VM inside a specific experimenters' project.
- Furthermore, in coordination with the other components of the SoftFIRE middleware, it also installs a Zabbix Agent on each VM requested to be instantiated by a particular experimenter.



Physical Device Manager

The Physical Device Manager provides reservation of physical resources:

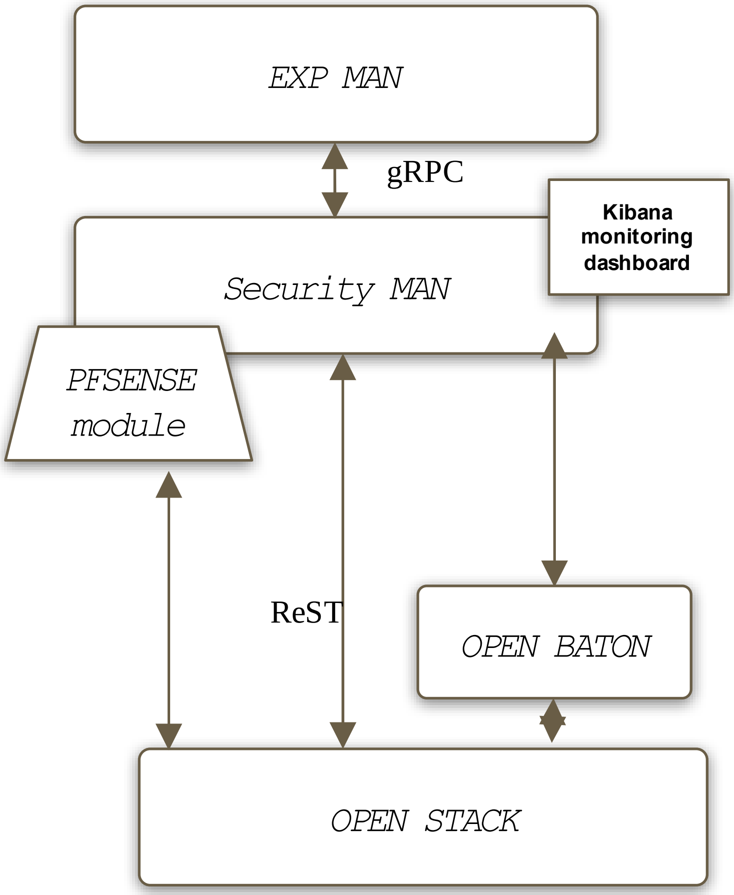
- Experimenters may reserve a UE and connect remotely via Teamviewer



Security Manager

The Security Manager provides security resources which can be used for experimenting with such technologies:

- Suricata
- PFSense



Agenda

- Use-case: Testing a ERP service
- Use-case implementation
- Live demo

19th February 2018 ICIN – SoftFIRE Tutorial 87

Part 1: From the Design to the Implementation: an example on a federated testbed

Enterprise Resource Planning (ERP) Service

Front-end

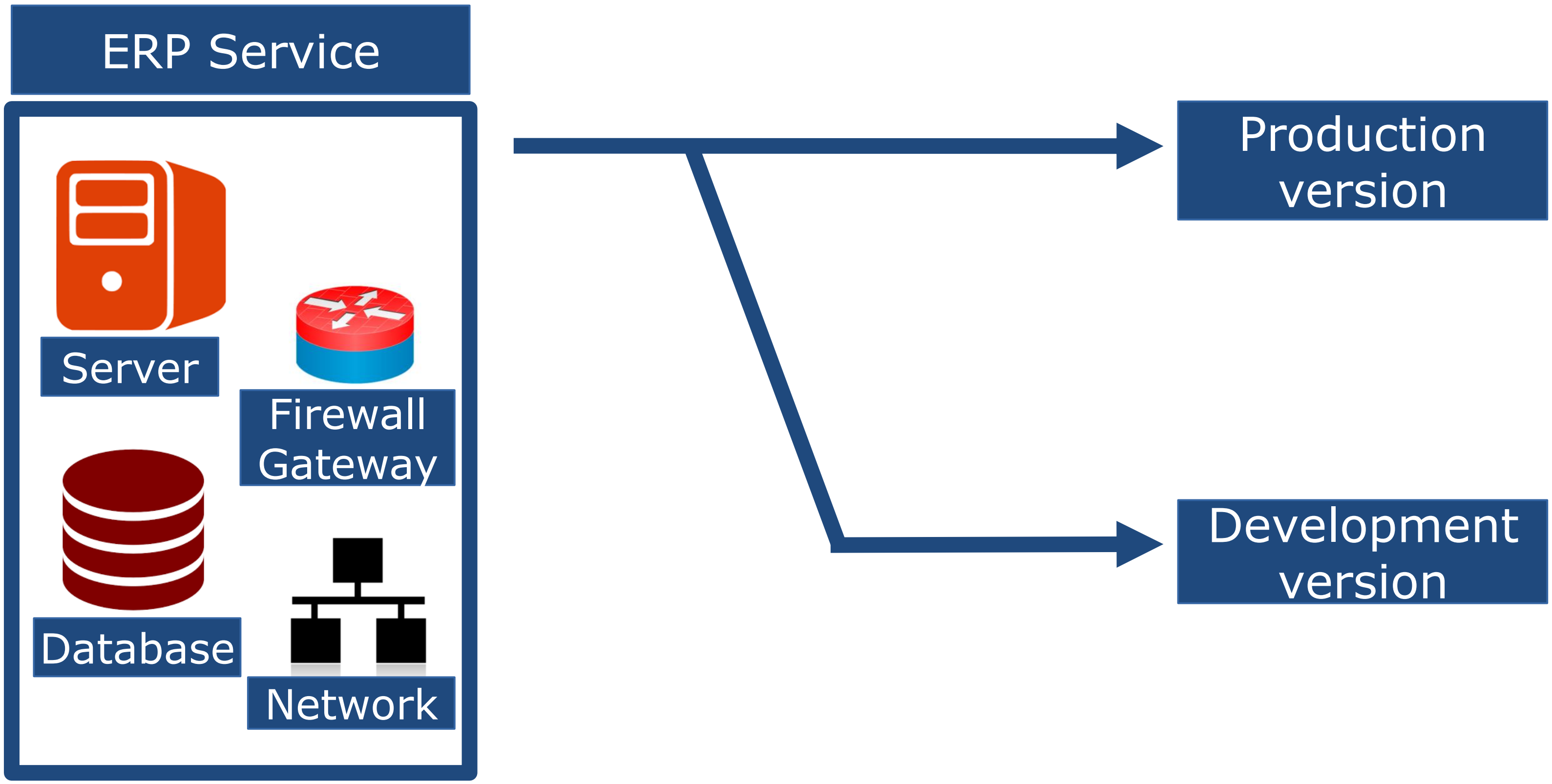
Back-end

End users interact with the front end

Front-end makes the ERP service available (e.g. with a html page)

Back-end manages ERP data and ancillary functions

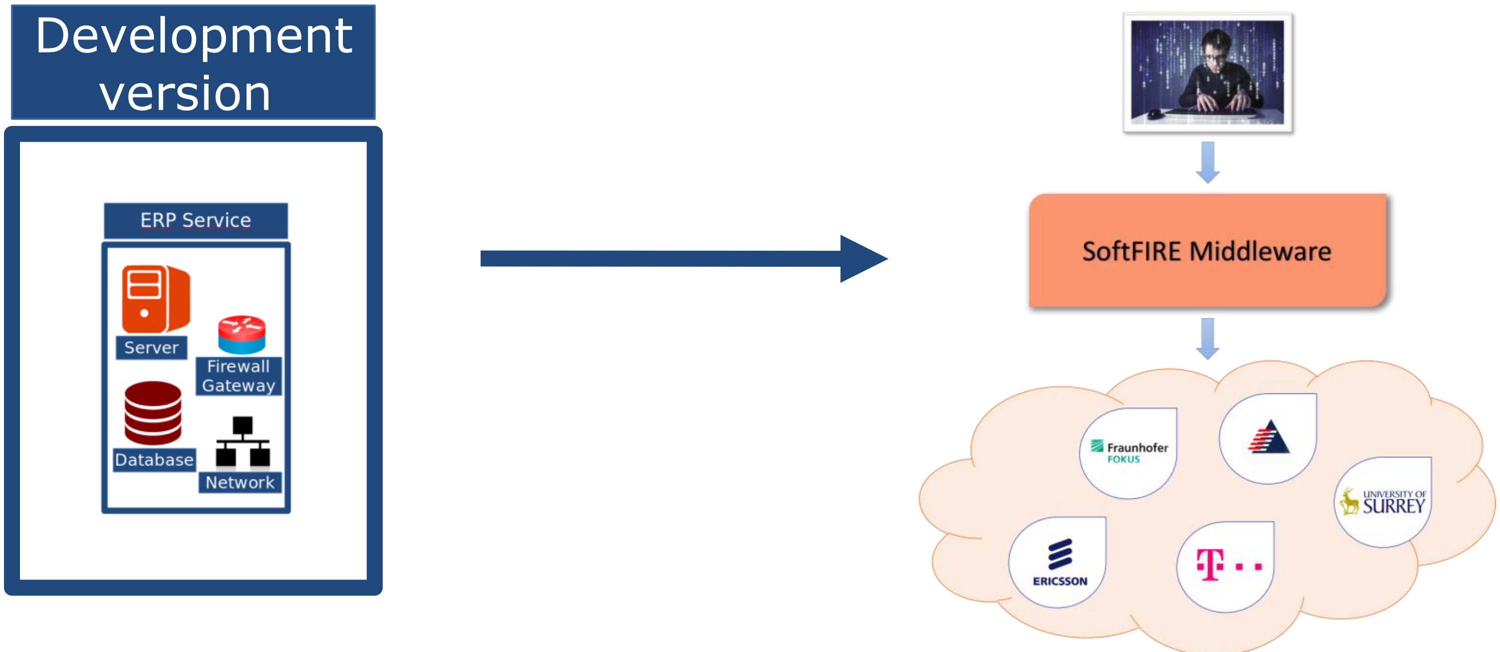
ERP service


ERP Service
Server
Firewall Gateway
Database
Network
Production version
Development version

Testing

What we need:

- A testing environment that simulates the production environment.
- A fast and easy way to setup the environment in order to focus on the service development itself.
- The testing environment should be isolated from the production environment.



Using SoftFIRE: Benefits

- Adaptability
- Easy to specialize
- On-demand provisioning

Adaptability

Starting from the project requirements, it is possible to develop your own Network Service that describes the network topology, the virtual machines and the software to be installed.

SoftFIRE is able to elaborate the service description and deploy the Network Service.

You do not have to worry about how it is deployed.

Easy to specialize

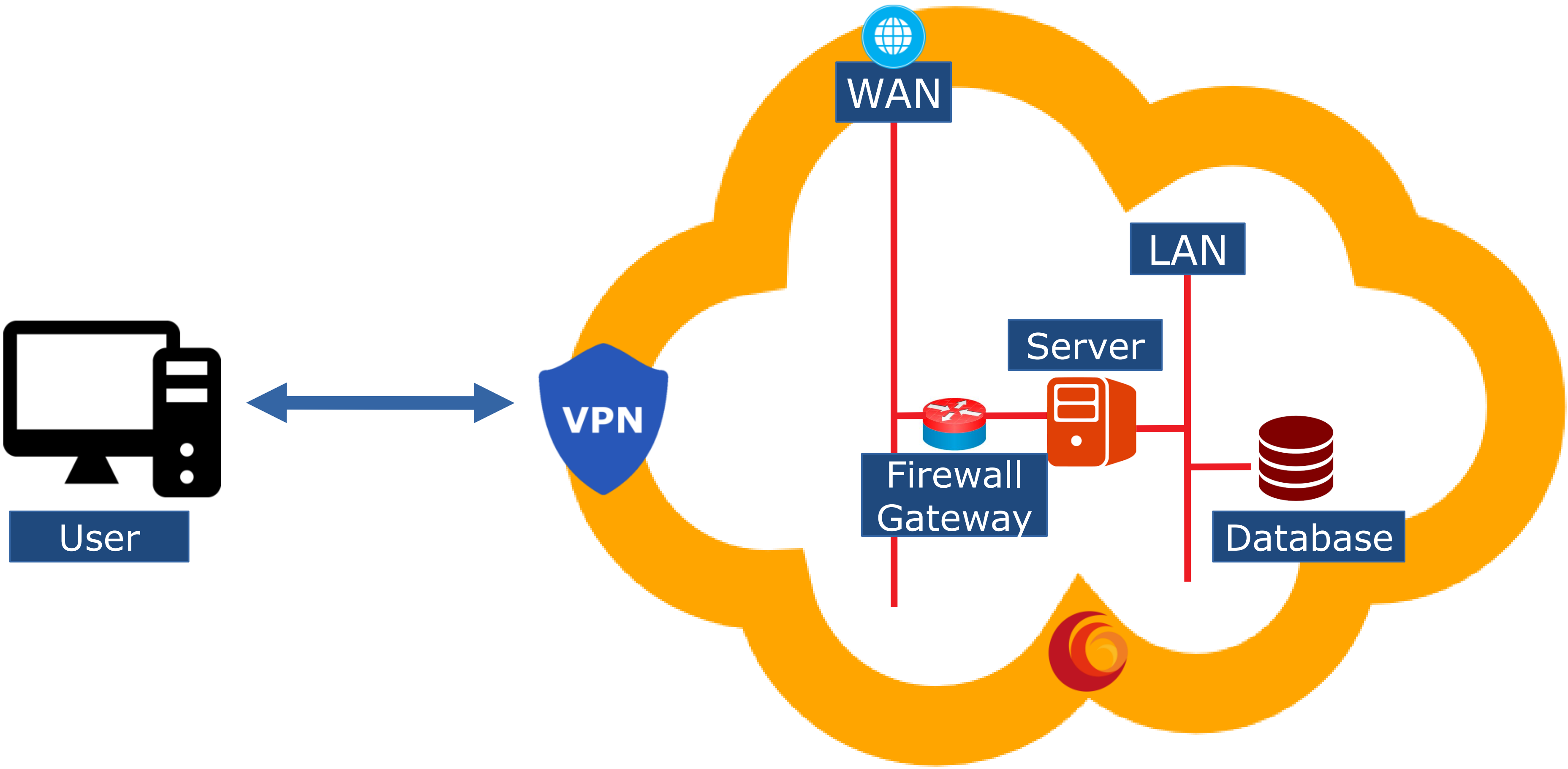
You can implement your own managers in order to deploy a specialized Network Service (e.g. ERP frontend NFV manager).

On-demand provisioning

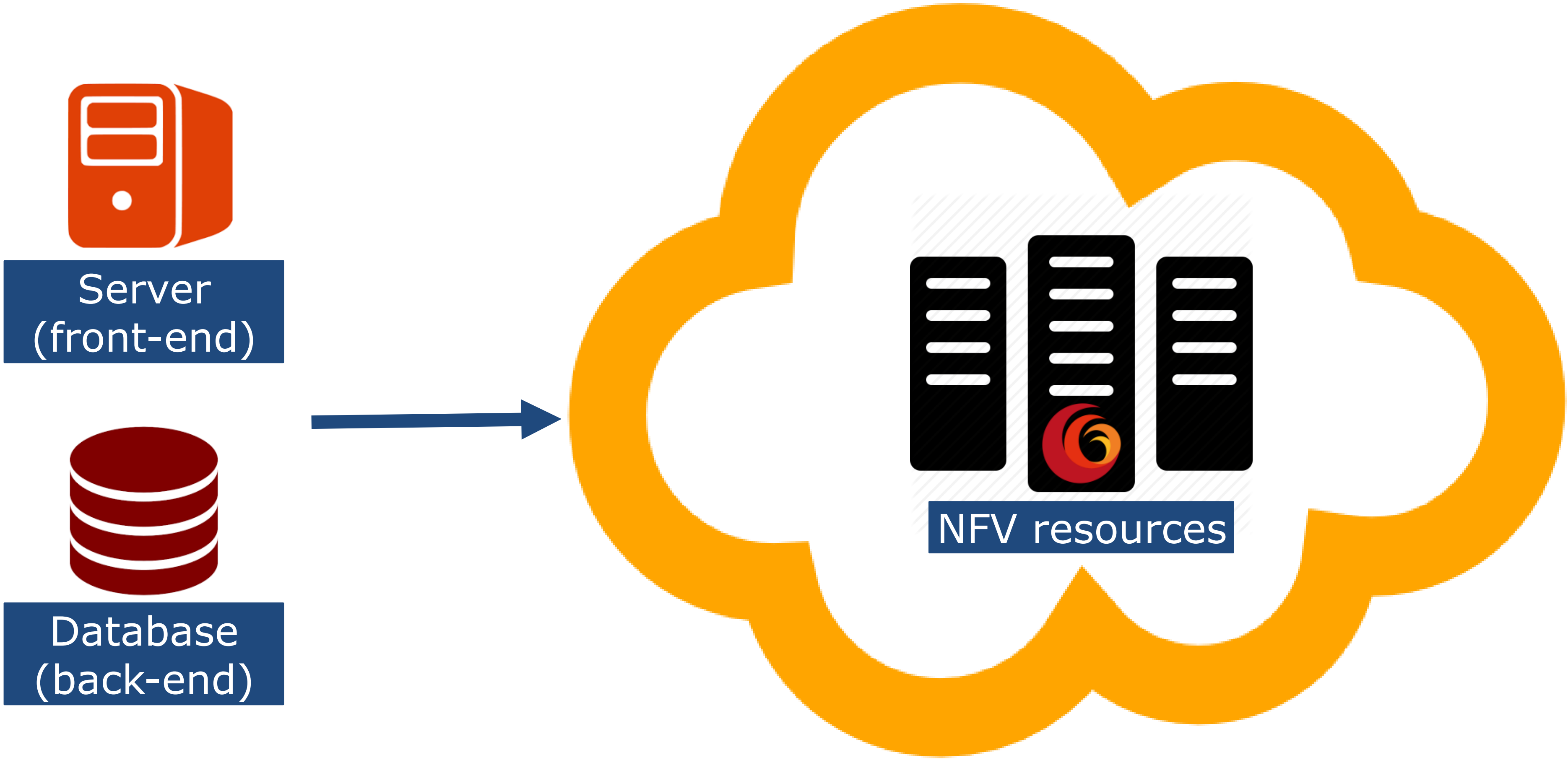
Your service can be easily scaled out requesting to SoftFIRE more resources. This way, you are able to perform stress test the service.

Use-case implementation

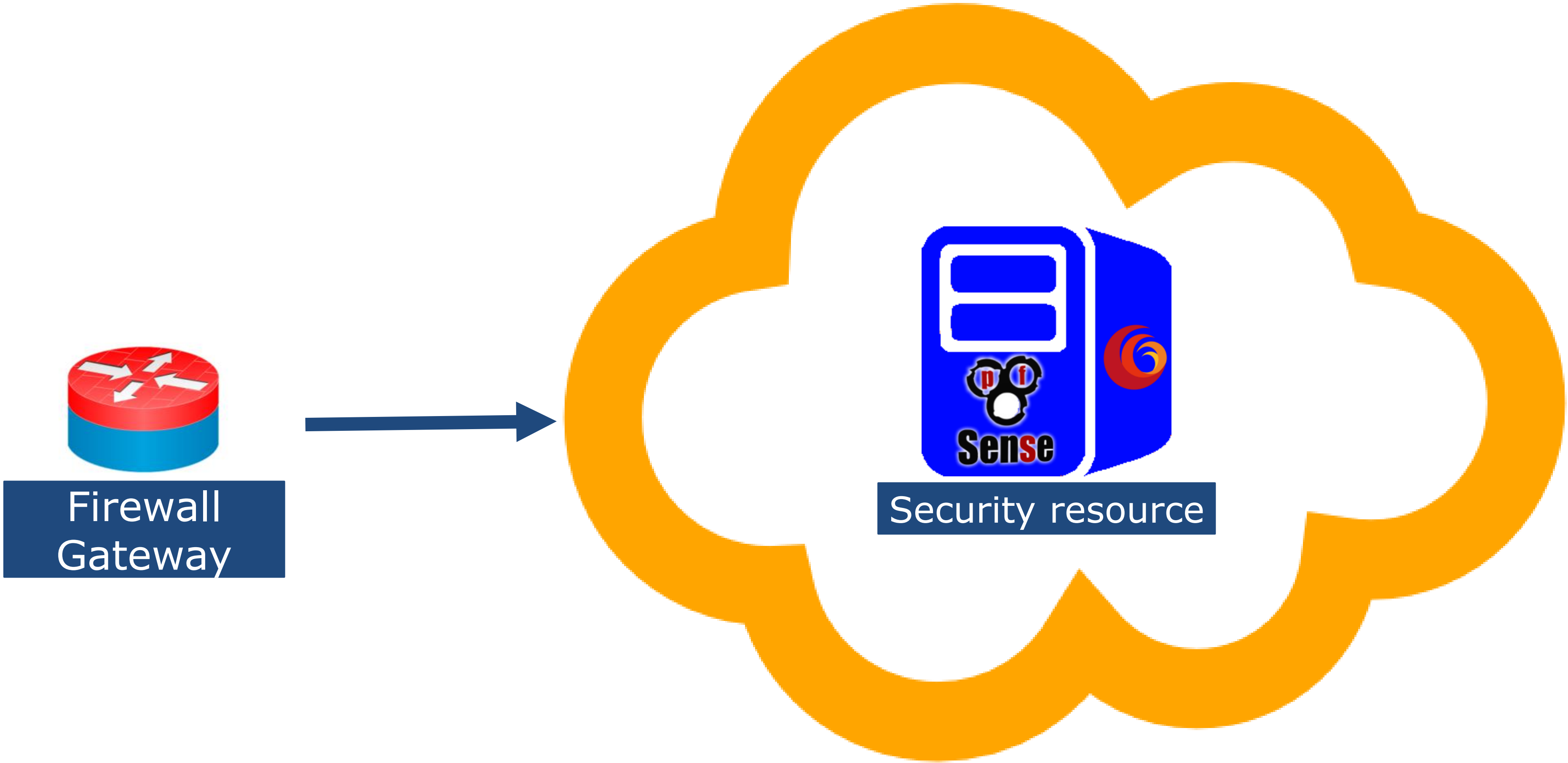
ERP service topology



SoftFIRE resources



SoftFIRE resources



Define the use-case

Language: TOSCA (Topology and Orchestration Specification for Cloud Applications)

CSAR
(Cloud Service ARchive)

```
|— Definitions
|       └── service.yaml
|— Files
|       └── dev-service_nsd.csar
|— TOSCA-Metadata
        ├── Metadata.yaml
        └── TOSCA.meta
```

.CSAR content

```yaml
description: "Testing a service inside federated testbed NFV"
imports:
    - softfire_node_types: "https://......./softfire_node_types.yaml"
topology_template:
    node_templates:
        client-server_nsd:
            type: NfvResource
            properties:
                resource_id: dev-service
                testbeds:
                    ANY: surrey
                nsd_name: "dev-service_nsd"
                ssh_pub_key: "ssh-rsa ..."
                file_name: Files/dev-service_nsd.csar
        pfsense-gw:
            type: SecurityResource
            properties:
                resource_id: pfsense
                lan_name: testing_lan
                wan_name: wan
                testbed: surrey
tosca_definitions_version: tosca_simple_yaml_1_0
```

Definitions/service.yaml

.CSAR content

TOSCA-Metadata

```
name: ICIN Tutorial
start-date: "2018-02-06"
end-date: "2018-02-22"
```

Metadata.yaml

```
TOSCA-Meta-File-Version: 1.0
CSAR-Version: 1.1
Created-By: SoftFIRE
Entry-Definitions: Definitions/service.yaml
```

TOSCA.meta

Files

dev-service_nsd.csar

Custom Network Service Descriptor front-end and back-end

Custom Network Service

dev-service_nsd.csar

CSAR (Cloud Service ARchive)

```
|— Definitions
|     └— dev-service_nfv.yaml
|— Scripts
|     ├— server
|     |     ├— install_server.sh
|     |     ├— backend_configure.sh
|     |     └— start_server.py
|     └— backend
|          └— install.sh
|— TOSCA-Metadata
      ├— Metadata.yaml
      └— TOSCA.meta
```

TOSCA-Metadata

```yaml
name: dev-service_nsd
description: NSD containing frontend and backend VNF
provider: SoftFIRE
image:
    upload: false
    names:
        - Ubuntu-16.04
vim_types:
    - openstack
```

Metadata.yaml

```
TOSCA-Meta-File-Version: 1.0
CSAR-Version: 1.1
Created-By: SoftFIRE
Entry-Definitions: Definitions/dev-service_nfv.yaml
```

TOSCA.meta

dev-service_nfv.yaml

```yaml
backend:
    type: openbaton.type.VNF
    properties:
        ID: backend
        vendor: SoftFIRE
        version: 1.0
        endpoint: generic
        type: backend
        deploymentFlavour:
          - flavour_key: m1.small
    requirements:
        - virtualLink: testing_lan
        - vdu: VDU1
    interfaces:
        lifecycle:
            INSTANTIATE:
                - install.sh
server:
    type: openbaton.type.VNF
    properties:
        ID: server
        vendor: SoftFIRE
        version: 1.0
        endpoint: generic
        type: server
        deploymentFlavour:
          - flavour_key: m1.small
    requirements:
        - virtualLink: testing_lan
        - vdu: VDU2
    interfaces:
        lifecycle:
            INSTANTIATE:
                - install_server.sh
            CONFIGURE:
                - backend_configure.sh
```

Nodes template

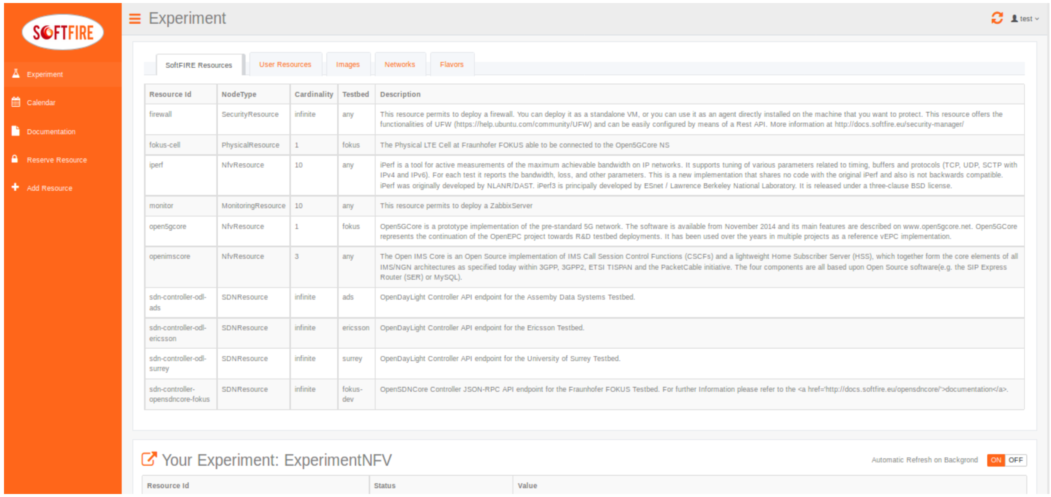
```yaml
relationships_template:
    rel_backend-server:
        parameters:
            - testing_lan
        source: backend
        target: server
        type: tosca.nodes.relationships.ConnectsTo
```

Relationships template

Upload the use-case

Now we can archive everything as a .csar archive

ERP_service.csar



Live Demo: https://www.youtube.com/watch?v=hehzsSOKGok


Q
A