Demonstration of MEC Applications deployment with OSM

Open Source MANO

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Davide Borsatti
Università di Bologna

whitestack

Sergio Tarazona
Whitestack

Table of Contents

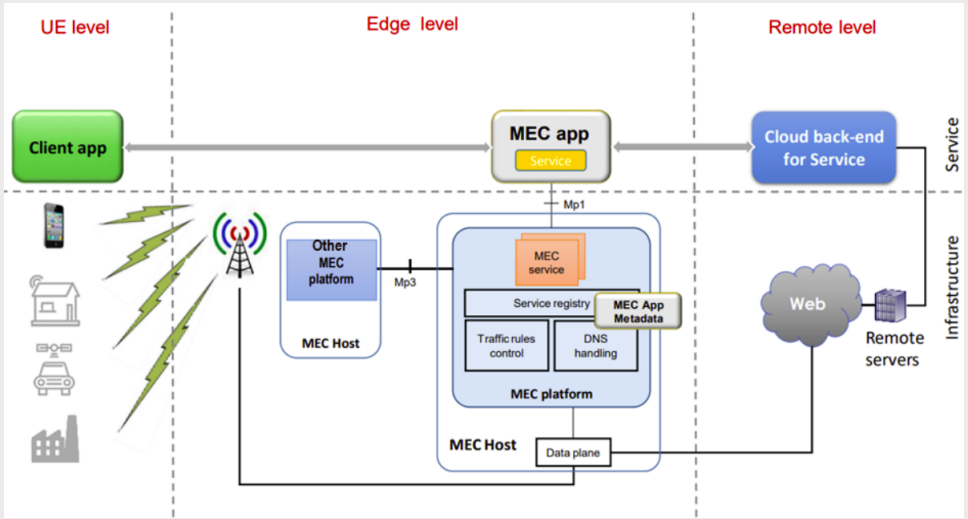
1: MEC Principles

Network Modularization

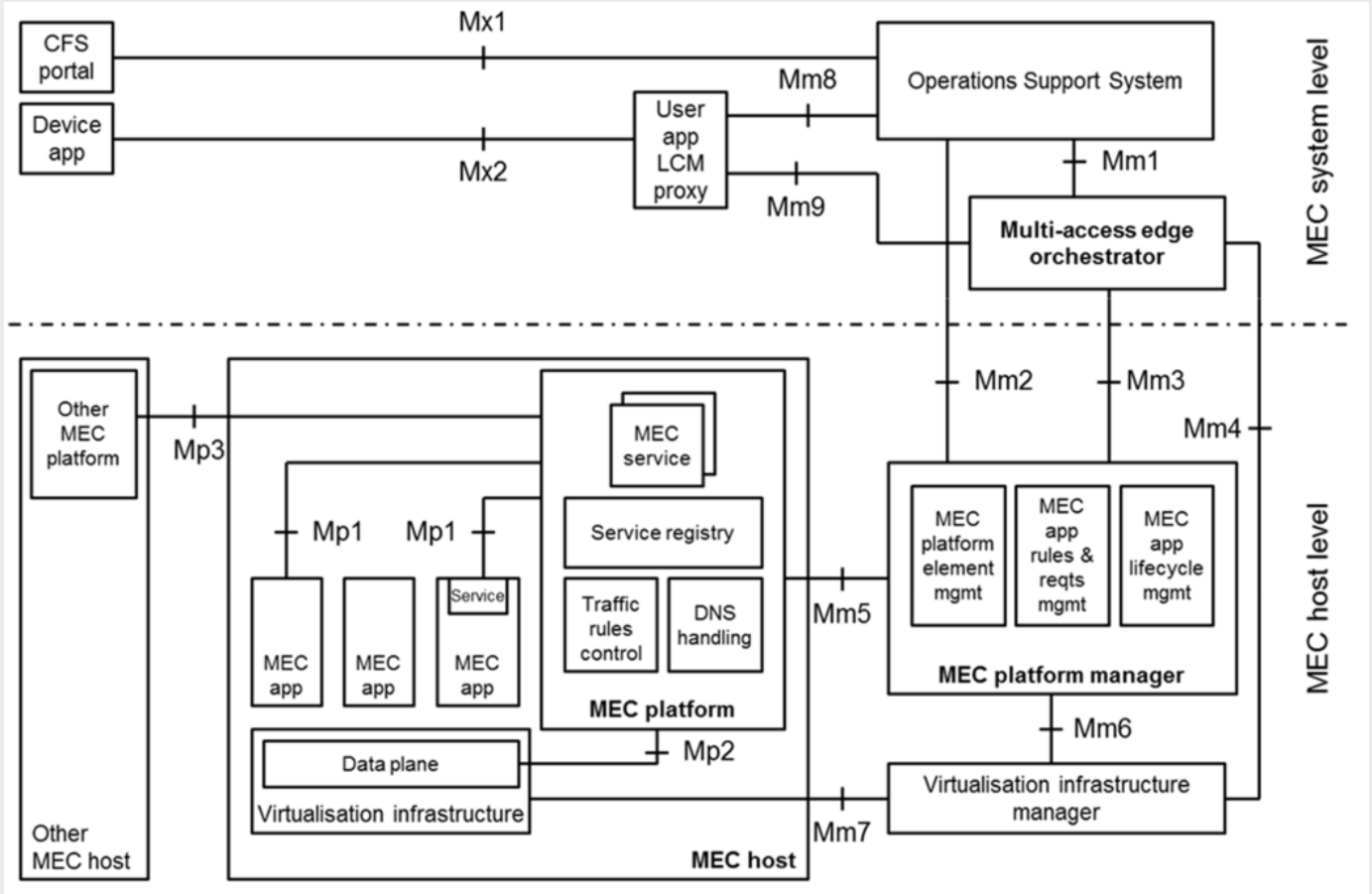
Multi-Access Edge Computing

Multi-access Edge Computing (MEC) offers application developers and content providers cloud-computing capabilities and an IT service environment at the edge of the network. This environment is characterized by ultra-low latency and high bandwidth as well as real-time access to radio network information that can be leveraged by applications.

MEC provides a new ecosystem and value chain. Operators can open their Radio Access Network (RAN) edge to authorized third-parties, allowing them to flexibly and rapidly deploy innovative applications and services towards mobile subscribers, enterprises and vertical segments.

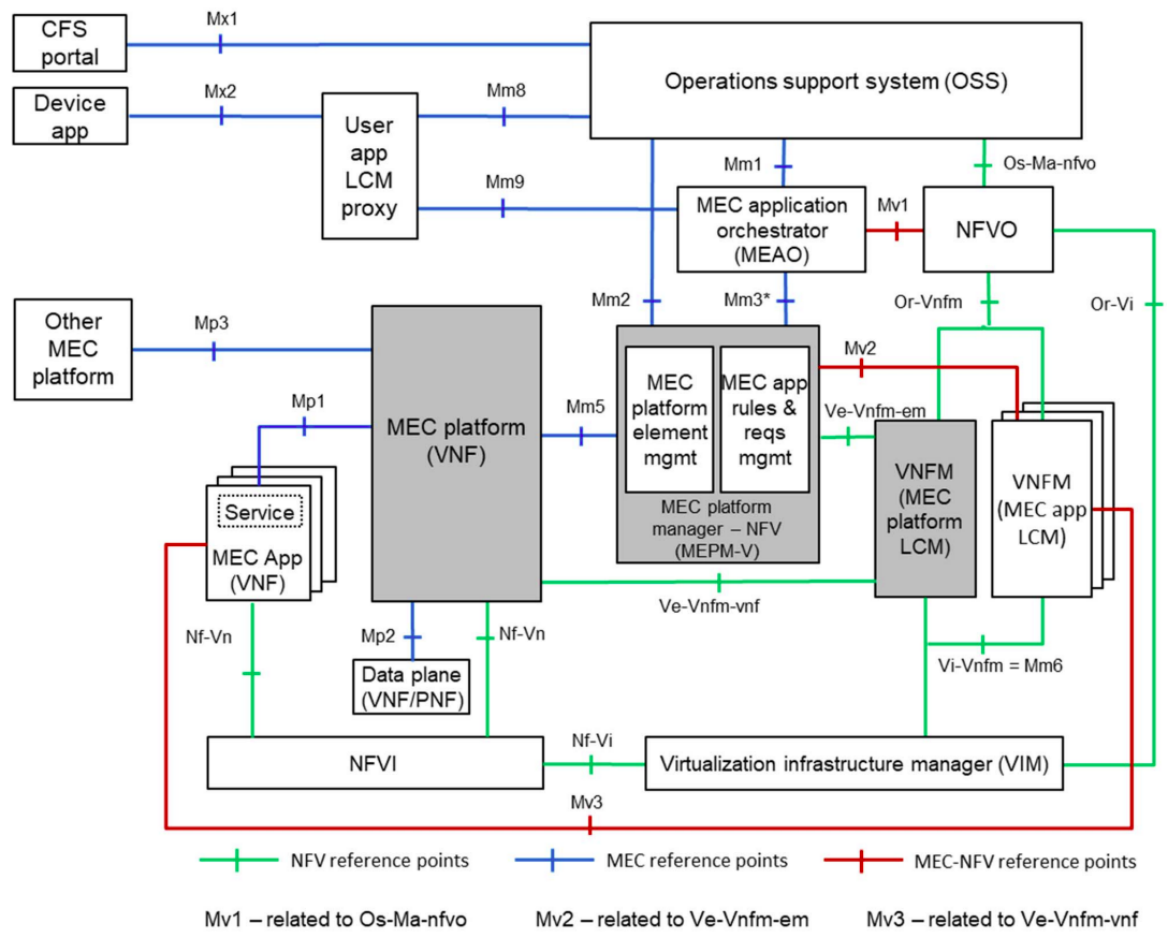


Multi-Access Edge Computing Architecture



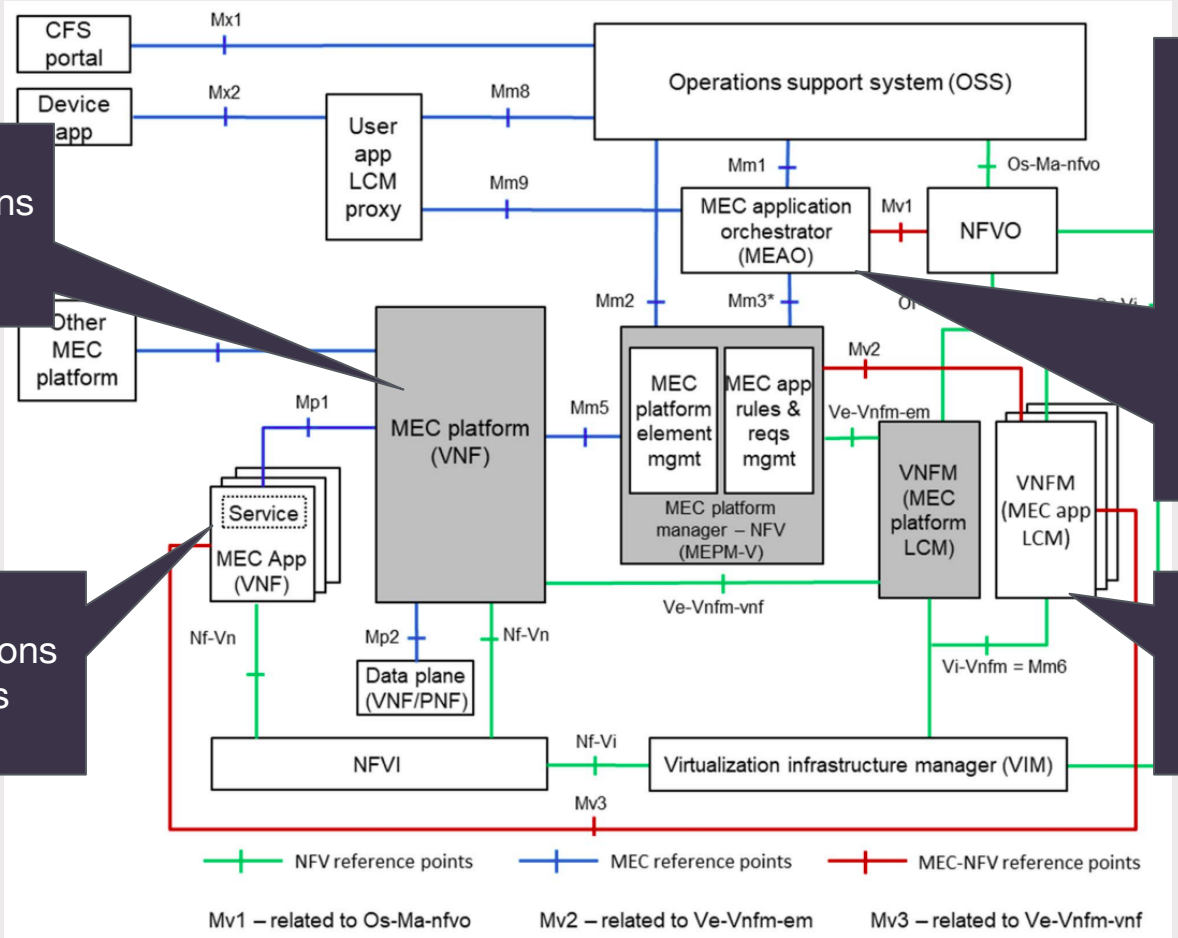
From ETSI GS MEC 003 V2.2.1

MEC-in-NFV Architecture



From ETSI GS MEC 003 V2.2.1

MEC-in-NFV Architecture



MEC Platform runs as a VNF

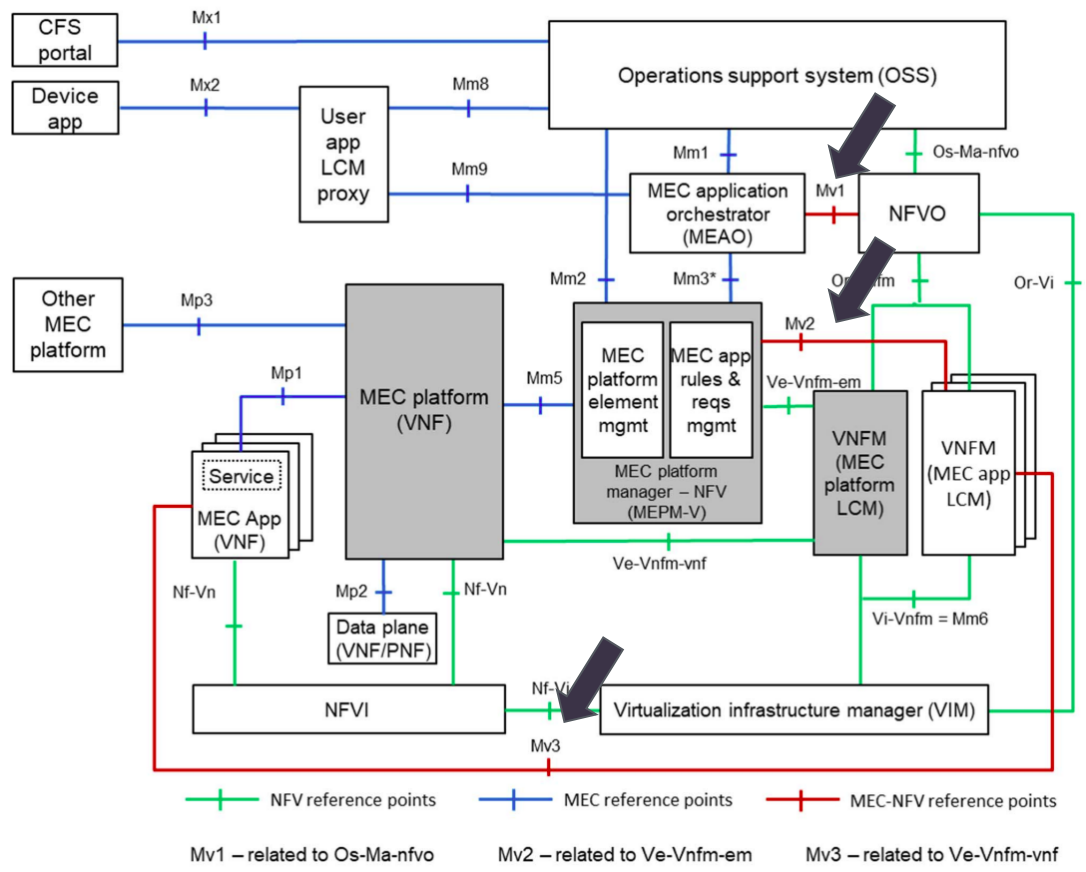
Management of MEC apps delegated to NFVO which manages these as part of one or more NFV network services.

MEC Applications run as VNFs

MEC Applications lifecycle managed by their VNFM

From ETSI GS MEC 003 V2.2.1

MEC-in-NFV Architecture



3 “Hybrid” Reference points:

- **Mv3**: at this point no specific changes to Ve-Vnfm-vnf are expected (i.e. it can be used as is)
- **Mv2**: Necessary changes are being addressed by NFV IFA as part of FEAT12 work (MECinNFV)
- **Mv1**: work identified, coordination plan is on-going

MEC descriptor (AppD) must be linked to NFV descriptor (VNFD). This has been addressed as part of Rel 3 work using Non-MANO artifact capability as defined in Annex B of ETSI GS NFV-SOL 004 v. 2.5.1 and higher.

From [MEC Public Overview]

MEC 011 interactions

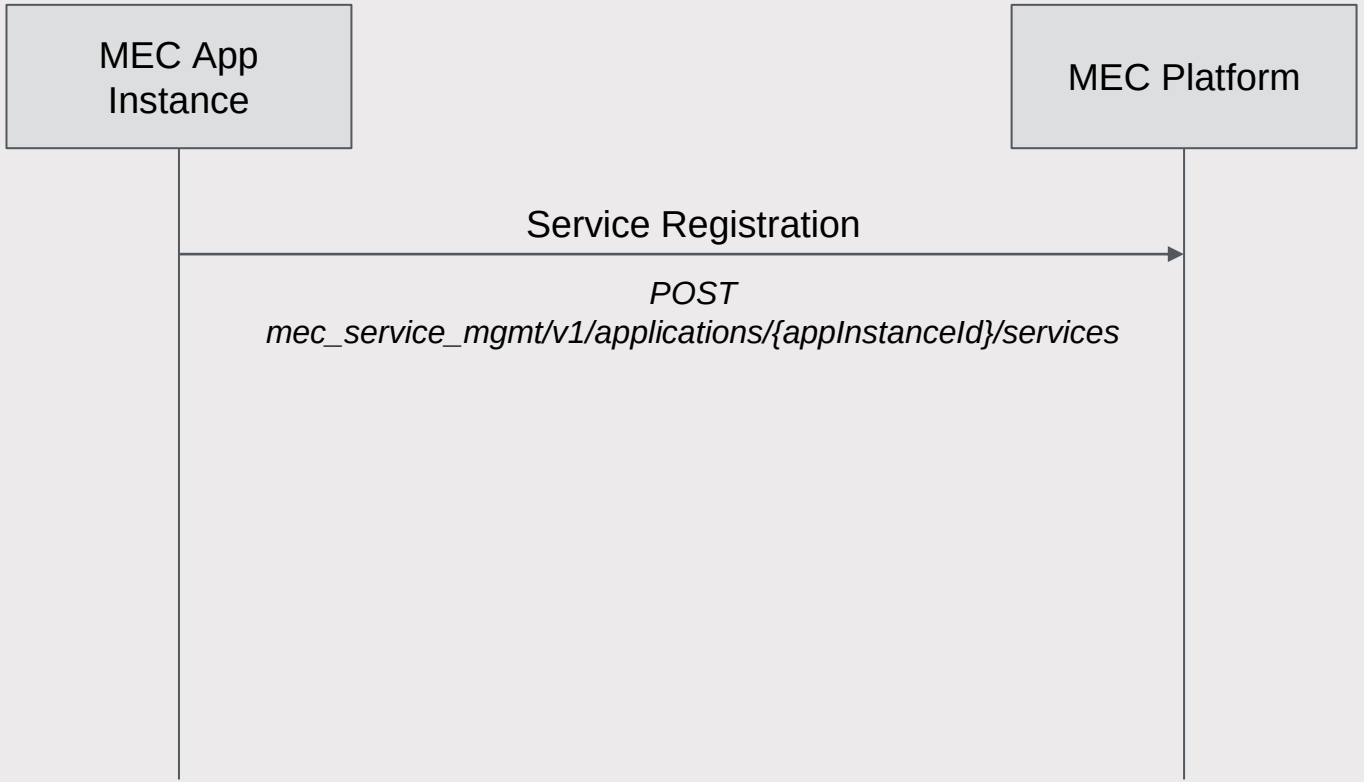
“The MEC platform, as defined in ETSI GS MEC, offers an environment where MEC applications may discover, advertise, consume and offer MEC services. Upon receipt of update, activation or deactivation of traffic rules from the MEC platform manager, applications or services, the MEC platform instructs the data plane accordingly. The MEC platform also receives DNS records from the MEC platform manager and uses them to configure a DNS proxy/server.”

From ETSI GS MEC 011 V2.2.1

MEC 011 interactions

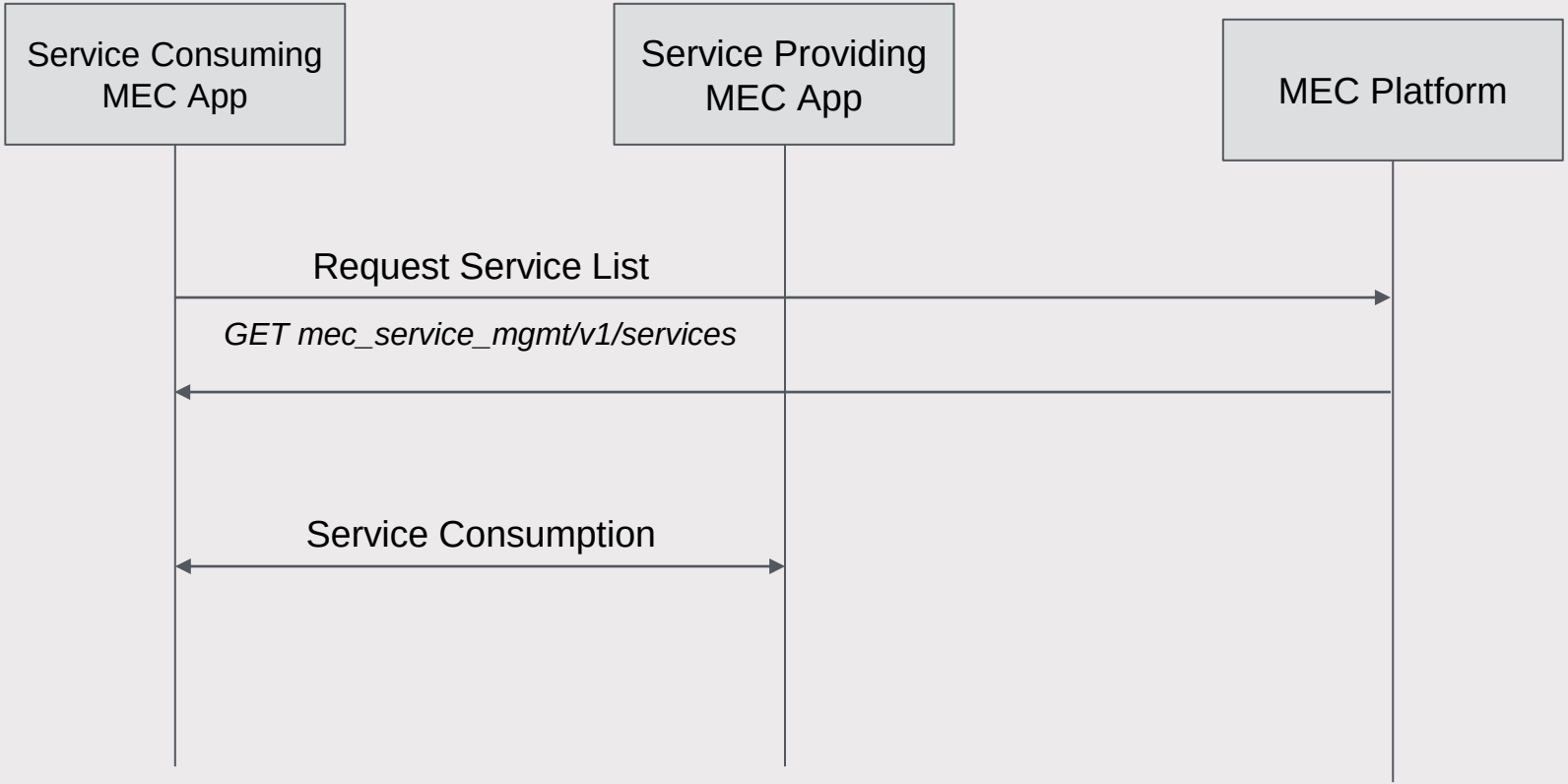
From ETSI GS MEC 011 V2.2.1

MEC 011 interactions



From ETSI GS MEC 011 V2.2.1

MEC 011 interactions



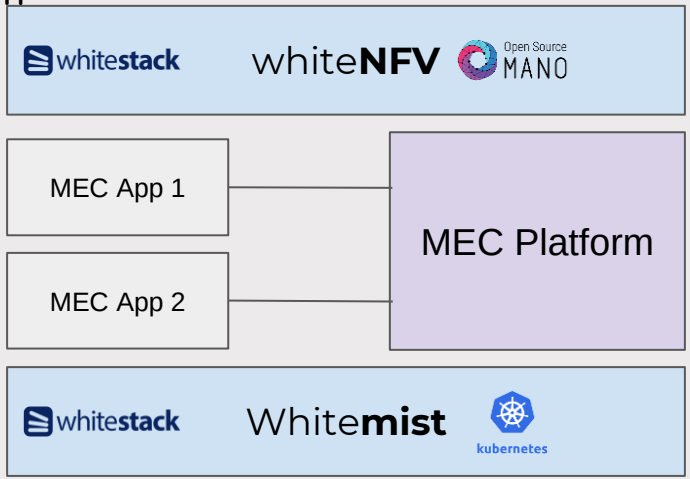
From ETSI GS MEC 011 V2.2.1

2: MEC demo using WhiteNFV

MEC demo

We’re going to test MEC-platform using MEC Apps designed by Università di Bologna:

- WhiteNFV will be used to orchestrate the MEC platform and apps.
- Whitemist will be used as kubernetes environment
- We will test interactions based on ETSI standards:
  - ✓ MEC App is running
  - ✓ Service registration
  - ✓ Request service list
  - ✓ Service consumption

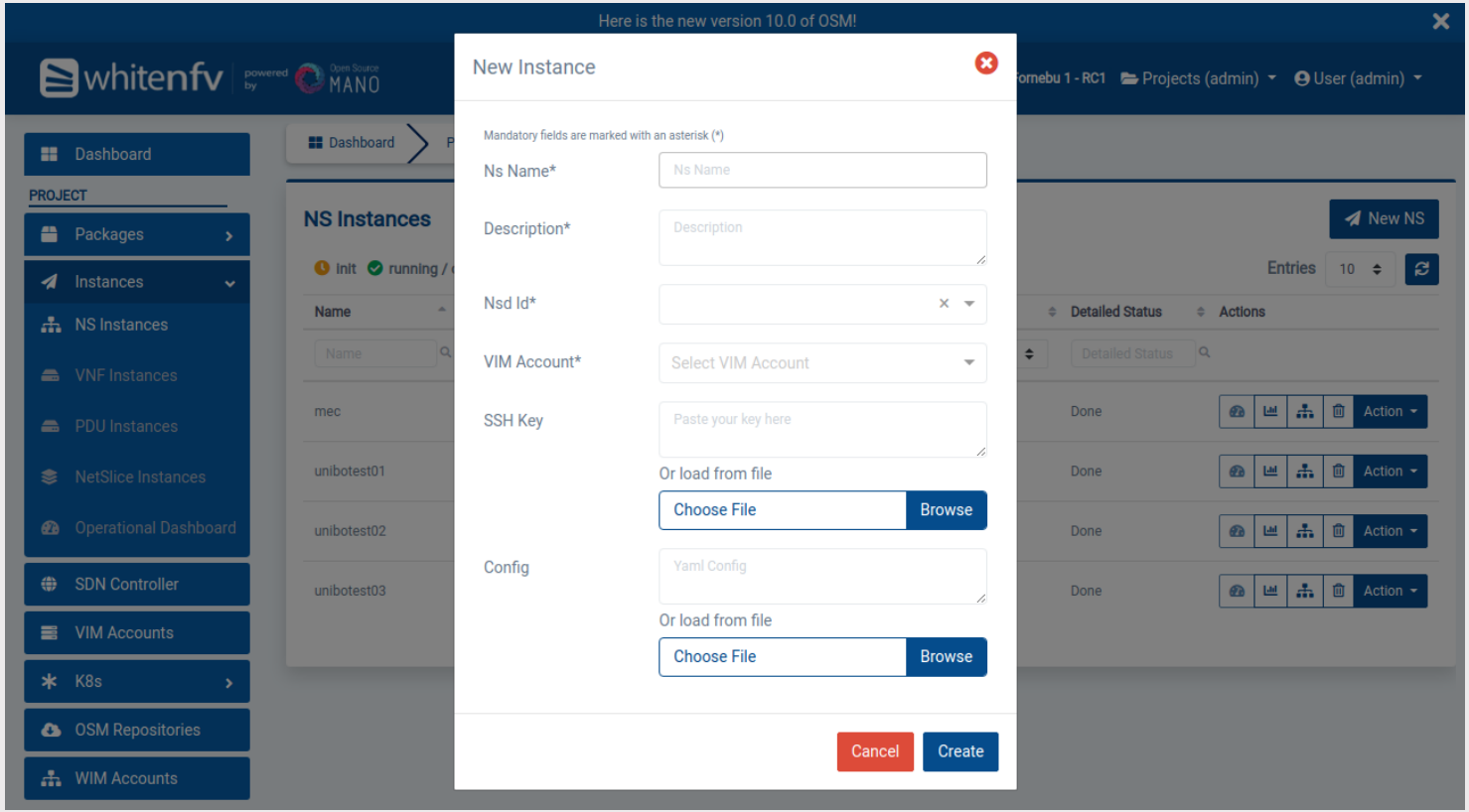


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

https://github.com/DendoD96/unibo-mec-platform

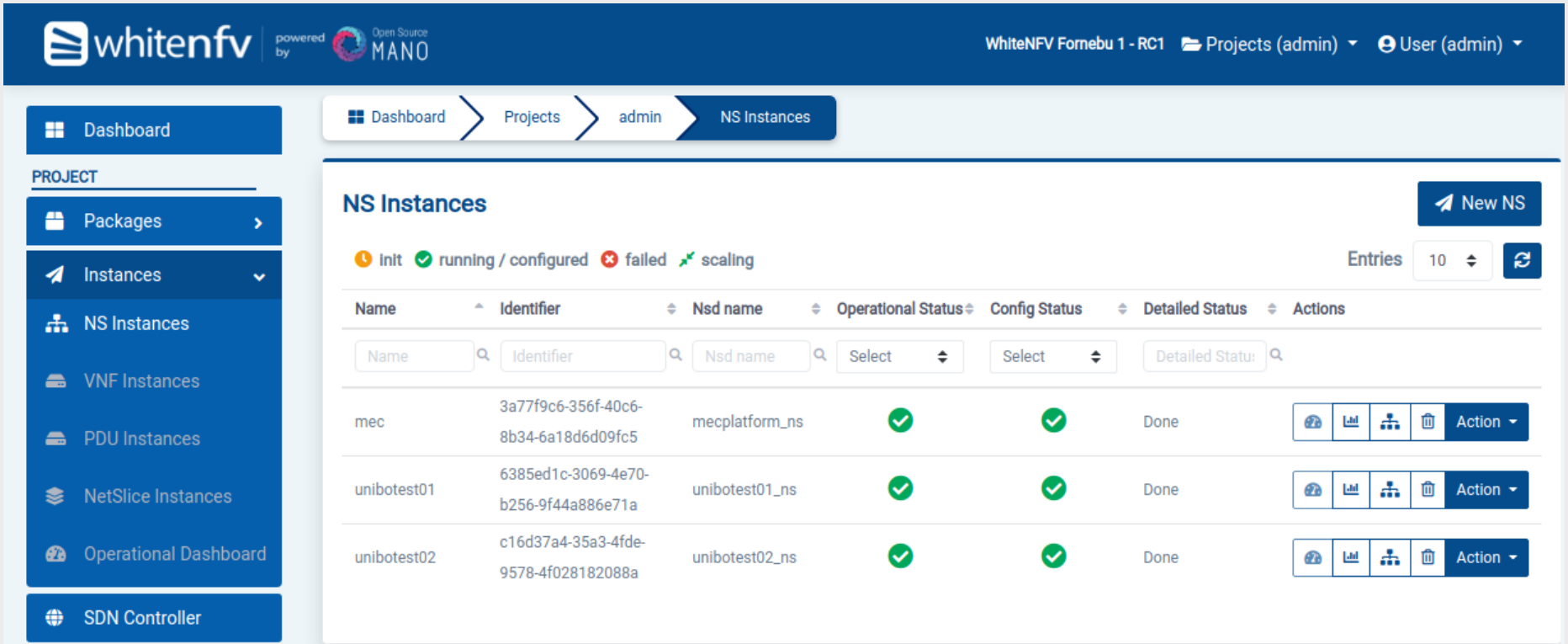
MEC demo

NS creation in WhiteNFV:



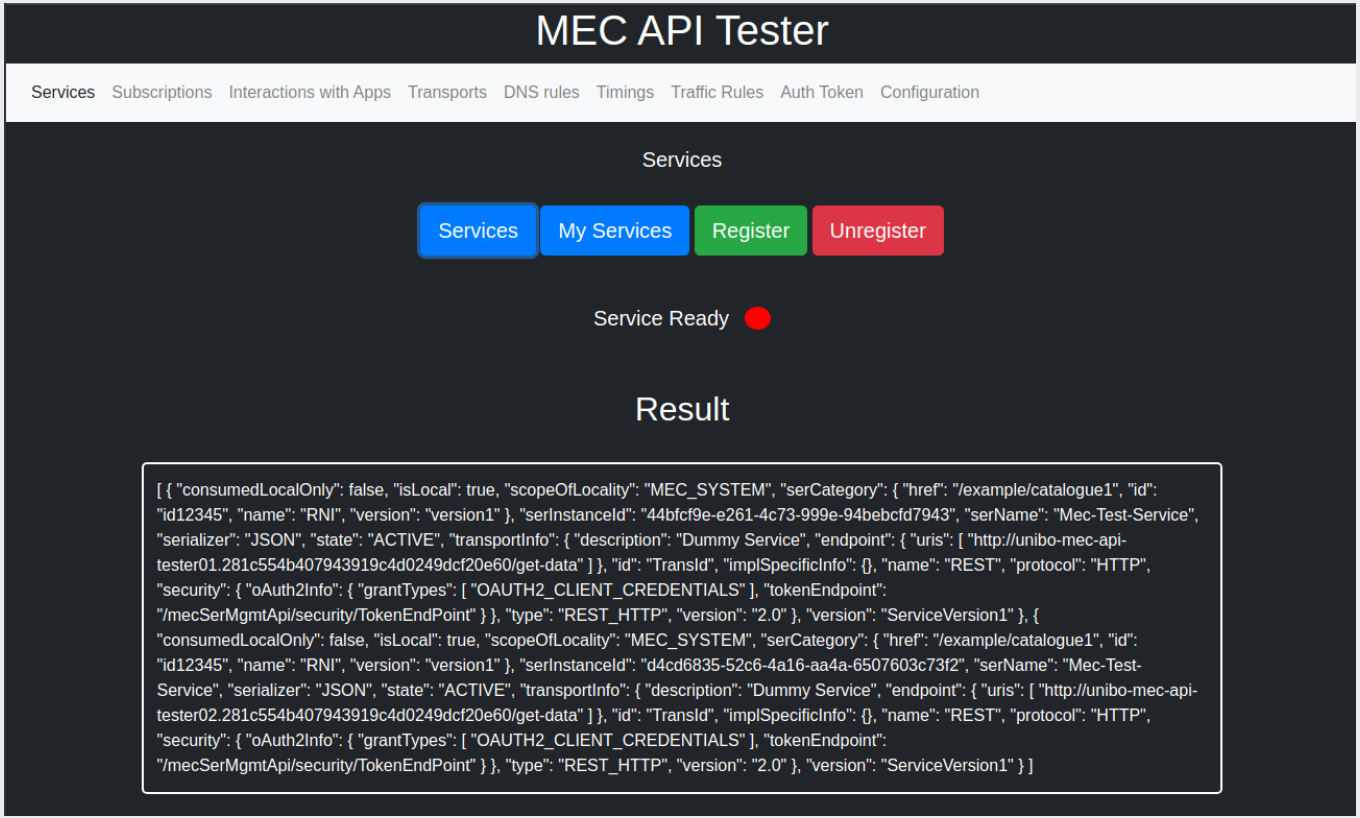
MEC demo

Checking the status in WhiteNFV:



MEC demo

After register the 2 apps:



MEC demo

Details of registered apps:

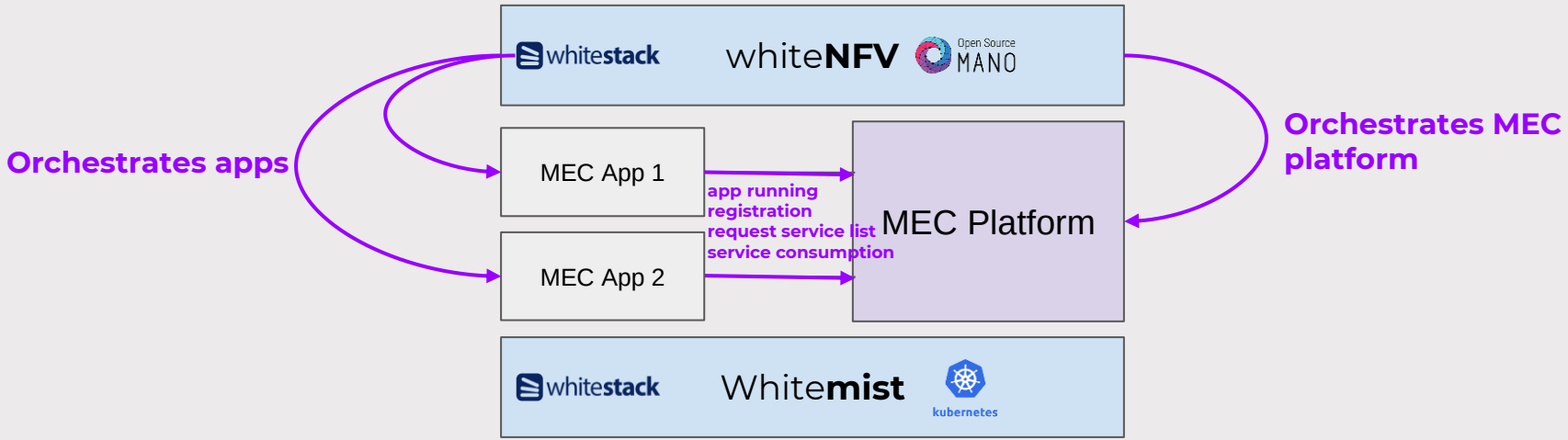

```
{
  "consumedLocalOnly": false,
  "isLocal": true,
  "scopeOfLocality": "MEC_SYSTEM",
  "serCategory": {
    "href": "/example/catalogue1",
    "id": "id12345",
    "name": "RNI",
    "version": "version1"
  },
  "serInstanceId": "44bfcf9e-e261-4c73-999e-94bebcfd7943",
  "serName": "Mec-Test-Service",
  "serializer": "JSON",
  "state": "ACTIVE",
  "transportInfo": {
    "description": "Dummy Service",
    "endpoint": {
      "uris": [
        "http://unibo-mec-api-tester01.281c554b407943919c4d0249dcf20e60/get-data"
      ]
    },
    "id": "TransId",
    "implSpecificInfo": {},
    "name": "REST",
    "protocol": "HTTP",
    "security": {
      "oAuth2Info": {
        "grantTypes": [
          "OAUTH2_CLIENT_CREDENTIALS"
        ],
        "tokenEndpoint": "/mecSerMgmtApi/security/TokenEndPoint"
      }
    },
    "type": "REST_HTTP",
    "version": "2.0"
  },
  "version": "ServiceVersion1"
},
```

```
{
  "consumedLocalOnly": false,
  "isLocal": true,
  "scopeOfLocality": "MEC_SYSTEM",
  "serCategory": {
    "href": "/example/catalogue1",
    "id": "id12345",
    "name": "RNI",
    "version": "version1"
  },
  "serInstanceId": "d4cd6835-52c6-4a16-aa4a-6507603c73f2",
  "serName": "Mec-Test-Service",
  "serializer": "JSON",
  "state": "ACTIVE",
  "transportInfo": {
    "description": "Dummy Service",
    "endpoint": {
      "uris": [
        "http://unibo-mec-api-tester02.281c554b407943919c4d0249dcf20e60/get-data"
      ]
    },
    "id": "TransId",
    "implSpecificInfo": {},
    "name": "REST",
    "protocol": "HTTP",
    "security": {
      "oAuth2Info": {
        "grantTypes": [
          "OAUTH2_CLIENT_CREDENTIALS"
        ],
        "tokenEndpoint": "/mecSerMgmtApi/security/TokenEndPoint"
      }
    },
    "type": "REST_HTTP",
    "version": "2.0"
  },
  "version": "ServiceVersion1"
}
```

MEC demo result

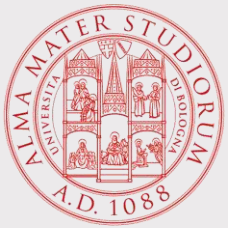
Using OSM (WhiteNFV) we are able to deploy MEC platform and 2 apps on top of kubernetes cluster (Whitemist), then app declares that is running and registers to MEC Platform, finally the apps retrieve the service list from MEC platform.

Every interaction is ETSI standard compliant.



Thanks!


Open Source
MANO


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA


whitestack