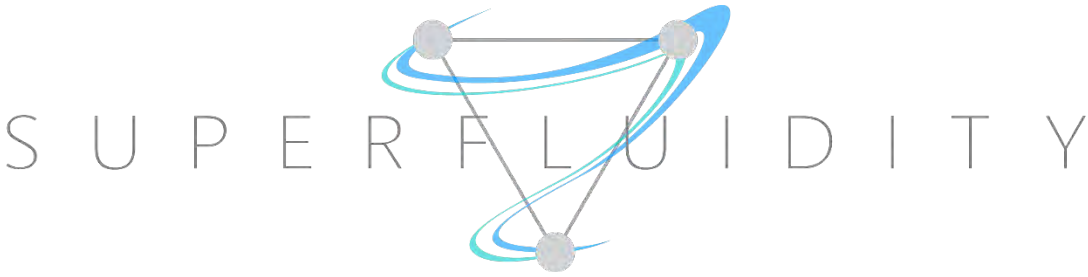


SUPERFLUIDITY

A SUPER-FLUID, CLOUD-NATIVE, CONVERGED EDGE SYSTEM

Research and Innovation Action GA 671566

DELIVERABLE D7.1:

USE CASE SCENARIOS FOR DEPLOYMENT AND VALIDATION

Deliverable Type:	Report
Dissemination Level:	PU
Contractual Date of Delivery to the EU:	30.06.2017
Actual Date of Delivery to the EU:	21.07.2017
Workpackage Contributing to the Deliverable:	WP7

Editor(s):	Philip Eardley (BT)/Francisco Fontes (ALB)
Author(s):	Stefano Salsano (CNIT), Claudio Pisa (CNIT), Lionel Natarianni (NBLF), Bessem Sayadi (NBLF), Erez Biton (ALUIL), Philip Eardley (BT), George Tsolis (CITRIX), Michael McGrath, Vincenzo Riccobene (INTEL), John Thomson (ONAPP), Francisco Fontes (ALB), Carlos Parada (ALB), Rufael Mekuria (USTR), Radu Stoenescu (UPB), Costin Raiciu (UPB), Juan Manuel Sánchez (TELCARIA), Elisa Rojas (TELCARIA), Cyril Soldani (ULG)
Reviewer(s)	John Thomson (ONAPP), Elisa Rojas (TELCARIA)

Abstract: This Deliverable describes the use case scenarios and the performance and functional assessment key indicators that will be used for the final validation and evaluation activities.

Keyword List: use cases, scenes, planning, validation, evaluation, test, metrics

INDEX

List of Figures

List of Tables

List of Abbreviations

5G-PPP	5G Infrastructure Public Private Partnership – https://5g-ppp.eu/
ADC	Application Delivery Controller
BBU	Base-band Unit
CDN	Content Distribution Network
CPX	Citrix NetScaler (Container version)
CRAN	Cloud Radio Access Network
(D)DoS	(Distributed) Denial of Service
EMS	Element Management System
eNB	E-UTRAN / Evolved Node B – also known as eNodeB
EPC	Evolved Packet Core
EU_x	(Denotes different) End-User(s)
FQDN	Fully Qualified Domain Name
GTP	GPRS Tunnelling Protocol
ISDN	Integrated Services Digital Network
KPI	Key Performance Indicator
LBaaS	Load-Balancer as a Service
LCM	Life-cycle Management
LTE	Long-term Evolution (5G standard)
MANO	Management and Orchestration
MAS	Management and Analytics System
ME	Mobile Edge
MEC	Multi-access Edge Computing
MEO	Multi-Edge Orchestrator
NFV	Network Functions Virtualisation
NFVI	Network Functions Virtualisation Infrastructure
NU_x	(Denotes different) Neighbour User(s)
ONF	Open Networking Foundation - https://www.opennetworking.org/
ONOS	Open Network Operating System - http://onosproject.org/

OpEx	Operational Expenditure
OPP	Open Packet Processor
OTT	Over-The-Top
OvS	Open vSwitch
P-GW	Packet Gateway
PoP	Point of Presence
QoE	Quality of Experience
QoS	Quality of Service
RDCL	RFB Description and Composition Language
RDCL 3D	RFB Description and Composition Languages Design, Deploy and Direct
REE	RFB Execution Engine
RFB	Reusable Function Block
RNIS	Reseau Numerique a Integration de Services, (European name for ISDN)
RRH	Remote Radio Head
SDN	Software-Defined Networking
S/Gi LAN	(S – refers to 4G) Gateway-Internet LAN; 3GPP reference point between P-GW and Packet Data Network
SGW	Serving Gateway (3GPP LTE)
SLA	Service Level Agreement
SLO	Service Level Objectives
TALE	Throughput, Anomaly, Latency and Entropy
TOF	Traffic Offloading Function
UE	User Equipment
URL	Uniform Resource Locator
vEPC	virtual(ised) Evolved Packet Core
VIM	Virtual(ised) Infrastructure Manager
VM	Virtual Machine
VNF	Virtual Network Function
VPX	Citrix NetScaler (Virtualised form)
xFSM	eXtended Finite State Machine

1 Introduction

The efforts of WP7 are focused on building a set of demonstrators that can showcase the various technologies that have been worked on, in the scope of Superfluidity, in a consistent and coherent manner. The aim throughout the Work Package is to consolidate the research outputs into an integrated system that fulfils the goals of the project as well as providing feedback and evaluation information back to the other technical Work Packages (WP3-6). The majority of work represented in this Deliverable is based on the work carried out in Task 7.1. In this Deliverable the aim is to design and describe the test scenarios and the associated validation scenarios.

1.1 Deliverable description

The Superfluidity project will establish an integrated demonstrator, in order to validate and assess the project's developments. This is intended to integrate, showcase and evaluate the project results, with a special focus on the Reusable Function Blocks (RFB), an important, architectural concept developed within the project. For that purpose, use cases are defined, each focusing on specific functional areas, covering almost all of the project technical areas.

The use cases were based on particular technology areas that partners wanted to collaborate on, based on initial rounds of discussion and were then combined into a meaningful storyline. By adapting the individual use cases to fit in a storyline, emphasis was made on the re-usability of the outputs of the stages and thus ensuring that common languages, APIs and models were used for the integration. By deciding to use a consistent integration approach also meant that it was important for the partners to consider the implications of certain choices and ultimately ensured that there would be a sequential storyline that didn't skip any steps.

To ensure that all the scenes and the associated use cases perform well together an integrated demonstrator was targeted with the use cases that had been identified being executed in a logical sequence. The storyline captured represents the expected lifecycle of a 5G operator, on its infrastructure, network and service platforms.

The objectives of the demonstrator are:

- To ensure that the use cases are well designed and linked to the overall objectives
- To help guide the development and integration
- To plan for the evaluation and validation of Superfluidity technologies
- To help on demo improvement, providing feedback and promoting the identification of additional aspects that are worth being added.

1.2 Earlier Deliverables Inputs

This report builds on previous project results, in particular:

- The use cases identified in D2.1
- The requirements and KPIs identified in D2.1 and the related, higher-level 5G-PPP targets
- The architecture of Superfluidity as described in D3.1
- The four independent demos executed after the 1st year of project at the 1st review, which were not integrated (briefly summarised in "Annex A: Initial demos", for the benefit of readers from the general public that may find the demos of interest, as they have not yet been described in any Deliverable)

Deliverable D2.1 described a wide range of use cases that were expected to be relevant to Superfluidity, as well as the associated technical and business requirements. Those use cases fell in the overlapping areas, as listed:

- The wireless access part of the network
- Edge computing
- Video delivery
- Services delivered from the edge
- Monitoring
- Security

We want our integrated demonstration to collate aspects of all these areas, composing a broader spectrum than any of the individual D2.1 use cases alone. The primary purpose is to bring together most of our technology developments, in a series of simple, inter-connected, sequential use case scenarios (scenes).

D2.1 also provided an initial identification of relevant metrics and KPIs, which the demonstration will mostly integrate. Generically, they are related to:

- Analytics – the ability to collect metrics (monitoring) and analyse them
- Orchestration – take decisions on service deployment or migration
- Scalability – the ability to scale up /out
- Service agility – the time to develop, deploy or tailor a service
- Security – the ability to detect and isolate anomalous events

1.3 5G and 5G-PPP Indicators

The 5G-PPP community, as the main communication channel for the broader 5G community, proposes a list of requirements for the 5G networks that are to be achieved at an operational level. The Superfluidity project is convinced that, directly and indirectly, it could contribute towards achieving the following key performance enhancements:


- 1000 times higher mobile data volume per geographical area
- 10 to 100 times more connected devices
- 10 to 100 times higher typical user data rate
- 10 times lower energy consumption
- End-to-End latency of < 1ms
- Ubiquitous 5G access including in low-density areas

In addition, other specific 5G goals for Europe include:

- Retaining at least 35% of the global market share in Europe for equipment
- European industry driving the development of 5G standards and having at least 20% of the 5G Standards Essential Patents
- Supporting lower cost access to a wider spectrum of applications and services
- Reduction of the network management OpEx by at least 20% compared to today
- New lightweight but robust security and authentication metrics

The defined use cases aim at providing the right context to demonstrate Superfluidity's contributions towards the enhancement of most of the previously listed, performance indicators.

1.4 Project Technologies

The Superfluidity project tackles a broad spectrum of innovative technologies under the structure of different Work Packages. This work has been documented in several Deliverables already released and outlined in Section 1.2. Figure 1 depicts the mapping between these technologies and the respective Work Packages. In order to showcase as many of the project's achievements as possible, most of these technologies are included in the integrated demonstrator.

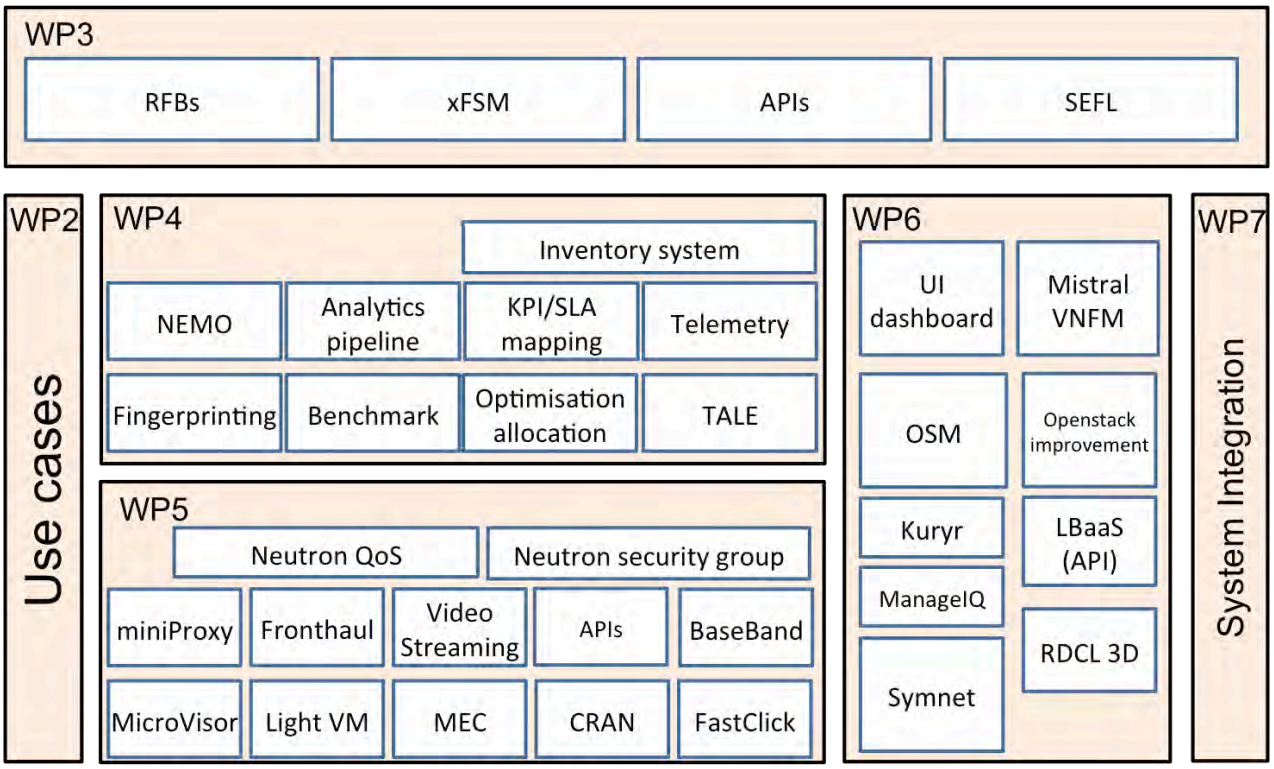


Figure 1: Graphical summary of Superfluidity's technologies

The integrated demonstration comprises the execution, in sequence, of a list of scenes (use cases). From an end user perspective, the story is built up around their experience of watching a video, streamed over the network. Video today is the dominant Internet application and the proportion of resources it needs is expected to increase over the next years. Cisco has reported that it will account for more than 80% of the Internet traffic by 2020 [5]. Furthermore, this technology can be incorporated into a wider range of novel user-centric applications, for example, augmented reality, or multi-user gaming, where a headset displays live video from other participants.

1.5 Use cases execution

The use cases selected to run in the integrated demonstrator are organised as a sequence of scenes. Each scene progresses from one to the next, triggered by some change in context (for example: more users arriving or a DDoS attack or some noisy neighbour effect on the platform) which, technology-wise, is noticed by a change (impact) in the monitored KPIs or by event detection. The use case corresponding to the scene thus demonstrates how the integrated system reacts in order to recover and return it back to the desired operating state.

The scenes run sequentially, sharing a common and evolving infrastructure. This way, the final conditions left by one scene will serve as initial conditions for the following scene. This strategy will

contribute to the achievement of a large integrated system, showcasing a vast amount of the Superfluidity technology all inter-operating, as if it was a real environment.

1.6 Document Organisation

Following this introductory section, Section 1, where we have referred to previous Deliverables and the project integrated demonstrator to be built, Section 2 summarises the scenes that comprise the integrated demonstrator. Section 3 goes through each scene sequentially and describes in depth the technical details of each, presenting the technology, challenges and evaluation plans. Section 4 summarises the preceding section, providing a simplified, global view of used Superfluidity technologies, challenges and evaluation plans for all scenes. Finally, Section 5 lists the conclusions that were made on the basis of the work presented in this Deliverable. Annex A is devoted to describe the four, standalone Demos that were showcased to the reviewers at the end of the first project year in September 2016 and also presented at the 5G Summit in Rome in November 2016.

2 Integrated demonstrator scenes

The following table summarises the sequence of the identified scenes (representing the use cases). The initial scene, Scene 0 is not a regular scene, but is included here due to its relevance in the overall demonstrator. It is in this first step that the infrastructure and the management/orchestration/design tools are deployed. A significant volume of complex work is required for this initial preparation, in order to identify, install and put in operation a large number of complex components that inter-operate, providing support to the integrated execution of Superfluidity components. Some of the tools and software (e.g. OpenStack, Kubernetes, Grafana, InfluxDB) are not specific results of the project but their configuration and modification have been important to realise the demonstrator.

Additionally, an end-user perspective is provided, to better understand the scene objective and the role of the end-users and other primary users in the scene execution. Other key roles and perspectives are highlighted but it is not the focus for this particular discussion and will be detailed in more depth in D7.2.

Table 1 – Summary of Scenes

Scene	Summary	End-User perspective/experience	Technical perspective
0	Infrastructure setup	No end-user involvement. Infrastructure provider is involved at this stage.	Establish the infrastructure (hardware & NFVI)
1.a	Superfluidity system design	No end-user involvement. The system designer, telecommunications operator and the network administrators would be involved at this stage.	Edge and cloud, network and service platforms design and deployment artefacts generation.
1.b	Initial components deployment (CRAN and MEC)	No end-user involvement. The system administrator and telecommunications operator would monitor and operate at this point.	Network and service platforms components deployment at edge and central clouds. CRAN and MEC are defined and deployed programmatically.
2.a	Workloads offline	No end-user involvement. To	Characterisation of

	characterisation	understand the likely impact of a workload on the deployed infrastructure it is important to model and classify the workloads. This is done by the tools and modelling experts.	workloads on the deployment environment, for scaling mechanisms support.
2.b	Workload (video streaming) / service deployment	No end-user involvement. The service provider will set up the services that will be exposed to the end-users.	Initial deployment of the workload based on a combination of model profiles and the service providers' inputs.
3.a	Central cloud services automatic scaling	An end-user streams a video from the network. The demand at the core grows or there are noisy-neighbours triggering scaling.	Other users are simulated using a simulated load profile and this load automatically triggers the server scaling based on predefined actions.
3.b	Services relocation to edge cloud	An end-user, now connected to the CRAN, also streams a video from the network.	An operator policy triggers a service component to be instantiated at the attachment edge and the video content is then streamed via the MEC.
3.c	Container based services deployment at the edge cloud	Due to the end-user reaching a certain level of content watched, a user-specific advertisement is displayed.	Using a service at the edge that combines video stream content and user-specific content into an advert stream.
3.d	Unikernel based services deployment at the edge cloud	The end-user continues to stream the video, meanwhile there is a DDoS attack at the edge cloud.	A DDoS attack is detected through the OPP running in the MEC and triggers a set of SDN rule modifications combined with xFSM.

3.e	Services' optimisation at the edge cloud	The end-user changes to utilise an encrypted variant of the video stream.	The ADC is realised through a deployment of Citrix NetScaler at the Edge cloud.
4.a	Alternative load balancing function at the central cloud	The end-user continues to stream the video. The network operator though decides to replace the open LBaaS with a commercial variant that promises better performance.	The VNF of the LBaaS is replaced with minimal service disruption to the video streaming service.
4.b	Alternative edge offloading function	The end-user continues to watch the video stream. The operator decides to switch the traffic off-loader to use fast-click for performance reasons.	The TOF used in 3.b is replaced with a Fast-Click based alternative as a plug-in replacement.
4.c	Advanced network control and management	Currently it is not possible to keep the end-user service running through this disruptive change.	Once modified the system operator can modify the SDN rules programmatically and change the end-user experience based on configurations and desired SLAs.

Figure 2 depicted below shows how each scene builds and follows on from previous ones.

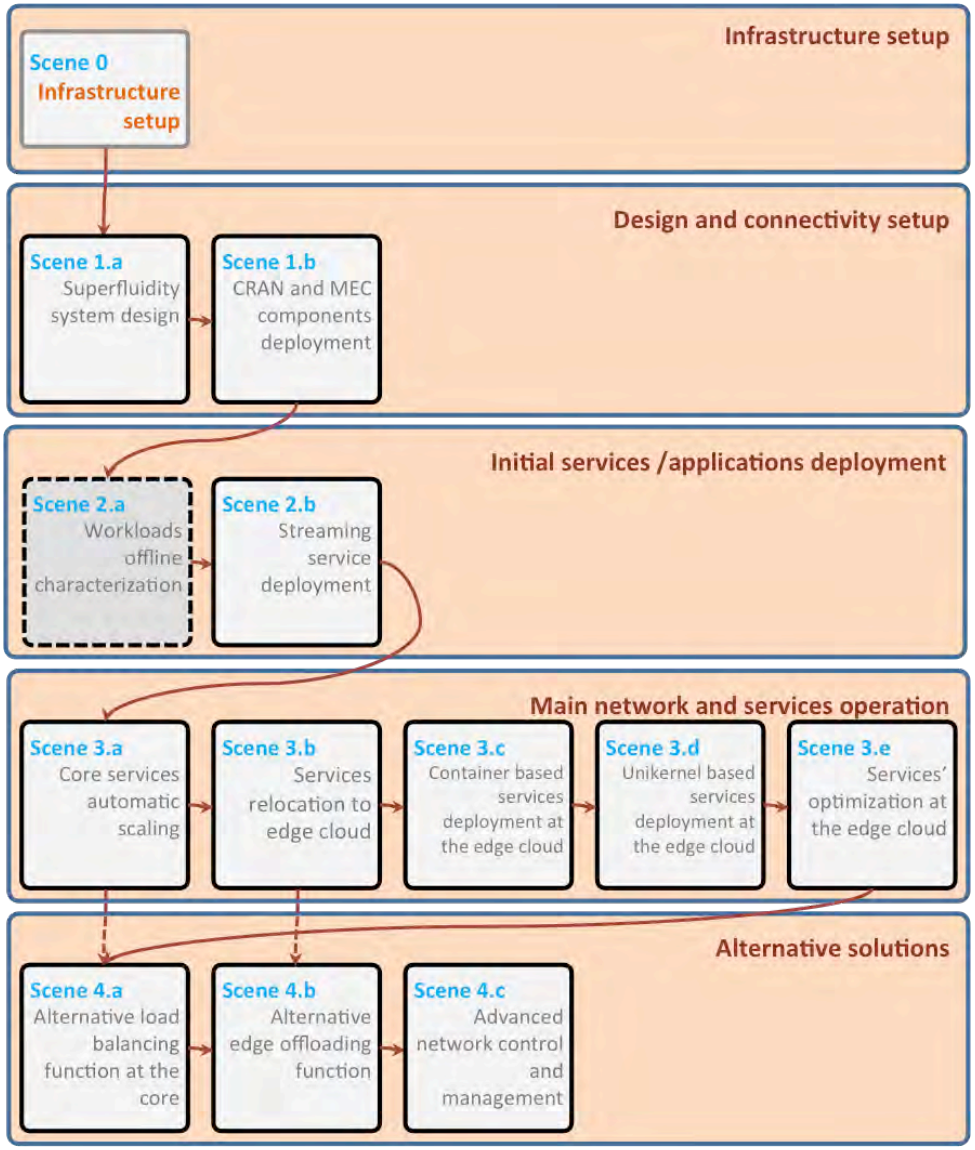


Figure 2: Relationship between Scenes

3 Use Cases Detailed Technical Description

This section describes the use cases from a technical perspective, detailing the Superfluidity technologies that are addressed, as well as the actions that need to be taken by the operator. The demo comprises a sequence of scenes, which constitute significant steps on the demo completion.

Note that the Superfluidity advancements are not only devoted to improving the customer experience, but also to increasing the operators' agility and efficiency, which benefit customers by lowering costs.

3.1 Scene 0: Infrastructure setup

3.1.1 Scene description

This is a special scene, where all the actions required to set up the distributed cloud infrastructure, as well as the deployment of all the static components, namely regarding management and orchestration functions (ManageIQ), as well as service design (RDCL 3D), are executed. RDCL 3D is a new Superfluidity developed technology that helps create and orchestrate RFB components that is described in a little more detail in Section 3.2.3. ManageIQ is a pre-existing software stack that has been modified by Superfluidity to help perform the management operations across multiple cloud sites as needed for the integrated demonstrator.

The project will use multiple virtualisation technologies, namely virtual machines (VMs) and different flavours of containers, in the realisation of the Superfluidity components that comprise the use cases. Thus, the infrastructure to be deployed must provide efficient, integrated support for the deployment, execution and operation/management of the underlying heterogeneous environment that will ultimately comprise the Superfluidity integrated system.

It is foreseen that to fully support the integrated demonstrated the following (not exhaustive) set of tools will be leveraged and modified for usage in Superfluidity:

- Openstack
- Kubernetes, Openshift
- Monitoring / Telemetry (SNAP)
- ManageIQ (with project developments)
- RDCL 3D (project result)

These components will span across the two defined project clouds:

1. Central cloud, where networking and services' core components are deployed.
2. Edge cloud, where access and edge networking and services distributed components is deployed.

All the details about the technology, such as versions or main configurations, is provided in Deliverable I7.2 (“System Integration Plan”) and updated later in D7.2 (“System integration report”).

3.1.2 Technical trigger to start this scene

This is the initial scene. The scene is built on top of the physical infrastructure made available for the project at both edge and central clouds.

3.1.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

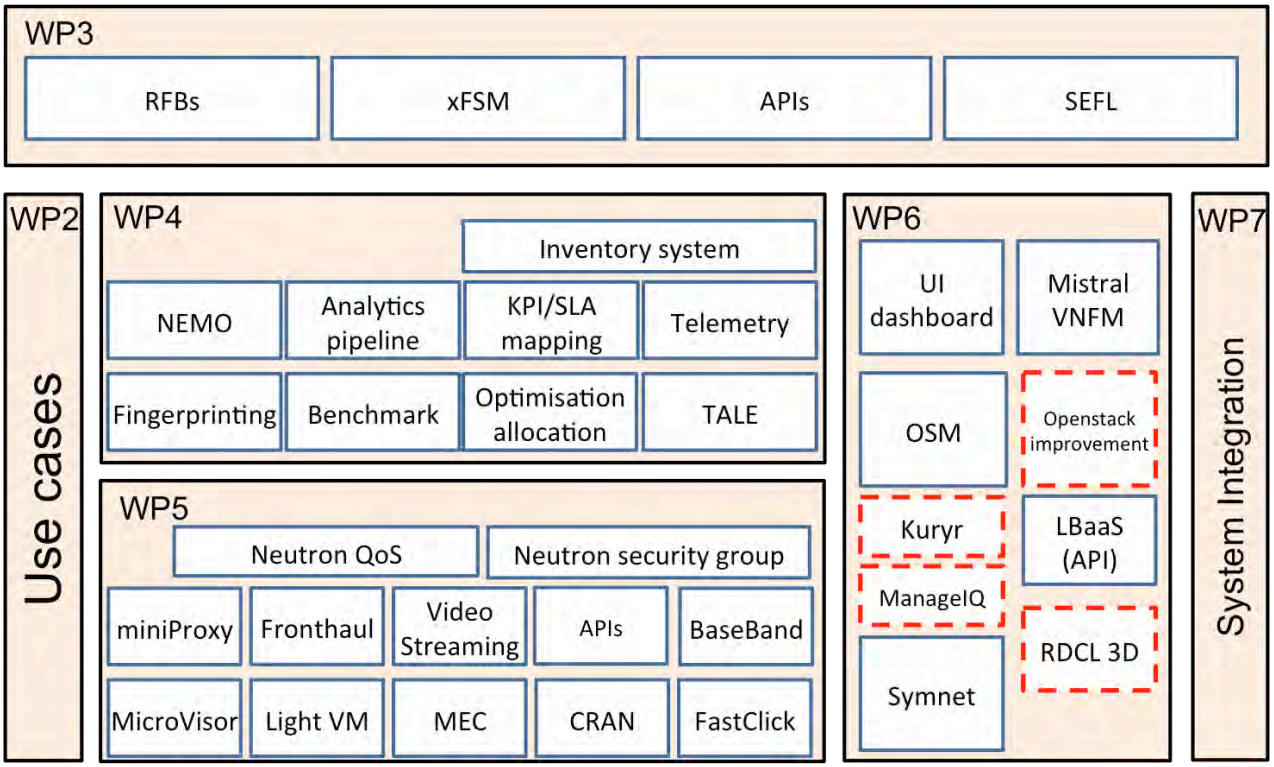


Figure 3: Graphical summary of Scene 0 technologies

3.1.4 Implementation challenges for the demo

The main challenges in this scene setup are:

- Identification of the requirements, set by the other scenes
- Identification of required infrastructure components, answering identified requirements
- Setup of the required components, with the existing dependencies
- Edge and central clouds interconnection

At the end of this scene, the Superfluidity integrated system can be designed (next scene) with the help of the RDCL 3D tool and the deployment of components can start.

3.1.5 Evaluation plan and validation metrics

No evaluation and validation is foreseen for this scene. The evaluation aspects will be covered through the evaluation of the other scenes.

3.1.6 Relevant 5G-PPP metrics

The infrastructure deployed in this scene shall indirectly contribute to the achievement of the enhancements targeted by 5G-PPP, through simplicity and efficiency, by allowing the other components to contribute directly to them.

3.2 Scene 1.a: Superfluidity system design

3.2.1 Scene description

This scene is responsible for designing the basic services to be deployed over the physical infrastructure, which in turn will be used to offer services and execute workloads. In particular, this scene will design the logical components of an Operator System: the CRAN infrastructure, the MEC platform, the EPC. A graphical design tool, RDCL 3D (*RFB Description and Composition Languages Design, Deploy and Direct*) is used to edit the descriptor files that represent the Operator System.

3.2.2 Technical trigger to start this scene

This scene is started when the operator wants to deploy a 5G network over its basic infrastructure, which has been deployed in scene 0.

3.2.3 Technical description and involvement of Superfluidity technology

RDCL 3D is a web application that supports the editing and deploying of NFV based services on top of existing orchestrators. It provides a web GUI that can be used to visually represent and design descriptor files for NFV services and components (e.g. VNFs). RDCL 3D is able to interact with the concrete descriptor syntax used by existing MANO orchestrators, as well as to represent services and components at a higher level of abstraction, following the RFB based architecture proposed by Superfluidity.

The design of the 5G network primarily involves modelling and decomposing the infrastructure into RFBs and then composing them with the RDCL 3D tool. The RDCL 3D tool logically interacts with the Manage IQ orchestrator, designing the network topology and configuration that will be implemented using Kuryr. Through RDCL 3D and the connected pieces it is possible to configure and deploy the BaseBand, Fronthaul, CRAN and MEC technological components.

We have not explicitly considered it in our demo, but the approach can be naturally extended to support different 5G networks (or “network slices”) over the same basic infrastructure and to

support the reconfiguration of an existing 5G network (e.g. for scaling up / down the capacity to react to end-user demands).

The following Figure refers to the map of technologies used in this scene.

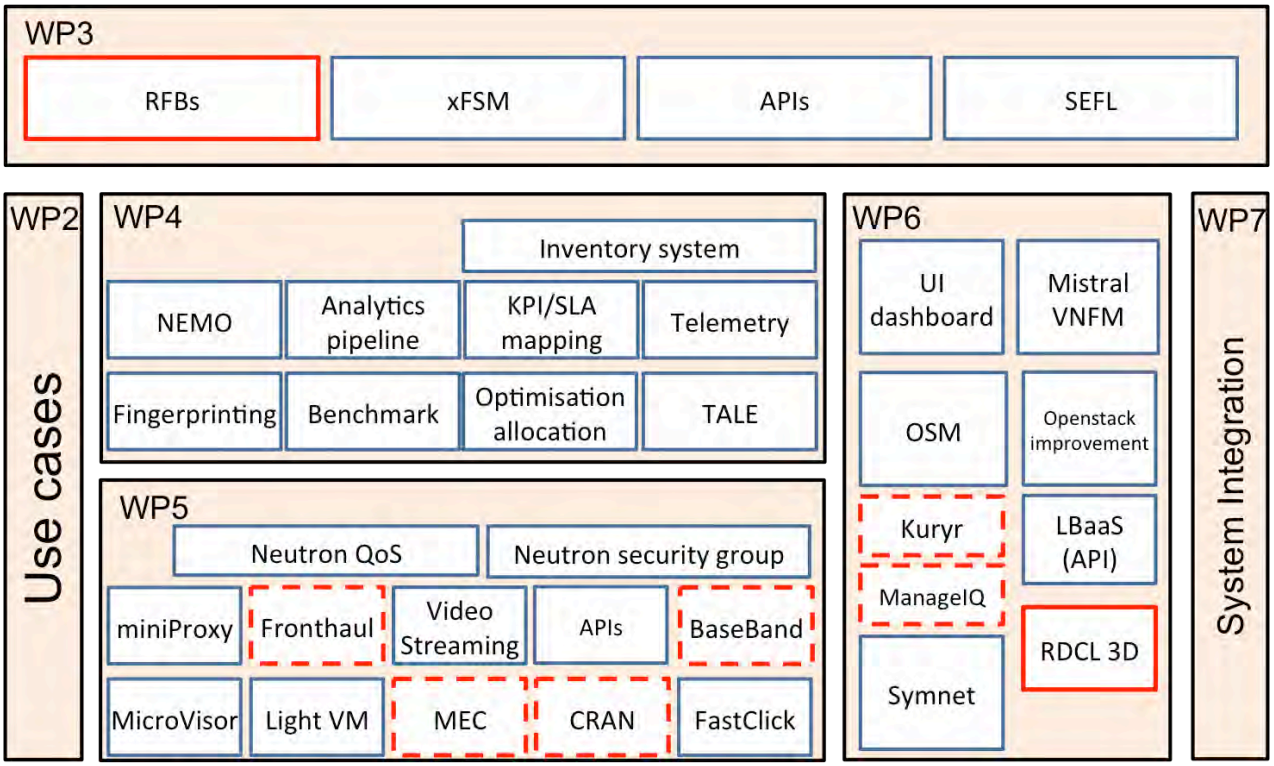


Figure 4: Graphical summary of Scene 1.a technologies

3.2.4 Implementation challenges for the demo

Existing models for the representation of NFV services do no provide the required expressiveness to represent the design and deployment of the 5G network that we are considering. Likewise, existing orchestrators are mostly focused on "higher level" cloud based services and cannot directly support our scenario. For these reasons, we have faced the following challenges:

- Enhance the existing models and/or consider new modelling languages
- Develop a new tool with its GUI (the RDCL 3D)
- Decouple the design phase from the orchestration phase and provide the required conversions
- Interact with a "meta orchestrator" like Manage IQ, which provides the required flexibility to control different orchestrator types and orchestrator instances.

For further information about the design choices underlying the modelling and on the RFB architectural concept the reader is invited to read D3.1, the Superfluidity Architecture.

3.2.5 Evaluation plan and validation metrics

This scene is concerned with the design phase performed by the operator. Therefore, the validation should be in terms of: expressiveness of the models and of the RDCL 3D tool, simplicity of usage, simplification of the design and deployment procedures, reduction of the time needed to design the 5G networks with respect to previous approaches. These metrics are not quantitative; therefore we will perform a qualitative self-assessment. Thanks to the improvements to the design phase discussed in the previous section, the solutions demonstrated in this scene provide a contribution towards the achievement of the following relevant 5G-PPP metrics:

- Reduction of the network management OpEx by at least 20% compared to today
- Supporting lower cost access to a wider spectrum of applications and services

3.3 Scene 1.b: CRAN+EPC and MEC components deployment

3.3.1 Scene description

This scene is responsible for instantiating the network components used in the following scenes. In particular, this scene must deploy, at the edge, the virtualised components of a mobile network, the CRAN, and some edge-level MEC elements, such as the TOF (*Traffic Offloading Function*). At the central cloud, this scene must deploy the core components of the mobile network, the EPC (*Evolved Packet Core*), as well as the Management and Orchestration (MANO) components of the MEC solution.

3.3.2 Technical trigger to start this scene

The operator uses the RDCL 3D tool to build a graph representation of the RFBs composition. From this graph, the RDCL 3D tool produces an artefact file, the RFBs.yml file, used for the deployment of the RFB composition.

In this scene, the RFB controller, from its REST API, loads the RFBs.yml and then initiates the complete deployment (resources + RFBs).

The process is agnostic about the underlying infrastructure and resources. The RFB controller receives a set of RFBs to be created and deployed. Resources and RFBs are decoupled using nodes as logical representation of resources and deployment requirements. The mapper registers available resources in a dedicated database containing all the available VMs and hardware for deployment.

After the full deployment of RFBs on nodes, the CRAN, EPC and MEC are deployed on the Edge and the Central cloud; the LTE network is operational and services, in the form of ME Apps may be deployed. At this step, the CRAN network can accept User Equipment (UE) connections.

3.3.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

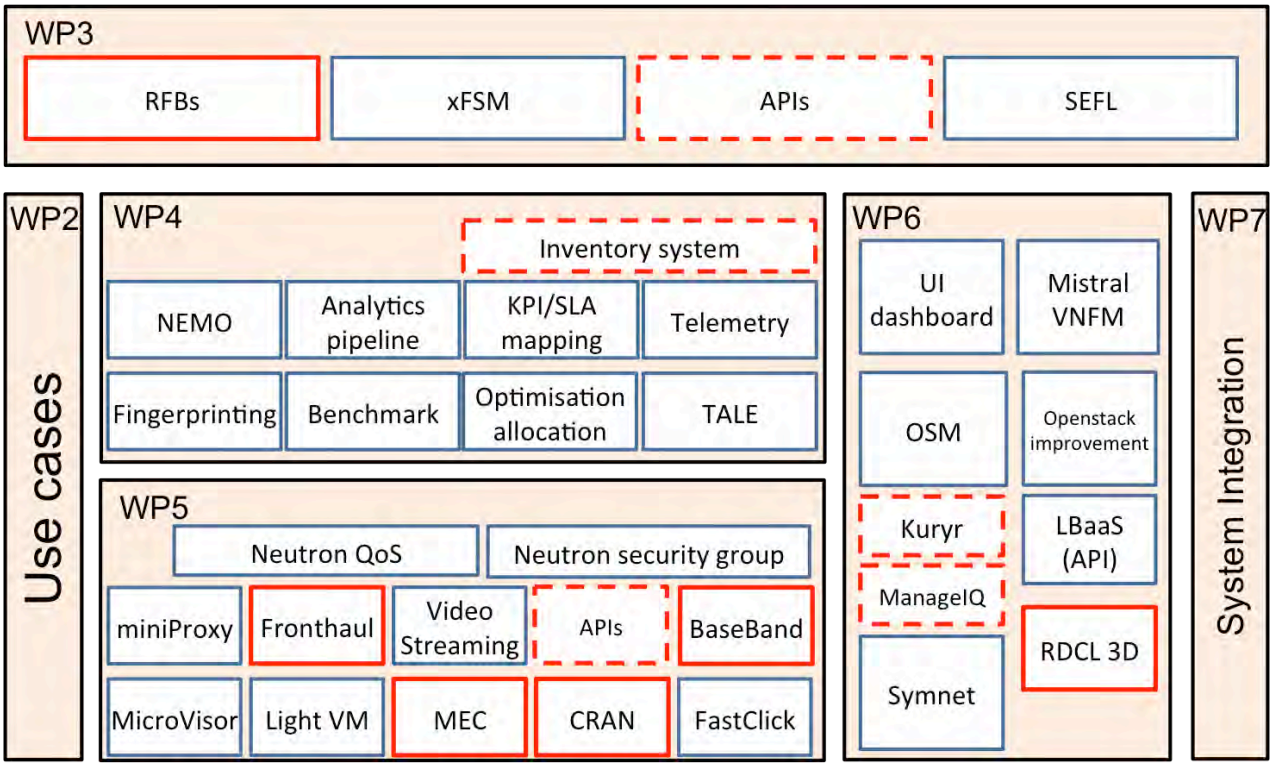


Figure 5: Graphical summary of Scene 1.b technologies

All deployments are performed by using the *RFB Execution Engine* (REE), which contains an internal high-level orchestrator used for resource and RFB deployments. The role of the REE is to decouple resource deployments from service deployments and implement the RFB graph model execution.

Function composition needs to be deployed on nodes. The objective of the first step is to build nodes from available resources, onto which RFBs are deployed in the second step. To address the resource element, the REE provides drivers on its southbound to control and deploy resource items (e.g. OpenStack Heat). The available resources items are stored in the resource repository, which is also managed by the REE orchestrator.

The deployment of the whole platform is performed in two main steps. In each step, the RFB controller performs a step to read the RFB artefact file. See the Figure below for details.

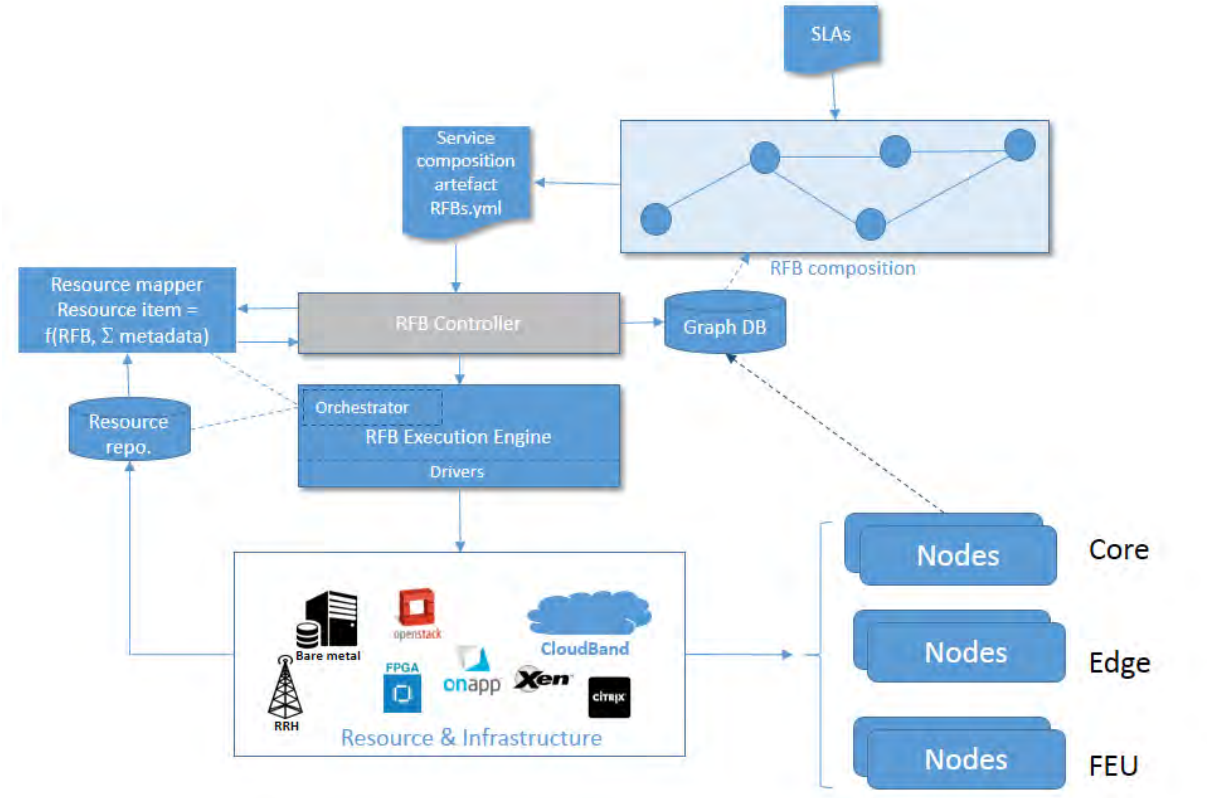


Figure 6: Resources allocation for RFB

a) On the first step (see Figure 6), the RFB controller receives a set of RFBs to be created. Related resources are provided by the resource mapper (VM and bare metal servers).
   - The RFB controller is invoked with RFBs.yml, which contains the full composition described as an RFB service graph.
   - The REE, from its orchestrator, initialises related resource deployment by interpreting each RFB and translating each of them into a resource element (e.g. VM, hardware server) from the resource mapper.
   - Related resource elements are initialised; from this point, each resource element is represented as node and stored in the graph DB.

b) For the second step (see Figure 7), the CRAN+EPC and MEC RFBs are deployed: RRH, BBU, EPC, SDN fronthaul controller and its northbound application to set up the link between the RRH and the BBU, MEC ToF and MEC multi-edge orchestrator (MEO).

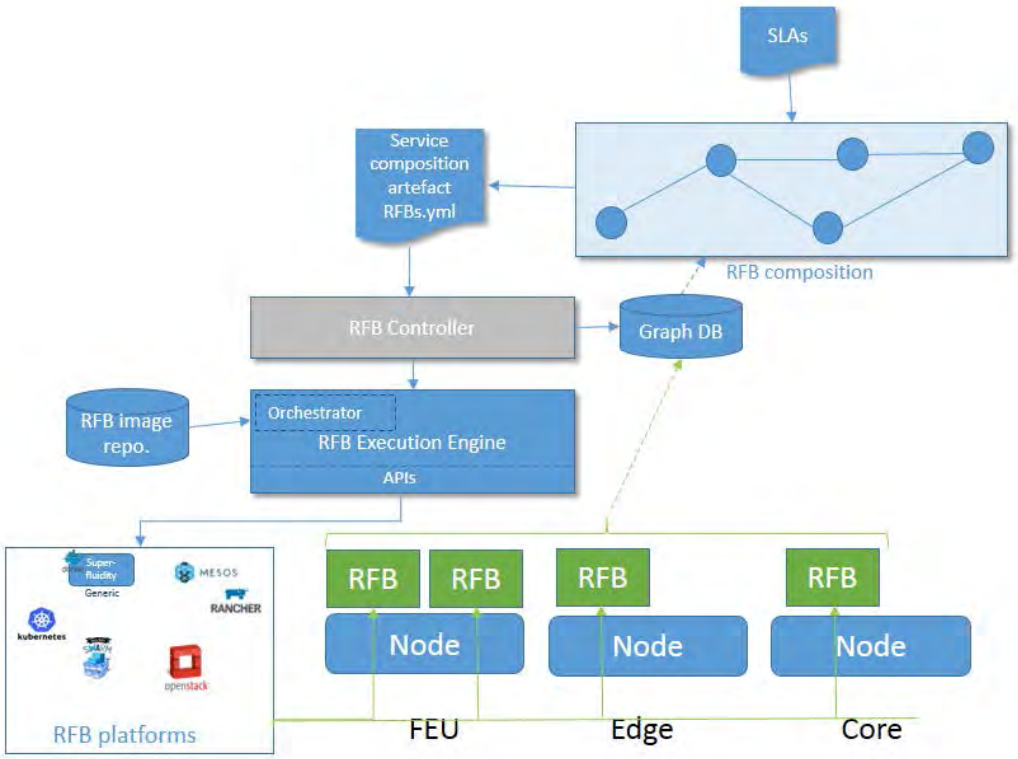


Figure 7: RFBs deployment

- The second step starts from deployed resources and nodes; the REE reparses the RFBs.yml and each RFB and its related metadata are extracted and interpreted
- According to the requirements and affinities (describes in RFB metadata), the REE orchestrator deploys RFBs on corresponding nodes using the RFB image repository

3.3.4 Implementation challenges for the demo

a) Provide a composition model for CRAN deployment as RFB

A service composition format has been defined according to service requirements and customer SLAs for the CRAN and fronthaul deployment.

We provide a model for service composition as an artefact file (RFBs.yml) that contains the RFB chaining graph and related metadata used for the deployment.

The composition structure built for the demo follows SLA original requirements. We enrich the artefact file with refinements to describe specific capabilities (redundancy, compute & network capacity, etc.) as a metadata sub-section for each RFB.

b) Decoupling RFB from nodes for dynamic deployments

Successive service compositions, deployments and 'tear-downs' can be performed using the same resource deployment system. It means that resources (nodes) do not have to be redeployed

between successive RFB deployments. The same RFBs.yml artefact file is used for the two steps (resource and service deployment). It is also use to remove RFBs and nodes.

c) Provide a high level orchestrator, enabling multi-level orchestration for CRAN

All deployments are performed by the RFB Execution Engine (REE), which contain an internal high-level orchestrator used for resource and service orchestration.

The role of the REE is to decouple resource from service deployments and orchestration, and to implement the RFB graph model execution.

A light and generic RFB platform is used for service deployment. The REE allows orchestration from state of the art RFB platforms for orchestration such as Google Kubernetes, Docker Swarm and Apache Mesos.

All deployed RFBs are registered and managed as a graph, using a graph database and related query language (e.g. Neo4j, Cypher).

d) Guarantee MEC ToF component does not interfere with backhaul traffic

MEC ToF will be in the path between CRAN and EPC (S1-U interface), with traffic flowing through it. It will be deployed as an RFB, as a VM. Due to the S1-U operation, MEC ToF activation and operation shall not interfere with exchanged control packets. The right traffic filters must be setup from the beginning. These will be updated as Mobile Edge (ME) Applications are instantiated.

3.3.5 Evaluation plan and validation metrics

The evaluation plan will focus mainly on quantifying the improvement in the **instantiation time** for the RFBs and nodes.

We also intend to evaluate the **orchestrator efficiency** for resource-RFB optimisation, **cost** and **performance optimisation** (locating RFBs in the specific geographic area).

Some other fronthaul and SDN capabilities (**latency**, **throughput**) measurements will also be performed.

3.3.6 Relevant 5G-PPP metrics

Metrics will be measured in milliseconds for instantiation time and other CPU usage, memory allocation and network performance and load for different deployment patterns.

3.4 Scene 2.a: Workload offline characterisation

3.4.1 Scene description

This scene describes the pre on-boarding characterisation of workloads before deployment onto the Superfluidity system using automated service characterisation based on the application of telemetry and analytics. Workload characterisation is used to optimise service performance and to improve predictability of service behaviours under operational conditions. Secondly, characterisation is used to support the identification of metrics, which can be used for implementing a scaling strategy for the service. If the value(s) of a metric or combination of metrics identified by the characterisation are approaching pre-defined thresholds then scaling actions can be triggered in order to maintain SLA compliance. Finally, the scene focuses on mapping of platform metrics to service level key performance indicators. By selecting only the metrics of relevance for a given KPI from the hundreds of available metrics this enables telemetry collection reduction.

3.4.2 Technical trigger to start this scene

When on-boarding a workload onto the Superfluidity system, characterisation is carried out as a pre-deployment step to develop a clear understanding of the workload's behaviour under the operational conditions. The characterisation process supports the identification of the most influential metrics, which must be monitored and collected during a deployment. Secondly, the process supports the development of models/heuristics, which are used by the Superfluidity service orchestration layer to manage the performance of the workload. Specifically the Orchestrator must maintain the SLA defined for the workload for the demonstration. The SLA contains the specific KPIs that must be maintained by the workload in order to ensure the required end user QoE. Telemetry also provides on-going monitoring of a service during operational deployment. For the purposes of the Superfluidity demonstration, the telemetry will be used to monitor the performance of the workload under different scenarios (e.g. deployment with noisy neighbours). Telemetry will also enable the monitoring of the Superfluidity system elements, such as fronthaul and SDN capabilities as outlined in Scene 1, which require the measurement of throughput and latency for example.

The rest of the process is described in the following sub-sections.

3.4.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

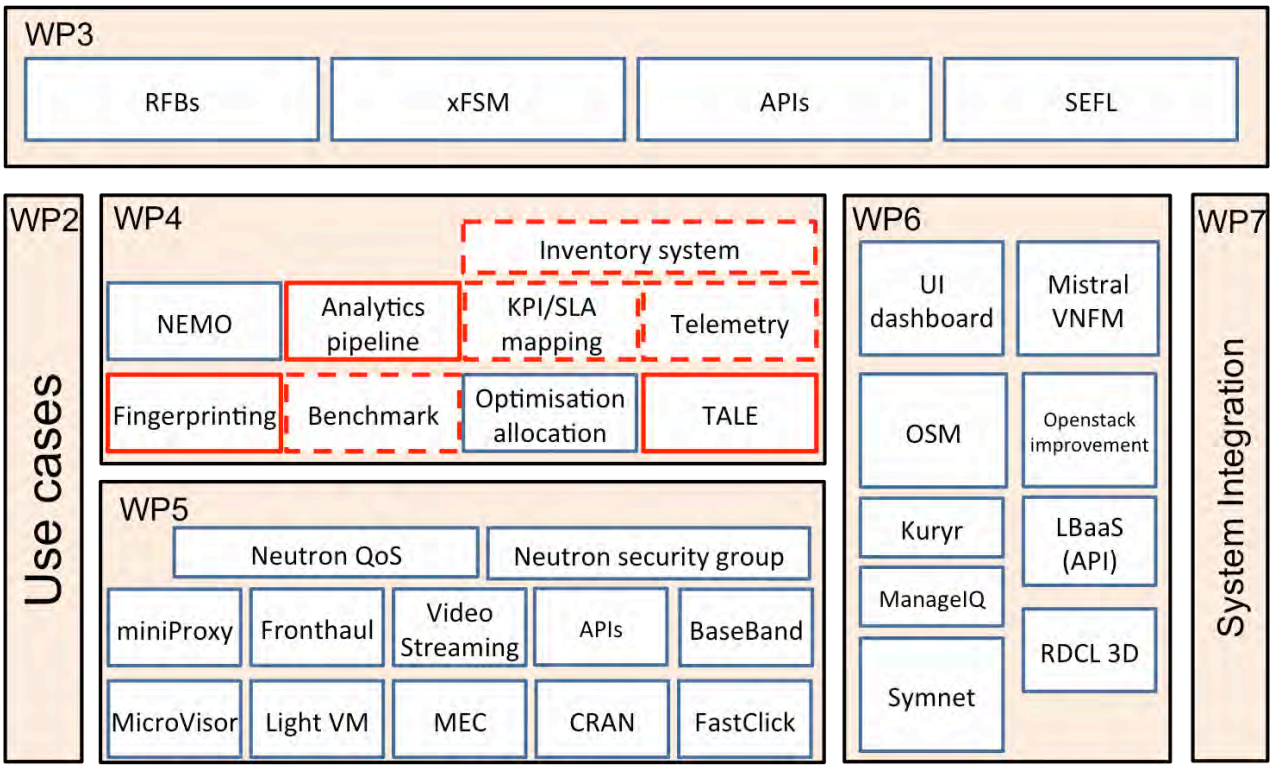


Figure 8: Graphical summary of Scene 2.a technologies

Off-line Phase (Characterisation and KPI Mapping)

The characterisation process starts by deploying the workload in an offline phase on the Superfluidity system using an automated characterisation framework developed in WP4. This framework has the responsibility for deploying the workload, executing predefined test cases (e.g. measurement of throughput etc.), application of workload traffic (e.g. HAMMER), collection of telemetry data, processing and persistence of the data and data modelling. The snap[1] telemetry framework provides telemetry data collection and is pre deployed on the Superfluidity system with a set of plugins to collect metrics from the layers within a deployment stack i.e. from the infrastructure layer up to the service layer. The specific plugins deployed depend on whether the workload is being deployed as a virtual machine, container or Unikernel.

The data is initially used to characterise the performance of the workload using the Throughput/Anomalies/Latency/Entropy (TALE) approach developed in WP4 to identify opportunities for optimisations e.g. improving throughput performance as shown in **Error! Reference source not found.**. Secondly, the characterisation process is used to identify metrics, which can be used for scaling triggers. This data is used as a key input in Scene 3.a to inform the

[1] http://snap-telemetry.io/

design and implementation of a scaling modelling for services deployed in the data centre testbed (i.e. Central testbed).

The second step of the characterisation process is to identify the metrics that most significantly influence the KPIs of the workload to be deployed. The metrics data collected during the execution of the test cases by the framework are ingested into an analytics pipeline. The analytics pipeline implements different techniques depending on the type of workload and relationships that are defined. For the Superfluidity demonstrator an ensemble approach (a mixture of statistical/clustering and machine learning approaches) is used to identify which platform metrics have the most significant influence on a workload's KPIs. This allows the number of metrics to be collected and monitored to be significantly reduced in the Superfluidity demonstrator i.e. reducing hundreds of metrics to tens of relevant metrics.

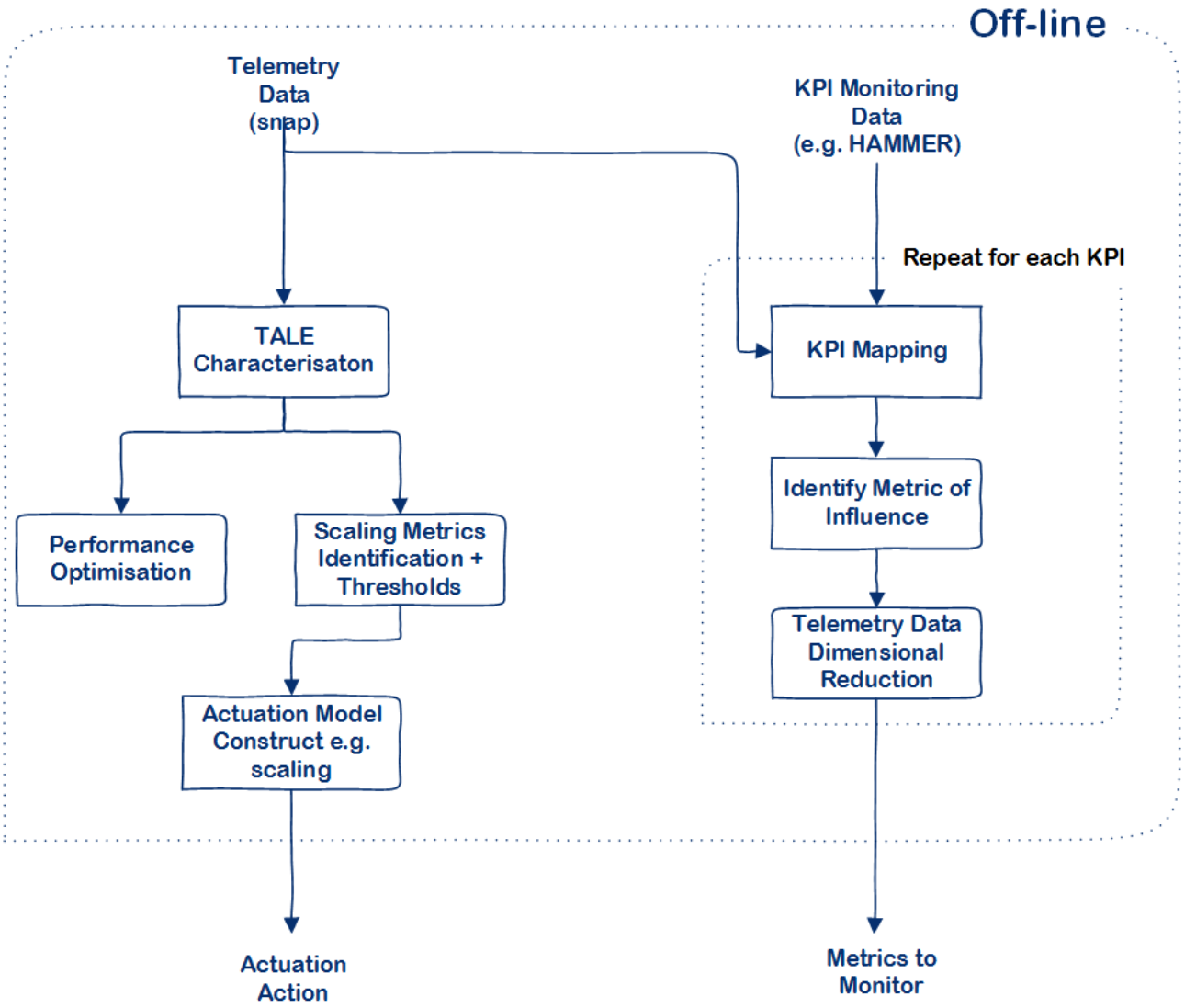


Figure 9: Superfluidity workload metrics identification

3.4.4 Implementation challenges for the demo

The main challenges for this scene are the following:

- The potential need to characterise workloads using different virtualisation approaches (VMs and Containers) and bare metal (Unikernels)
- Telemetry framework may need to support VMs, Containers and Unikernels (requirement for RFB platform – see Scenes 1a + 1b). Unikernels are particularly challenging as the minimal kernel footprint means that most of the standard Linux counters are not available and therefore Unikernels appear as essentially black boxes to a telemetry agents.
- Scaling model implementation – where to locate scaling logic i.e. as an extension to telemetry or implemented as part of the service orchestrator. The service orchestrator is the most appropriate. However, this would require customisation of the pre-existing solutions, which will be adopted by Superfluidity.

3.4.5 Evaluation plan and validation metrics

The relevant metrics to be evaluated in this scene are the following

- The user request arrival rate (number of TCP connections per second) has a significant effect on the stability of network throughput performance. The service exhibits throughput stability when the user arrival is less than or equal to the number of users the system can support per second. The specific number will be identified for the Superfluidity demonstrator.
- The maximum network throughput directly affects the number of users that can be supported. The specific threshold will be identified for the Superfluidity demonstrator.
- KPIs – Throughput and Latency

3.4.6 Relevant 5G-PPP metrics

The performance of the telemetry and associated analytics do not directly influence the 5G PPP metrics but rather have an indirect influence on them within the Superfluidity system.

- 10 to 100 times more connected devices
    - The telemetry system and the performance of the real-time analytics need to be scalable in order to support the 10 to 100 increase in the 5G network footprint i.e. 10 to 100 times more connected devices.
- 10 times to 100 times higher typical user data rate
    - Telemetry is a key tool in the process of service optimisation in order to achieve a 10 to 100-fold increase in user data rates. This will require service characterisation and optimisation prior to deployment followed by continuous monitoring and service rebalancing order to sustain the increase in user data rates.
- End-to-End latency of < 1ms.

- Telemetry and analytics will be a key tool in helping to achieve end-to-end latency of <1ms through the identification of bottlenecks and resource contention in the service stacks i.e. both hardware and software.

3.5 Scene 2.b: Streaming service deployment

3.5.1 Scene description

This scene deploys a video streaming function at some central point in the network or data centre. In addition, it deploys virtualised cache nodes, enabling a content delivery network. The goal of video streaming server is to provide transmuxing (format conversions), such as fragmented MPEG-4 supported in the ISO standard MPEG-DASH [7], or MPEG-2 TS as in Apple HTTP [6]. In addition, digital rights management is mandatory, using different schemas such as Widevine[2], PlayReady[3] etc. to support all the encryption schemes, media formats and so on it is best to use a video streaming server that handles all conversions on the fly, which is where we introduce Unified Origin[4] and the Unified Streaming Late Transmuxer (LTM)[5].

3.5.2 Technical trigger to start this scene

In traditional video streaming the server runs at the central cloud on a bare metal server. This server responds to user requests, possibly arriving through a networked cache, serving different devices with different protocols. Running the video streaming server virtualised in the central network introduces more flexibility as envisioned in the Superfluidity architecture. Contrary to the performance of running directly on bare metal hardware, the performance limits and requirements may be impacted by the virtualised configuration and implementation, and more work is needed to further understand scaling in this type of deployment.

3.5.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

[2] http://www.widevine.com/
[3] https://www.microsoft.com/PlayReady/
[4] http://www.unified-streaming.com/products/unified-origin
[5] http://www.unified-streaming.com/cases/late-transmuxing-improving-caching-video-streaming

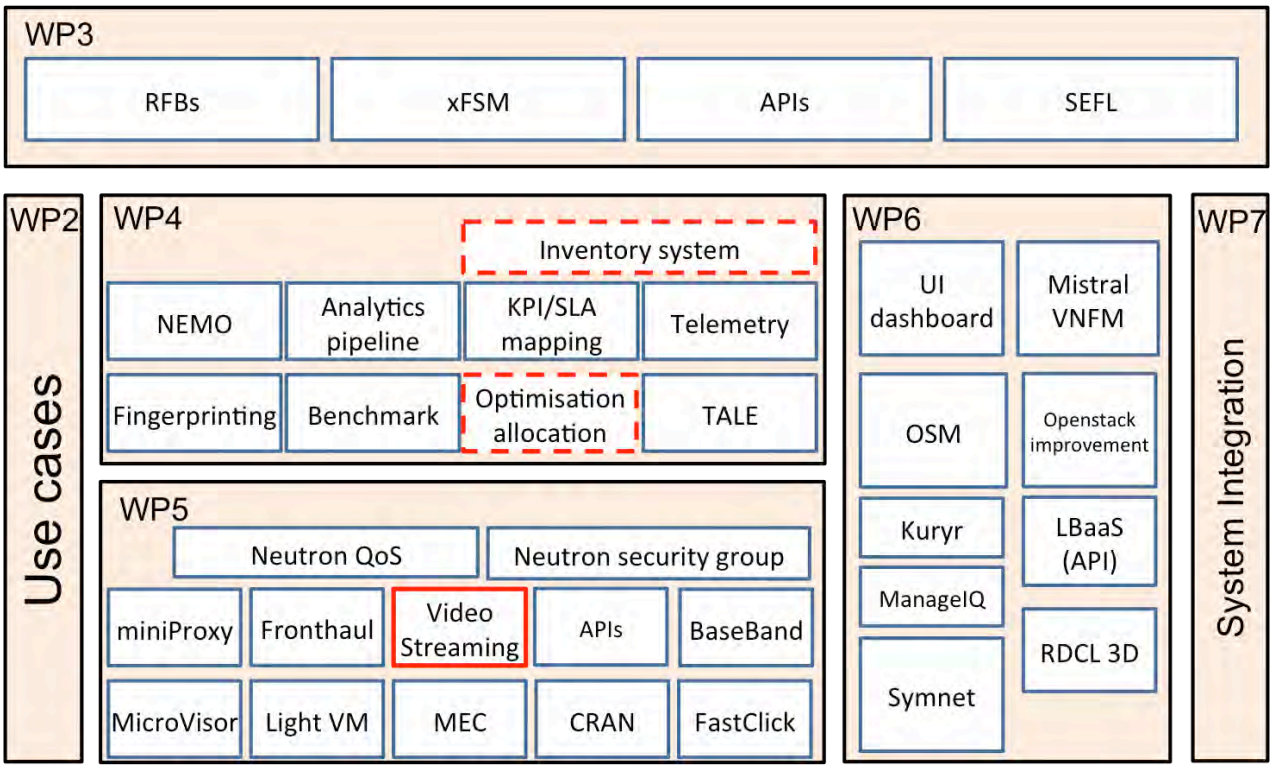


Figure 10: Graphical summary of Scene 2.b technologies

Superfluidity technology for virtualisation and profiling is important to enable this use case. Instances need to be provisioned reasonably quickly. Profiling technology is important to measure performance bounds and guarantee the desired performance of the implementation.

3.5.4 Implementation challenges for the demo

In this demo the video server and caching functionalities are split between different virtualised blocks. The block based decomposition enables separate deployment of the video streaming function and the cache nodes.

3.5.5 Evaluation plan and validation metrics

The system is evaluated using load testing using the Citrix Hammer [8] user workload simulation software. Key metrics for evaluation of streaming performance are the throughput achieved at the server and client and the latency perceived at the playback. More video specific video metrics exist typically derived from these fundamental metrics. For operators of video platforms these metrics are key. The server side KPI's defined can also be found in [9].

3.5.6 Relevant 5G-PPP metrics

A benefit of using a virtualised video streaming function and cache nodes makes video streaming possible in the 5G architecture, i.e. one of the envisioned use cases.

By using containers and virtualised workloads, supported with the orchestration framework, the video streaming workloads can be scaled according to the demand, reducing the overhead of continuous over-provision.

3.6 Scene 3.a: Central cloud services automatic scaling

3.6.1 Scene description

This scene is focused on the operational scaling of services that are deployed and running in the central cloud in an automated manner. The video streaming service from scene 2.b is the same as the one referenced here. There is a specific end-user, EU1, whom we are examining for the purpose of this description. The use case assumes though that there are multiple other users of the platform all performing operations similar to the EU1, who are collectively known as EUx. Other users that are using the operator for other services are labelled as neighbour users, NUx.

EU1 would like to watch a high quality video on demand, from a service provider, on their mobile device. EU1 would like to watch the video, free from any delays or interruptions and to watch the video without it being too low quality. In particular, EU1 is watching a video stream about a 5G EC Project. The video service provider utilises video streaming function and virtualised CDN for efficient delivery.

Other users of the infrastructure, the NUx, though are running machine-learning workloads and cause resource contention on the same physical server in the central cloud.

3.6.2 Technical trigger to start this scene

In this scene, we assume nothing about the topology of the network, the users and the workloads. The user, EUx is assumed to have an established video service as prepared in scene 2.b. The video streaming service provider though is using a Cloud based instance of their workload, which runs on an operator that also hosts multiple other tenants.

The NUx start increasing the network activity between two other Virtual Machine guests that are hosted on the same hardware as the video streaming (LTM) service. The infrastructure owner knows that the LTM provider pays more for a certain level of performance through pre-established service-level agreements. The owner also knows that the NUx workloads cannot be stopped or moved from that hardware due to certain hardware specific features being present.

3.6.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

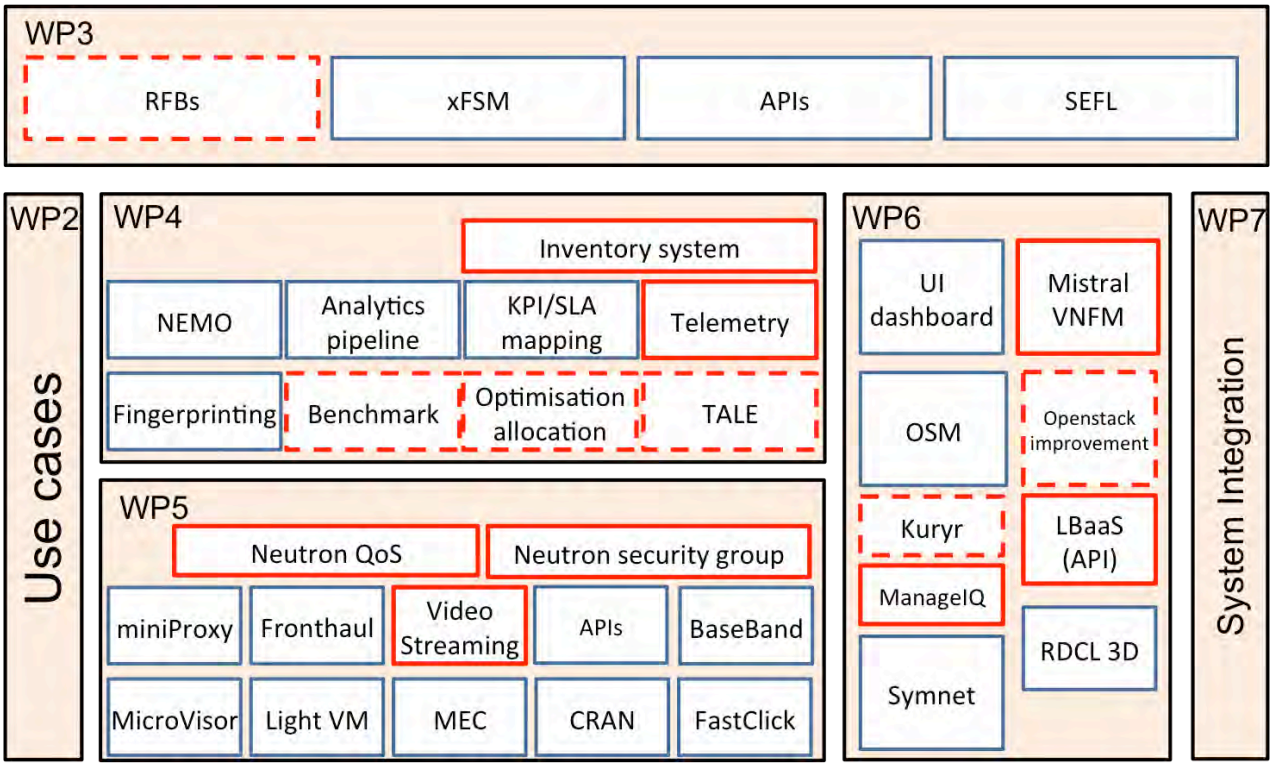


Figure 11: Graphical summary of Scene 3.a technologies

The network activity between the VMs on the same hardware causes a spike in key metrics associated with the LTM service, which are being monitored and, as such, the following actions are taken:

1. Snap telemetry framework is configured to collect the metrics identified in scene 2. The telemetry data is ingested by a scaling model. The model comprises of:
    a. The service level objectives (SLOs) which encompass the SLA that must be maintained;
    b. The KPIs for the SLOs; the platform metrics mapping to KPIs;
    c. The actuation thresholds for the metrics and the actuation actions.
2. A new LTM service is spawned on a second physical server at the central cloud
3. Temporarily a Quality of Service (QoS) policy is put into place on the hardware with the NUx and LTM workloads that reduces the maximum network activity
4. The load-balancer is reconfigured to redirect traffic to the new instance, LTM2, once it is ready
5. The QoS constraint put in place in step 2 is relaxed

3.6.4 Implementation challenges for the demo

Although an implementation of this scene already exists as Demo-1 (see Annex A: Initial demos), for it to be fully fledged we need to introduce more controls in the feedback loop. Currently we can monitor coarse-grained network activity. This only allows the capturing of overall data throughput, which is currently measurable between Virtual Machines on the same hardware. An overview of how the online telemetry can be fed back into the decision making process is provided in Figure 12.

- Need better telemetry input as acquired from SNAP. Using this framework we can then acquire lots of data about a lot of parameters.
- Selectively choosing which data to monitor and process also requires work, which is currently done in the offline analytics phase in scene 2.a. There is some effort needed to pipe the work from the offline analysis back into the orchestration framework.
- Orchestration improvements – to allow for fine-grained control of where we place workloads, will require improvements to the actions available, which are currently limited to start, pause, stop, move. By better tying in with the schedulers and orchestration system, we will be able to exact finer control.

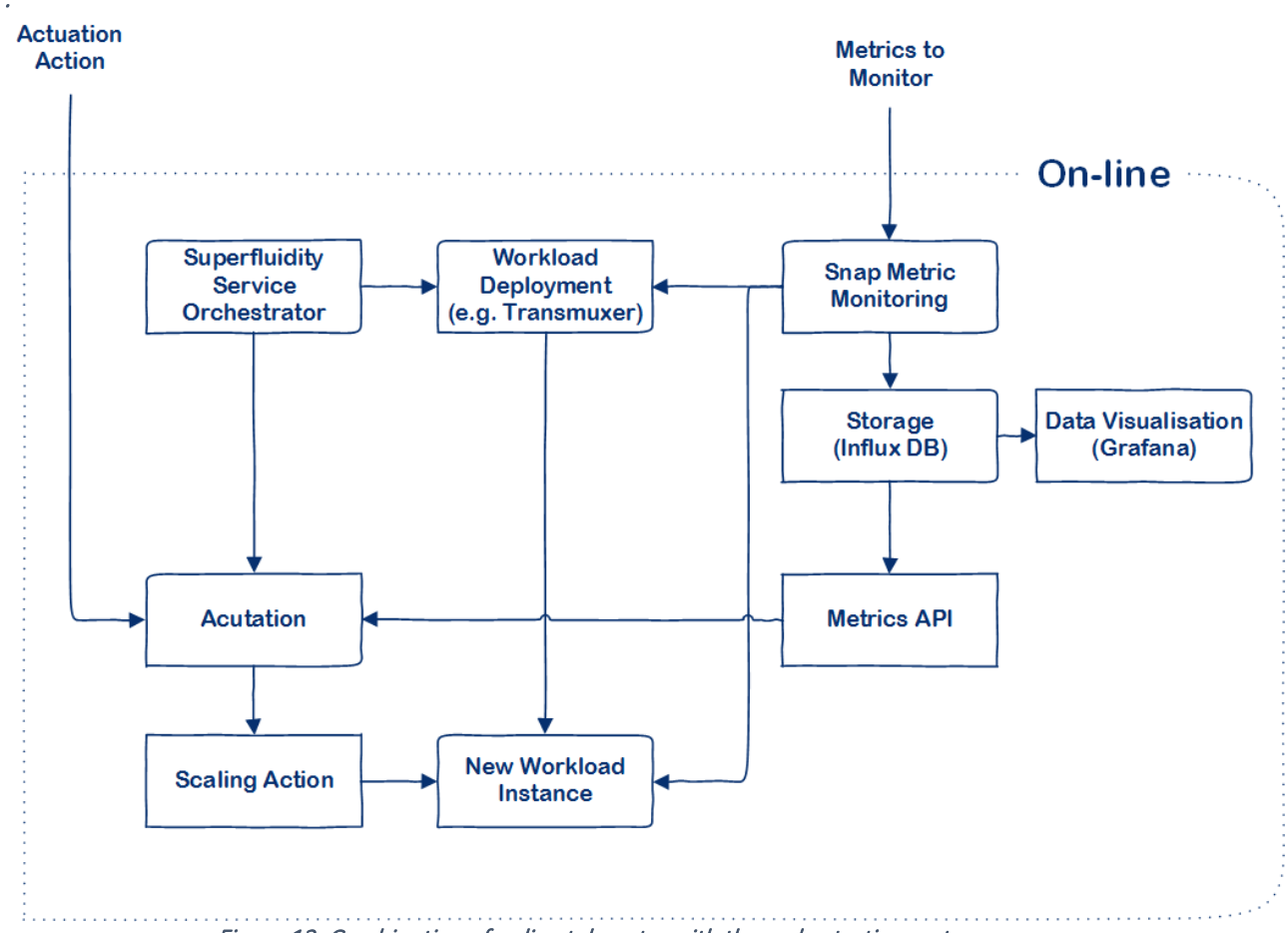


Figure 12: Combination of online telemetry with the orchestration system

3.6.5 Evaluation plan and validation metrics

The evaluation plan is to quantify, in terms of User-Experience, how the video streaming service performs in the scenario where Superfluidity features are not enabled versus when they are enabled.

We will be measuring how long the observation window should be for different types of metrics, including if we should capture any historical information for certain types of data.

The main evaluation though will come through changing the number of users utilising the workload, as simulated by Citrix Hammer. Hammer can be used to simulate a large number of users, using different types of net workloads, which in this case are simulating users also requesting the video service. Different users can be used to simulate different types of requirements, as specified in their profiles and the orchestrator can then be checked to see how it assigns users based on the different types of requests it is receiving.

Metrics include:

- Service provisioning time in order of seconds not days

Currently getting administrators to change the allocation of services is a manual task and takes a long time. Automating this will allow the orchestration system to cut down the time taken to react the service incidents and also to react to standard increase and decrease in demand.

- Energy saving – 10 times lower energy consumption

  By having a reactive system (with some possibly predictive elements), we can reduce the number of dormant workloads in the infrastructure. Each workload will be fine-tuned and much smaller in space, memory, and CPU requirements as they will be based on smaller VMs, containers and unikernels. In this use case we will be particularly focused on the size of the LTM instance. This is further explored in Scenes 3.c and 3.d.

- User experienced data rate

  With feedback from the LTM client on end-users equipment, we will also get feedback on the data rate actually achieved from users. We will be looking at seeing how to globally maximise this.

- Monitoring information

  To be able to make informed decisions, we will need to gather a set of metrics, which are relevant to the workloads. Without a-priori or offline analysis it is impossible without domain expert information to know which metrics should be monitored. By careful analysis we will be able to determine which metrics are more relevant for particular types of workloads. This will then go towards improving the workload models and is foreseen as an evolving task.

- Reaction time

For an action to be useful it has to be performed in a timeframe from when a condition was detected otherwise at best it won't influence the system and at worse make it unstable.

- Contextual information

  To make a correct decision on where workloads should be placed, the models will also need some environmental variables that are dynamic in nature. This contextual awareness will be important for gaining the maximum out of the system as possible.

- Service context analytics

  As per the monitoring metric, we will also need to determine the status of the various services that may be a combination of multiple different metrics. The analytics also implicitly suggest processing of data, which means that we will need to dedicate some resources for the processing of this information.

- Standard Network Node Level KPI

  For this particular workload we will be investigating network throughput and latency. However as the workloads become more complex and/or there are variants we will be expecting to also read other network parameters (including jitter and errors for example) for determining what the optimal service level involves.

- Quality of Experience Metrics-Analytics

  To be able to take more informed decisions, the system will also have to process QoE parameters. For instance a particular user may be suffering from a degraded level, which would otherwise be missed if only coarse grain monitoring is used.

- Scalability

  This scenario revolves around the ability to spawn new instances of the Unified Streaming Origin instances. The VIM, the monitoring system and all other constituent parts need to be able to scale to the number of workloads expected in 5G deployments. We therefore need a scalable system that can handle the amount of workloads and decisions that are expected. This includes being able to continue services in the order of ~100 ms (fast service migration).

- Load-balancer

  An integral part of this demonstration is being able to use a load-balancer to redirect between heavily loaded and lightly loaded or new services. This redirection forms part of the user experience, whereby the more seamless it is, the less chance that the user will discontinue using the service.

- Location independent scaling

  For a truly Superfluid system, the workloads should be executable anywhere in the infrastructure. This means that workloads should be able to run, as much as possible, on other types of hardware while being hardware agnostic. For performance reasons though, it is important that the properties of the hardware are known in case there is a way to


accelerate certain workloads. In this scenario, the new instance of Unified Streaming LateTransmuxing server can be set up on any of the compute node instances within an OpenStack managed framework.

- Manageability of the whole platform

  The manageability of the platform is important for this scenario as we can only place resources and migrate them within the scope of a single VIM currently. This becomes increasingly important in Scene 6 where we will manage both VMs and containers together.

- Infrastructure scalability

  For our platform and demo to grow we need the ability to add new hardware into the deployment and for this then to be accessible by the VIM and the rest of the system, without needing the whole existing infrastructure to suffer degradation or temporary non-availability.

3.6.6 Relevant 5G-PPP metrics

- 10 to 100 times more connected devices.
  - By moving the processing and decision making towards the end-points of the 5G network we expect that we will be able to support many more users.
- 10 times lower energy consumption.
  - By only provisioning workloads as they are necessary, rather than having to have them pre-provisioned we expect to massively decrease the amount of idle power consumption used.
  - Supporting lower cost access to a wider spectrum of applications and services

The Superfluidity platform as described in this scene enables many more workloads to run on a given set of hardware, which will increase the competitiveness and reduce costs for those services.

- Reduction of the network management OpEx by at least 20% compared to today

A large part of this scenario revolves around moving away from having an administrator that needs to manually intervene for many different types of routine and expected conditions. It will also help to more efficiently utilise the resources, meaning less hardware will be needed and providing a further reduction in management and power OpEx.

3.7 Scene 3.b: Services relocation to edge cloud

3.7.1 Scene description

This scene demonstrates the migration of a service from the central to the edge cloud. The service is initially completely provided from the central cloud and, at a certain point in time, based on service operator policies, some of the service components are relocated to the edge, serving the customer from this point onwards. This migration process is achieved by using the MEC (*Multi-access Edge Computing*) solution and this process is transparent to the customer, who does not experience any service disruption.

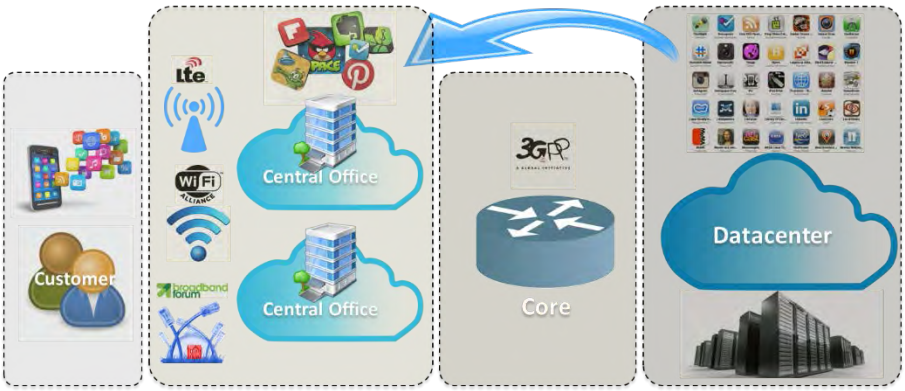


Figure 13: MEC concept

3.7.2 Technical trigger to start this scene

A user uses his Web browser to watch a video, streaming it from the network. This service is provided from a centralised streaming server deployed at the central cloud.

In this scene, the operator's policy for this streaming service states that:

- "Whenever a user requests a video streaming service from a CRAN edge with MEC support, it is worthwhile to deploy an LTM ME App at that edge" (*Late TransMuxing Mobile Edge*)

This policy aims to minimise the amount of traffic traversing the central cloud, reducing backhaul bandwidth, increasing bandwidth efficiency and ensuring a good quality of service for the users.

Since this is associated to caching services, the network and other future users will also benefit from having the content at the edge in raw format, independently from the platform used to access that content.

3.7.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

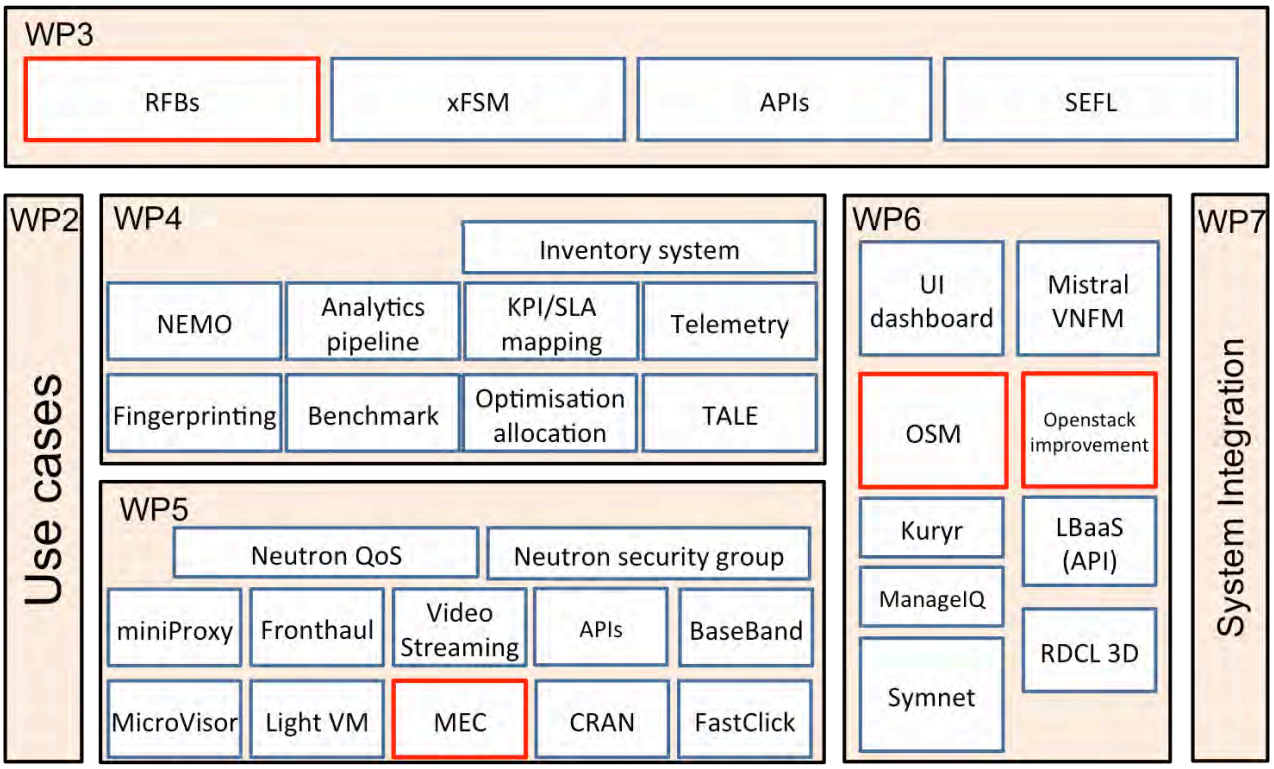


Figure 14: Graphical summary of Scene 3.b technologies

The sequence of steps performed in this Scene is as follows:

1. The LTM Manager receives notifications from the Central Unified Origin Application anytime a user starts or stops a session
2. The RNIS Core continuously collects information about the user location (edge/cell) from edge RNIS services
3. The Session Analyser analyses User Notifications and User Locations and detects where a user starts a session from a given edge
4. A new user starts a video session from an edge
5. The LTM Manager triggers the instantiation of an LTM App on this new edge and offloads the appropriate traffic to the edge
6. Once the process is completed, the user starts consuming the video directly from the LTM, which in turn gets the raw video from the Backend
7. After some time, the user stops consuming traffic, leaving the service (assuming they are the only user watching video at this edge)
8. The LTM Manager removes the offloading rules from the TOF and triggers the instantiation of the LTM Application on this edge

The following Figure depicts this sequence.

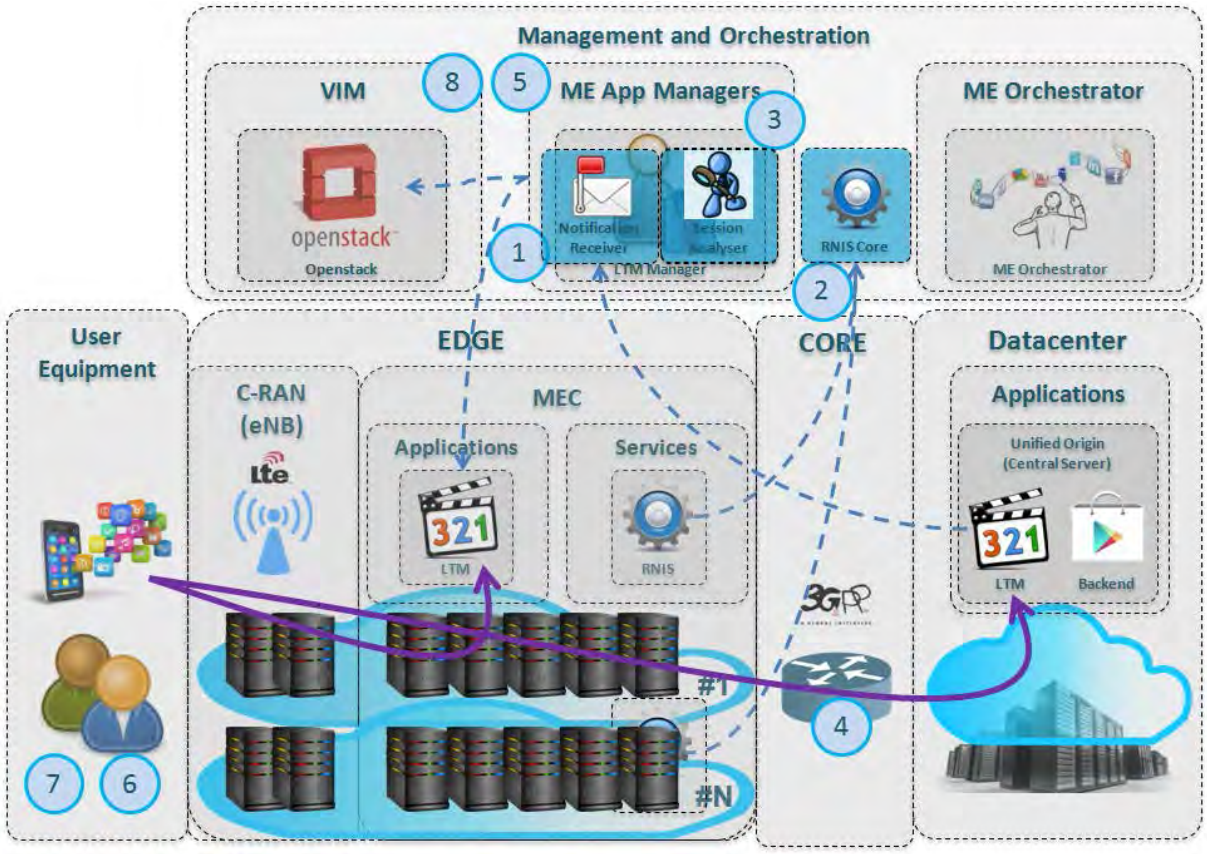


Figure 15: Scene 3.b workflow

The solution here proposed requires a number of new features, beyond the current MEC scope:

- Service level entity analysing video streaming sessions;
- A MEC control entity, based on standard RNIS (*Radio Network Information Service*), gathering and making available UE's locations.

3.7.4 Implementation challenges for the demo

The main challenges for this scene are the following:

- Detection of one or more users requesting video content from the same edge. This requires the development of new components and the integration with the MANO component
- Production of light weighted and fast instantiation of the LTM ME App at the edge, and the provision of the TOF in order to offload the traffic to the edge ME App. This process shall ensure service continuity, making this process seamless for the user and for the mobile network
- Deploying the transmuxing and caching functions as RFBs. The original content is cached and then the transmuxer generates segments on the fly

3.7.5 Evaluation plan and validation metrics

The most relevant metrics to be evaluated in this Scene are the following:

- Instantiation time
  - Time elapsed between the decision that a new ME App needs to be instantiated on a given edge and the time where the ME App is fully instantiated and operational (in seconds)
    Note: Addresses Superfluidity references to "Instantiate services on-the-fly" and "Shift them transparently to different locations"
- Bandwidth savings
  - Amount of bandwidth saved by using the edge transmuxing technology (in Mbps or %)
- Latency reduction
  - Latency reduction by using the edge technology (in ms or %)
- Service continuity
  - Amount of time where the service is down (in seconds)
- Cache storage reduction
  - Media segments caching only on the edge (in MB or %)

3.7.6 Relevant 5G-PPP metrics

- End-to-End latency of < 1ms
  - MEC contributes significantly to the end-to-end latency reduction, by getting applications closer to the end users, serving them from the edge.
- Service deployment time < 90 min
  - In this particular case, service deployment time at the edge is significantly lower than the target of 90 min. It shall be in the order of seconds.
- 10 times lower energy consumption
  - MEC contributes to reduce the energy consumption, by deploying applications at the edge only when they are required, making a more efficient use of the infrastructure resources.
- 10 times to 100 times higher typical user data rate
  - MEC contributes to the increase of the user data rate by making an efficient use of the bandwidth, in particular the backhaul, between the RAN (Radio Access Network) and the Central cloud.

3.8 Scene 3.c: Container based services deployment at the edge cloud

3.8.1 Scene description

This scene assumes that the initial starting conditions are like those presented in scene 3.a. The difference though is that the content, instead of a single video from a popular 5G project, it adds a collection of advertisements mixed into the video. The difference with advertisements as opposed

to video content is that they have a much higher value for the publisher and the advertisements can potentially be targeted at individual users.

3.8.2 Technical trigger to start this scene

A user from scene 3.a is starting the next video and in before it starts (in between) an advertisement is played from a third-party site, which uses a specialised software provided by the Unified Streaming Platform.

The user, EUx, doesn't have any knowledge or choice in the advertisement that is received but triggers the advertisement action once a video is chosen based on Unified Remix[6], a technology developed by the Unified Streaming Platform for content insertion in the streaming media server.

3.8.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

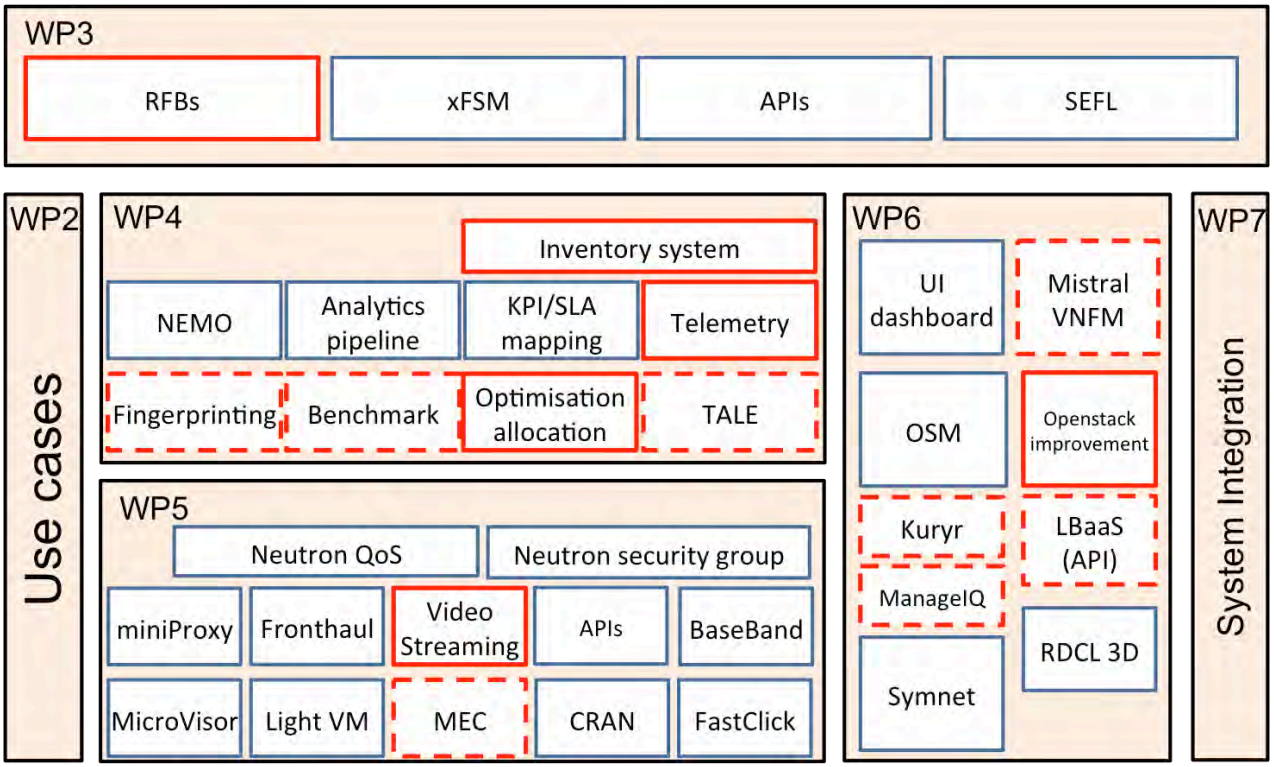


Figure 16: Graphical summary of Scene 3.c technologies

[6] http://www.unified-streaming.com/products/unified-remix

Once the advertisement action has been triggered it is important to quickly start up the service based on Unified Remix being close to the end-user and offering the personalised advertisement content.

Rather than triggering a VM with LTM capabilities, instead a lightweight version of Remix is instantiated near the end-user, which starts up in ~300-500ms. This service then handles the interactions related to the advertisement and the video streaming from the edge.

3.8.4 Implementation challenges for the demo

There is a need to transfer the container images for the Unified Remix instance from the Central cloud to the edge location (rapidly) and start it. The application should then pick up some of the configuration and state data from the end user to integrate it with the video streaming. This requires changes on the application side to enable the interactive advertisement such as implementation of RFB's in containers.

The LTM instance will be chained to other instances needed for Unified Remix to work. The Unified Remix instances will ideally be located close to the end user and relocatable if necessary and as such will be implemented using Containers rather than VMs. To get the VMs to interact with Containers and vice versa without having to lose significant performance using multiple layers of encapsulation we need to use Kuryr. Kuryr provides the network bridging fabric that allows VMs to interact with the containers.

The other technical challenge faced in this use case is that we start to introduce multi-cloud support where we have a central cloud creating instances on a remote edge cloud using Manage IQ and Ansible playbooks. The ports that are created need to be exported back up through Ansible / Heat and to ManageIQ so it can link up the ports for each of the connected instances.

3.8.5 Evaluation plan and validation metrics

The time taken to create the service from a cold and warm start will be evaluated to understand how long the service takes to start in real-world settings. The latency between the advertisement and the VM / container running the image will also be evaluated. The overall utilisation of resources will be monitored and compared to see how well the system performs compared to traditional legacy advertisement streaming setups.

3.8.6 Relevant 5G-PPP metrics

The following metrics were identified as relevant to take into consideration for this scene:

- Latency between end user and service
- Service provisioning time (<1s)
- Using a-priori set up services vs. on the fly deployments of personalised service applications

- Service density
- Higher user data rate
- Higher mobile data volume

By allowing services to start up on the fly, the number of services available to an end-user can increase by a large proportion, even if in fact they are not on all the time.

3.9 Scene 3.d: Unikernel based services deployment at the edge cloud

3.9.1 Scene description

This scene demonstrates the use of stateful SDN and the rapid deployment of network functions in the MEC to mitigate a DDoS attack. In this scene the DDoS attack is carried out at the edge network. A stateful SDN switch (based on OPP) is programmed with an xFSM that provides a first level of mitigation of the attack, which can be triggered immediately without the intervention of an SDN controller. A second level of mitigation of the attack is provided by the fast de-instantiation of LTM VMs and the instantiation of Unikernel-based VNFs in the MEC. The latter are VNFs that can analyse packet flows in more detail and are able to drop packets belonging to the DDoS packet flow.

3.9.2 Technical trigger to start this scene

At the beginning of this scene, video streaming is using late transmuxing in the MEC: several LTM transmuxing VMs have been instantiated and the traffic offloader is diverting video streams towards these. Moreover, the stateful SDN switch deployed before the traffic offloader allows packet flows through it.

Then a DDoS attack floods the network from the edge: legitimate user traffic is interleaved with DDoS packets. This attack can saturate the available network capacity and potentially cause service disruptions.

3.9.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

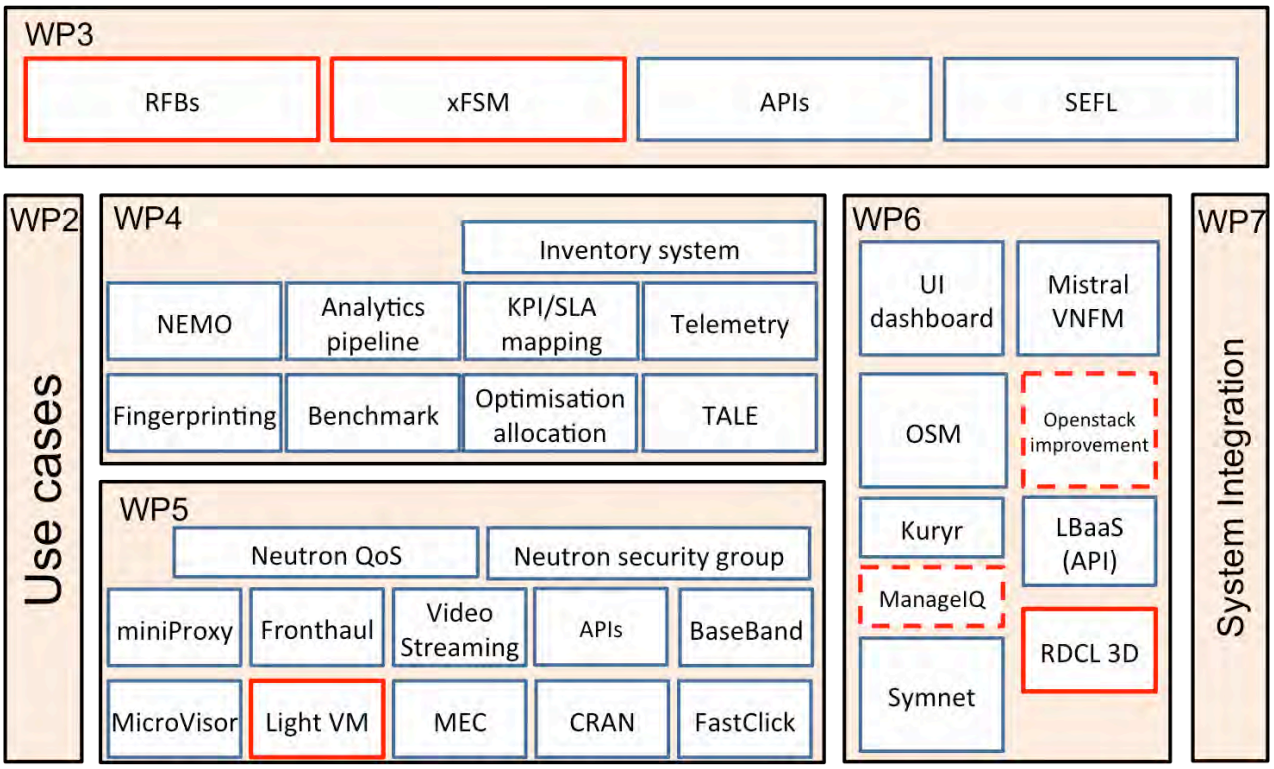


Figure 17: Graphical summary of Scene 3.d technologies

In this scene, an instance of the OPP stateful SDN switch is deployed between the edge of the network and the TOF (see Figure 18). When it detects a DDoS attack, the OPP (programmed through an xFSM) immediately implements countermeasures to mitigate the attack and then triggers the orchestrator, which performs a sequence of actions:

- Instantiates Unikernel VNFs in the MEC infrastructure. If there are insufficient resources available in the MEC infrastructure because these are allocated to transmuxing VNFs, the video flows which are traversing the MEC are redirected to the central cloud of the network and the resources used by the transmuxing VNFs in the MEC infrastructure are freed. Note that the video service should not be disrupted by this action;
- Instructs the TOF to redirect the traffic containing the DDoS attack to the Unikernel VNFs in the MEC cloud. The Unikernel VNFs are more effective than the OPP in countering the DDoS attack, but their instantiation, although very fast (in the order of milliseconds), requires intervention from the orchestrator. When the above actions have completed, the OPP allows traffic to flow again transparently through it.

Note that when the OPP detects a DDoS attack it immediately (the reaction times are in the order of the microseconds) starts reacting to the DDoS attack, even before contacting the orchestrator. This allows for an immediate mitigation of the attack.

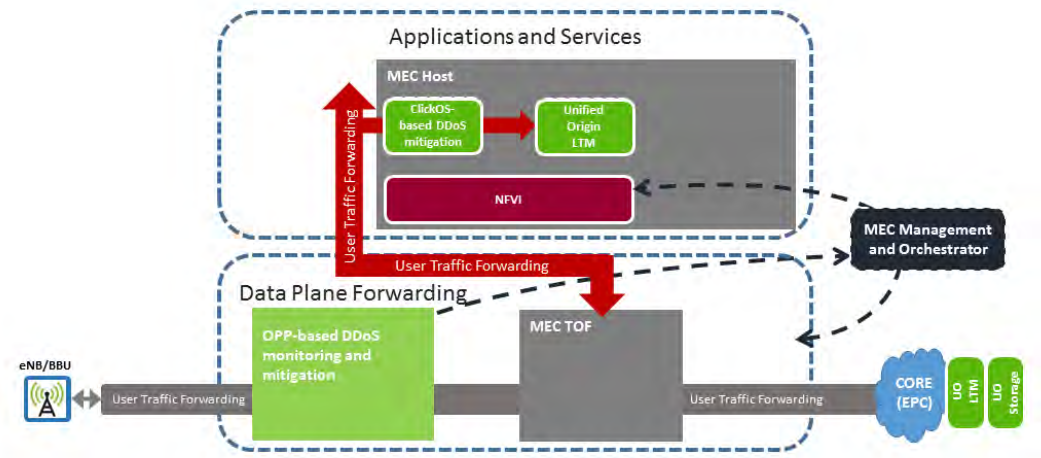


Figure 18: Scene 3.d overview

The following two subsections describe the new Superfluidity components that are introduced in this scene.

3.9.3.1 Unikernel based VNFs

Unikernels are virtual machines based on a minimalistic OS and tailored to a single application. Unikernels make an efficient use of computing and memory resources and can achieve orders of magnitude faster instantiation and boot times than paravirtualised or fully virtualised VMs.

In this scene, we make use of Unikernel-based VNFs which contain the logic to detect and block (or eventually mitigate) DDoS attacks.

3.9.3.2 Open Packet Processor (OPP)

The Open Packet Processor (OPP) is a stateful SDN switch that allows the execution of extended finite state machine (xFSM) directly at the dataplane without interaction with an external controller.

OPP supports the execution of per-flow programmable XFSMs by providing the following features: 1) programmable definition of a flow; 2) flow state retrieval and update; 3) per-flow registers (data variables) update; 4) enabling function definition; 5) evaluation of conditions to trigger per-flow state transitions.

In this scene, one (or more) OPP switch(es) are employed in the first stage of the DDoS mitigation: the OPP switches are programmed to monitor network traffic at the edge, in order to detect a DDoS attack and proactively mitigate it by dropping a portion of the traffic. A major advantage of using OPP in such scenario is the possibility of tracking and reacting to given events without the intervention of a controller, avoiding communication delays and the creation of potential bottlenecks.

It is worth noting that an external orchestrator will still be required. Indeed, in our envisioned scenario, the OPP distributed switches will be able to asynchronously notify the management and orchestration plane about suspicious activities. Thus the orchestrator will be able to create on demand Unikernel virtual machines towards which the switch(es) will reroute the traffic for the second stage of the DDoS mitigation.

An alternative deployment location for the OPP switch could be as a virtualised switch within the MEC. This would avoid the need for the deployment of components outside the MEC.

3.9.4 Implementation challenges for the demo

- The MEC server(s) should support the coexistence of Unikernel VMs and QEMU VMs. In this scene the resources employed by Unified Origin VMs in the MEC are freed to allow the instantiation of Unikernel VMs. The Unified Origin VM is available as a QEMU VM and thus the employed hypervisor should support both QEMU and Unikernel VMs.
- Implementation of the DDoS mitigation actions inside the Unikernel VMs. The logic required for the online analysis of traffic flows and the efficient detection of all types of DDoS attacks appears non-trivial.

3.9.5 Evaluation plan and validation metrics

The most relevant metrics to be evaluated in this scene are the following:

- Reaction time
  - The time elapsed between the start of the DDoS attack and the actual countermeasures taken by the OPP and Unikernel VNFs
- Instantiation time
  - The time needed to instantiate the Unikernel-based VNFs
- Traffic loss
  - The video traffic lost in switching the traffic from the central to the edge cloud (and back)

3.9.6 Relevant 5G-PPP metrics

- Aggregate Service Reliability ≥ 99.999%
  - Mitigating or countering DDoS attacks improves the reliability of the service

3.10 Scene 3.e: Services' optimisation at the edge cloud

3.10.1 Scene description

The objective of this scene variation is to validate that a commercially available, operator-grade, Application Delivery Controller (ADC) [1], Citrix NetScaler [2], can be deployed at the Superfluidity edge cloud platform, to enable migration of mobile network services, such as TCP Optimisation, Encrypted Video Traffic Management, Content Filtering, which are traditionally deployed behind the Packet Gateway (P-GW), to the Mobile Edge Computing (MEC) subsystem of Superfluidity.

Scene 3.e is an extension of the other scenes of this group, from the standpoint that it validates additional network services, potentially also chaining them, increasing the workload diversity.

3.10.2 Technical trigger to start this scene

The technical trigger for scene 3.e, as an extension to the other scenes of this group, is the consumption of new services by the end-user (mobile subscriber). Specifically, and depending on the network services that we decide to deploy:

- TCP Optimisation (top priority): This is transparently triggered when the end-user consumes services that utilise TCP as the transport layer protocol.
- Encrypted Video Traffic Management (medium priority): This is triggered by the end-user requesting a video stream from an OTT provider that delivers video using an encrypted transport, e.g. YouTube, Facebook or Netflix.
- Content Filtering (low priority): This is triggered by the end-user requesting content that is not allowed, either based on parental profile (not appropriate) or legislation (illegal).

3.10.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

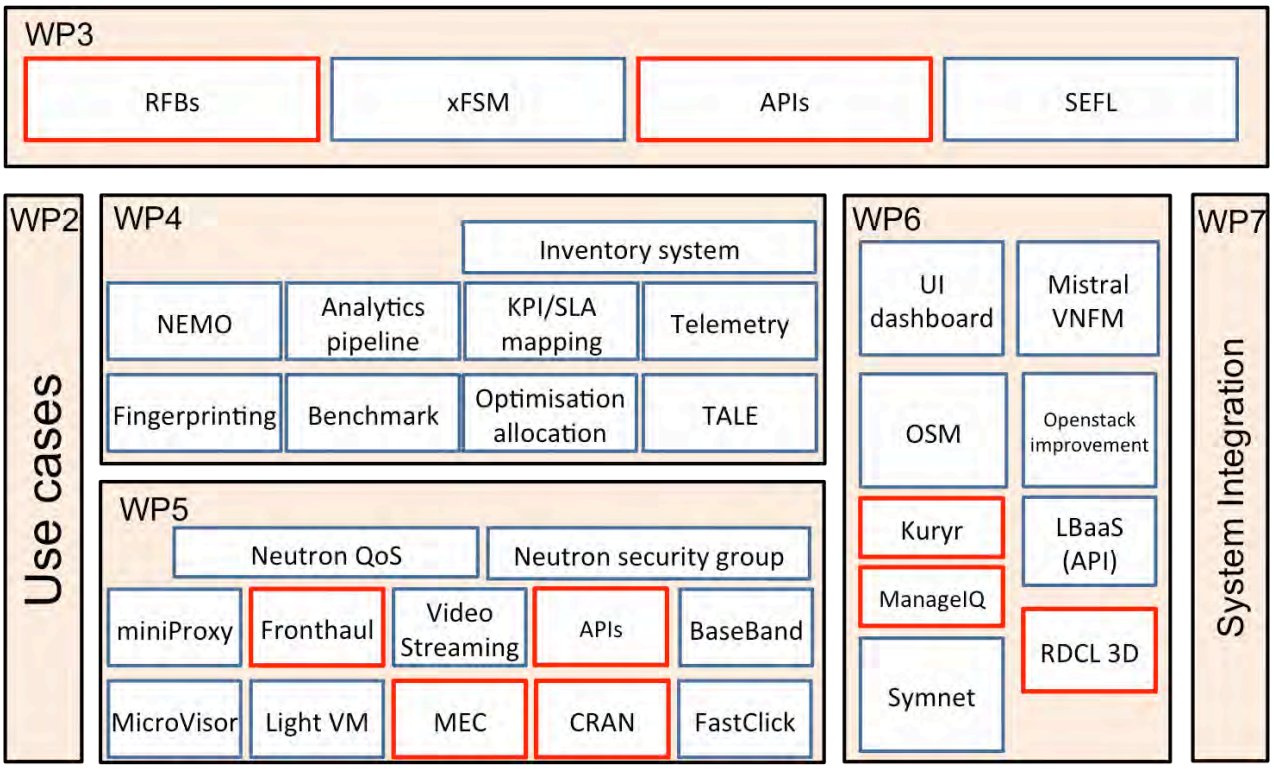


Figure 19: Graphical summary of Scene 3.e technologies

The delivery of diverse transport- (Layer 4) and application- (Layer 7) layer services, such as the ones proposed above, traditionally involves deploying large/specialised network appliances, which are installed in centralised data centres, behind the P-GW, i.e. in the S/Gi-LAN. Such an implementation is the Citrix ByteMobile ATM [4].

In scene 3.e, the Layer 4 (TCP Optimisation) and Layer 7 (Encrypted Video Traffic Management and Content Filtering) services are deployed in smaller/softwarised forms and closer to the mobile subscriber, in the Mobile Edge Computing (MEC) subsystem of Superfluidity, which was described in scene 1.b.

Scene 3.e will require the deployment of NetScaler ADC instances. Contrary to the infrastructure LB scenario that follows (scene 4.a, where NetScaler VPX is used), in scene 3.e the Docker-container version will be preferable (NetScaler CPX). This is to fulfil the fast and granular service instantiation requirements, considered vitally important for MEC. In that context:

- The NetScaler instances that will implement the TCP Optimisation, Encrypted Video Traffic Management and Content Filtering services will have to be deployed on the MEC subsystem of Superfluidity, and integrated with the end-to-end UE – CRAN – eNodeB – vEPC setup.
- Given that TCP Optimisation operates complementarily to services that use TCP as network transport, including Late Transmuxing, demonstrating it will require traffic steering or service function chaining capabilities.

- The orchestration aspects of deploying the new service instances and chaining the network services will have to be implemented, utilising the respective Superfluidity components.

Lastly, demonstrating the new services may require adding new content assets and implementing new traffic profiles in the traffic generator used by Superfluidity (Citrix Hammer).

3.10.4 Implementation challenges for the demo

In certain scenarios, this scene may involve chaining services implemented as monolithic VMs with services implemented as containers. To facilitate this, recent additions to OpenStack, such as Kuryr, will have to be integrated to implement the networking integration between service instances.

One of the key objectives of Superfluidity is to compare traditional in-network services with next generation implementations that utilise lightweight virtualisation schemes, data plane acceleration technologies, etc. Using the Docker-container flavour (NetScaler CPX) is a step in this direction, but we would like to compare it against other Superfluidity innovations, such as MiniProxy or MiddleClick, that promise even faster service instance instantiation times, more efficient packet processing, smaller injection overhead, higher scalability, etc.

For evaluating services like TCP optimisation, we will have to simulate realistic network conditions. The end-to-end UE – CRAN – eNodeB – vEPC setup implements an idealised mobile network environment. To validate the effectiveness of TCP optimisation, we will have to introduce network impediments, such as delay, jitter, loss, etc., either caused by radio bearer behaviours, scheduler policies or network congestion.

This TCP optimisation requires additional MEC ToF traffic handling rules (pass-through) at the middle of the S1-U GTP tunnels. This functionality will be added to the project MEC implementation. This differs from scene 3.b where the LTM application will be a traffic terminating one, with a local breakout for a high capacity transport network.

3.10.5 Evaluation plan and validation metrics

The evaluation plan of Scene 3.e will assess the services that are introduced, specifically in comparison to the scenario they are deployed in the traditional fashion, i.e. in S/Gi-LAN. Due to practical limitations (i.e. inability to fully replicate the traditional architecture), this will most likely will be in the form of a qualitative evaluation.

3.10.6 Relevant 5G-PPP metrics

- Service Creation Time Reduction
- Network Latency Reduction
- Network Throughput Increase
- Energy Consumption Reduction

3.11 Scene 4.a: Alternative load balancing function at the central cloud

3.11.1 Scene description

The objective of this scene variation is to validate that a commercially available, operator-grade, Application Delivery Controller (ADC) [1], Citrix NetScaler [2], can be deployed at the Superfluidity central cloud platform, as an alternative to the basic load balancing backend included with the Virtual Infrastructure Manager (OpenStack: HAProxy or Octavia). This demonstrates some of the key architectural concepts of Superfluidity including RFB service decomposition and the ability to interchange different VNFs that perform the same function.

Scene 4.a is an enhancement of scene 3.a (Section 3.6), in the sense that it aims to improve the load balancing and network analytics aspects.

3.11.2 Technical trigger to start this scene

The technical trigger for scene 4.a is the same as for scene 3.a: In reaction to an SLA/KPI degradation, the orchestrator initiates a scale-out action, new service instance(s) are thus deployed, and the infrastructure Load Balancer (LB) will have to direct/dissect traffic to them. The difference from scene 3.a is that, instead of the load balancing backend used by default by the Virtual Infrastructure Manager (OpenStack: HAProxy), Citrix NetScaler ADC [2] will be used.

3.11.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

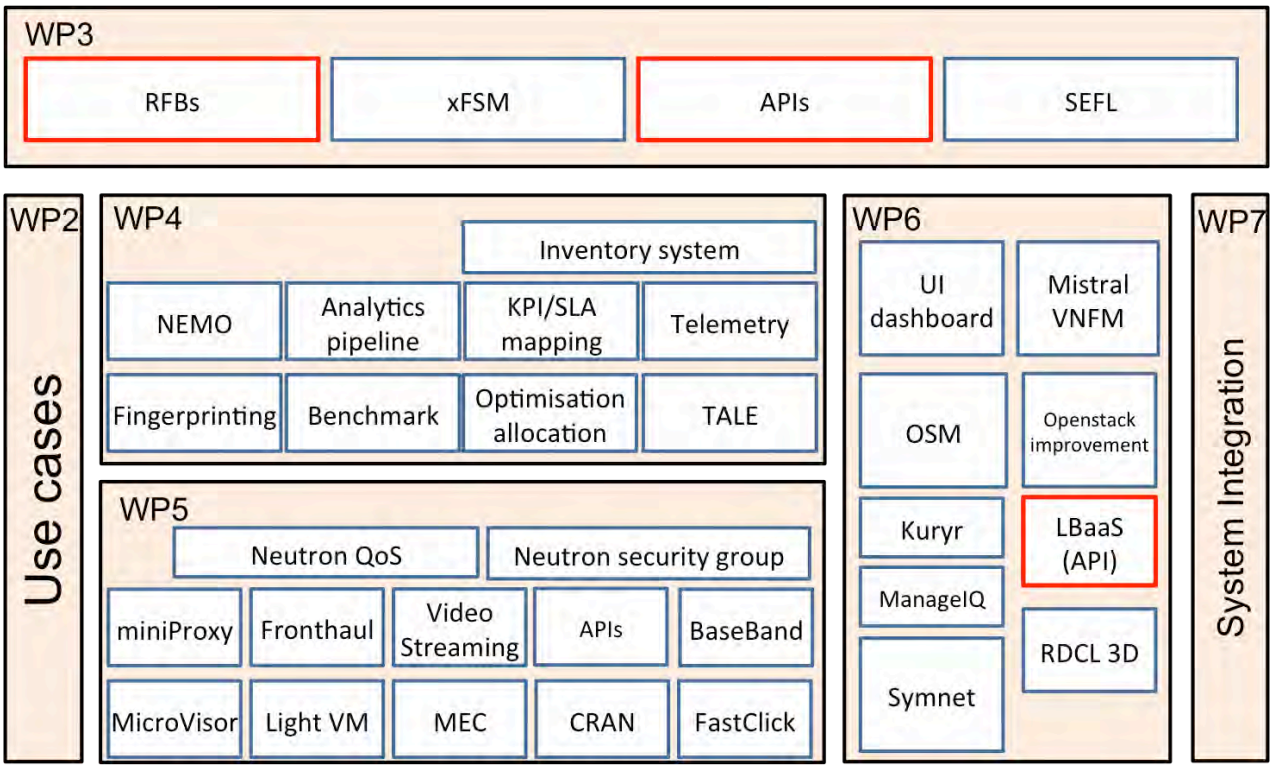


Figure 20: Graphical summary of Scene 4.a technologies

Scene 4.a requires the following additions to the Superfluidity platform:

- Deployment of NetScaler ADC instance.
- Deployment of NetScaler Management and Analytics System (MAS) [3], as the Element Management System (EMS) for NetScaler ADC.
- Deployment of the Neutron LBaaS (Load Balancer as a Service) driver that is required for integrating OpenStack with NetScaler ADC (as infrastructure load balancer). NetScaler MAS provides a function that automates this integration.

To facilitate comparisons against scene 3.a, the NetScaler LBaaS driver will have to be deployed side-by-side with the existing HAProxy LBaaS driver.

These new instances (NetScaler ADC and MAS) will be deployed as VMs, most probably the flavours for KVM.

Lastly, demonstrating the new services may require adding new content assets and implementing new traffic profiles in the traffic generator used by Superfluidity (Citrix Hammer).

3.11.4 Implementation challenges for the demo

Feasibility of this scene has been already verified in the (Citrix premises) OpenStack Reference Lab. The only risk is potential integration challenges with recently introduced OpenStack Ocata release that is used across the integrated demonstrator testbeds.

3.11.5 Evaluation plan and validation metrics

Scene 4.a will be evaluated in comparison to scene 3.a, focusing on the functionality and performance differences between using an operator-grade infrastructure load balancer and a basic implementation. In that context, the same validation metrics will be assessed, particularly in the areas of load balancing and network monitoring, and will be evaluated comparatively to the respective validation metrics of scene 3.a.

3.11.6 Relevant 5G-PPP metrics

- Reliability improvement

3.12 Scene 4.b: Alternative edge offloading function

3.12.1 Scene description

This scene is a variant of scene 3.b with an alternative implementation of the *traffic offloading function* (TOF) function of the MEC platform. Rather than using a VM as in scene 3.b, the TOF function is provided by a FastClick configuration.

This scene illustrates the modularity of the Superfluidity platform, where an abstract *reusable function block* (RFB) can support several concrete implementations. It will also demonstrate FastClick automatic resource allocation capabilities. Finally, it might also demonstrate some performance improvement brought by FastClick.

3.12.2 Technical trigger to start this scene

In this scene, we refer to a change in the availability of technologies: new technologies at the data plane level become available for realising the same scenario. The introduction of the new technologies is facilitated by the fact that the service is described in terms of RDCL (*RFB Description and Composition Languages*). The different solutions can coexist in different locations in the infrastructure or even in the same location, if they can rely on the same physical resources provided by the NFVI.

In particular, the idea is that the TOF is described in terms of RDCL and then (re)-implemented using FastClick and perhaps also OPP (this is optional, as OPP is already demonstrated in scene 3.d).

3.12.3 Technical description and involvement of Superfluidity technology

The following Figure refers to the map of technologies used in this scene.

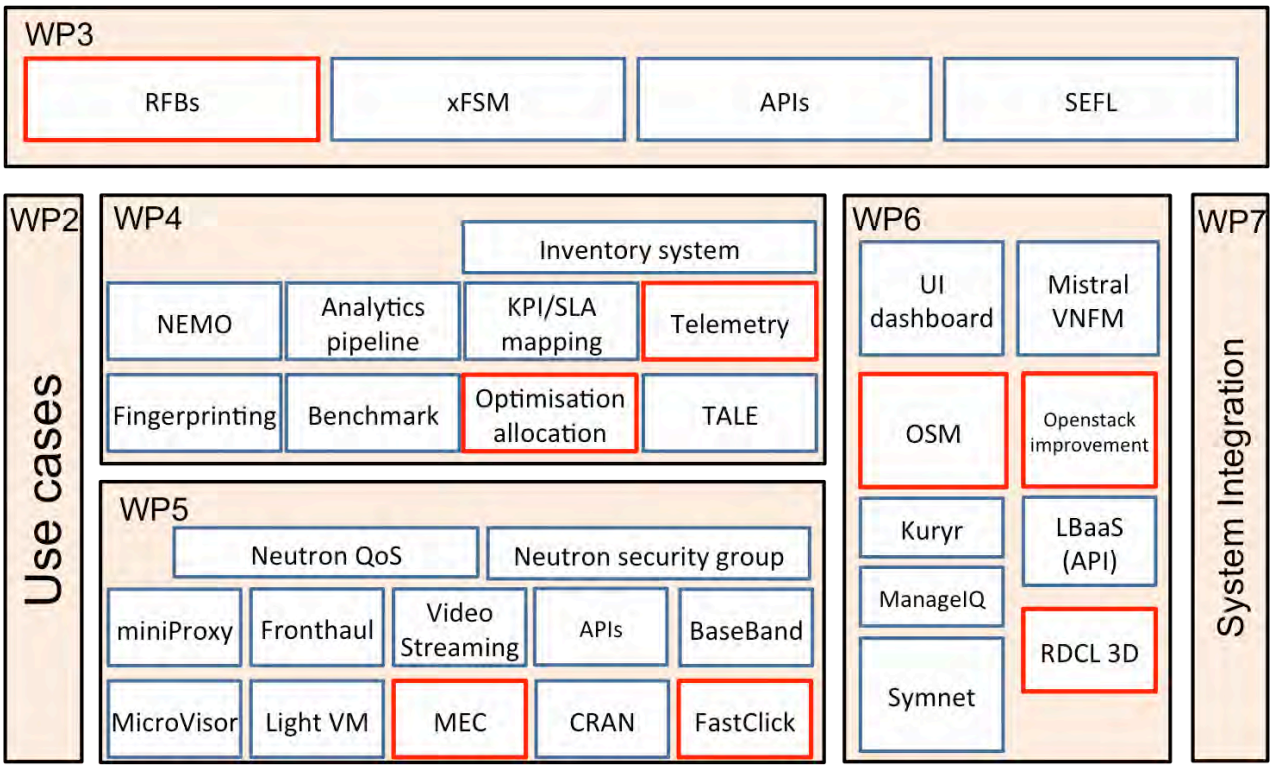


Figure 21: Graphical summary of Scene 4.b technologies

Starting from the existing MEC offloading solution, a new TOF, called *TOF-fastclick* can be added in parallel to the existing TOF. TOF-fastclick is instantiated on a hardware (or virtual) resource identical to the one of the existing TOF (to allow simple comparison of load). A load balancer can be added in front of the two TOFs. The load balancer is initially configured to send 100% of the traffic toward the standard TOF. Then a portion of the traffic (*e.g.* 50%) is diverted toward TOF-fastclick. The monitoring infrastructure allows analysis of the performance of the TOF-fastclick implementation, compared to the existing one.

3.12.4 Implementation challenges for the demo

In addition to the implementation challenges already mentioned in scene 3.b, this scene will require:

- A detailed description of the TOF functionality (both data plane and control plane) is needed.
- The description of the TOF should be translated in RDCL (*e.g.* using ETSI modelling).
- The TOF functionality must be re-implemented in FastClick (which will require not only to produce the configuration, but also to develop a few new Click elements for MEC-specific functions).

3.12.5 Evaluation plan and validation metrics

The validation metrics will be the same as for scene 3.b (*i.e.* instantiation time, bandwidth savings, latency reduction, service continuity and cache storage reduction). In this scene, we will focus on a comparison between the two (or three, if OPP is also used) TOF implementations.
In addition, we will evaluate development effort and ease of configuration for the FastClick-based TOF, and its use of physical resources.

3.12.6 Relevant 5G-PPP metrics

Relevant 5G-PPP metrics are the same as for scene 3.b.

3.13 Scene 4.c: Advanced network control and management

After the deployment of CRAN and MEC in Scene 1.b, the client can access a series of services on demand. However, the control of this communication is currently restricted to a set of parameters or even statically defined.

In this scene, Superfluidity leverages SDN to provide advanced management and control over the network resources, more specifically: (1) client access control, (2) extended network operator management, and (3) network reliability.

Notice although we will be working at the edge, these principles could be extended to the central cloud as well.

3.13.1 Scene description

In a logical and concise manner, Superfluidity's network resembles the network shown in Figure 22. The network is divided into an edge and a central cloud. Previous scenes have already shown how services may be deployed on any host node of the network and how they may be moved between the central and the edge clouds, and vice-versa.

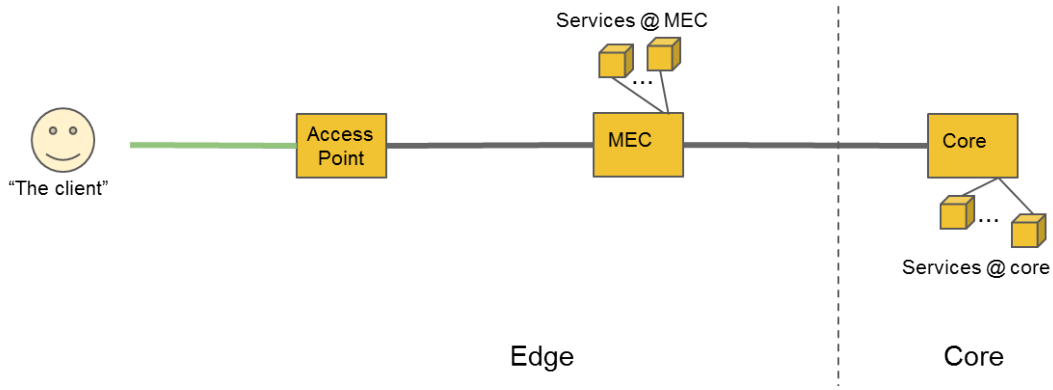


Figure 22: Simplified schema of Superfluidity's network deployment

To access host nodes, traffic usually traverses a hypervisor that chooses where to divert the traffic (specifying the VM or container). NFV orchestration uses this hypervisor to guarantee scalability and balance the load across the network. However, this has some limitations; e.g. filtering can only be applied based on L3 or L4 parameters.

As hypervisors are usually implemented as Open vSwitch (OvS) software switches and these switches are OpenFlow-capable, we could leverage them to extend their functionality with an SDN controller. For example, they could balance the load between nodes based on more sophisticated parameters (L2, L3, L4 or any combination of them), rerouting traffic transparently to the network manager, as depicted in Figure 23.

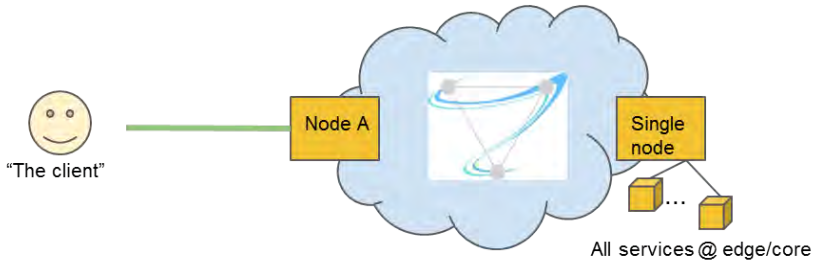


Figure 23: Unified vision of the network after applying Superfluidity's principles

The principles of scene 4.c are the following:

1. Client access control

   The client might have access to some services in the network or not. For example, he might have hired a "premium" service to access services at the edge, for extra throughput and minimum latency. This scene aims to provide a framework for this control. If a client has no service registered, he will be prompted to a portal in which he/she can subscribe to some services package and, after that, access the different services.

2. Extended network operator management

   In previous scenes, paths between the client and services were built automatically. However, this might be wrong for many reasons (bugs, security reasons, billing, etc.). In this way, scene 4.c focuses on providing a graphical interface such that the network manager can modify the network behaviour at runtime.

3. Network reliability

   The SDN component might fail for different reasons. Scene 4.c will provide reliability by deploying a cluster of SDN controllers, so in case one fails, another one can take the token.

4. Coordination of SDN and NFV

   Currently, the deployment of SDN and NFV components is somehow scattered. Coordination of both technologies could open the range of possibilities for configuration.

For scene 4.c, we aim to deliver the first three principles and, optionally, the extended functionality described in the last (fourth) principle.

3.13.2 Technical trigger to start this scene

When considering SDN, often the "human" part of the network (users, network managers) is not considered; an example of this is shown by the immaturity of the Northbound Interface [10]. In fact, most SDN research explores the three planes of the architecture defined by the ONF: Application, Data and Control, while the Management entity is still approached by very specific use cases (specific tools for profiling, network debugging or security issues), thus not generalised (these tools are usually developed for specific frameworks) and with a lot of standardisation effort still to be done. Particularly, network administrators with a low knowledge of programming, might need to consider going back to studying, as they will likely deal with more programming than ever, while ideally the automatisation of tasks with SDN should drive in the opposite direction, thus causing that network administrators require less knowledge in computer science and more in law or management, for instance [11].

Additionally, combinations of SDN and NFV technologies for an optimised network management and control is a hot topic under research. However, communication between both components still depends on specific deployments. The purpose of this scene is to provide a framework based on registration of services and dispatch actions that are generalised enough to cover different use cases.

3.13.3 Technical description and involvement of Superfluidity technology

SDN simplifies management tasks, allowing fine-grained control over the network. This scene is particularly focused on the "human" part of the network, as the clients will be able to subscribe to the services they want, and network managers will be able to add manual refinements to the network without losing the automatic control provided by the SDN and NFV frameworks.

Figure 24 refers to the map of technologies used in this scene.

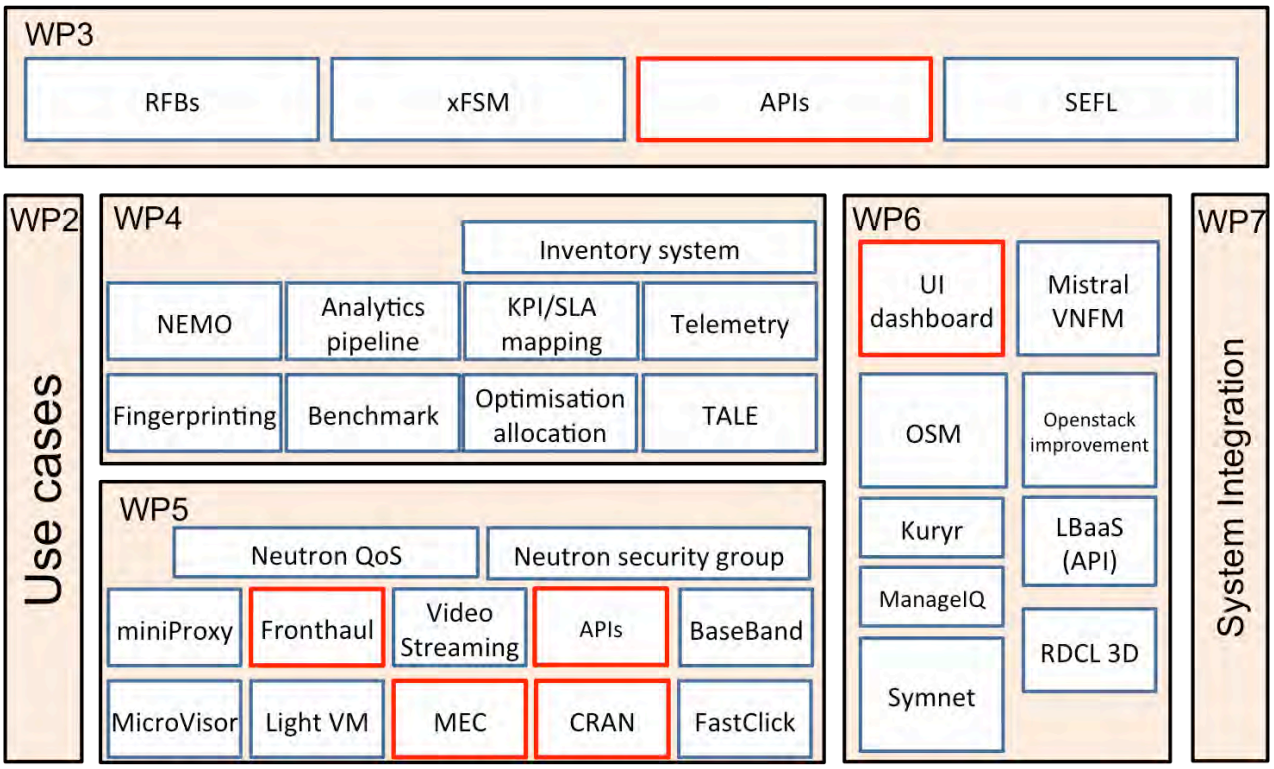


Figure 24: Graphical summary of Scene 4.c technologies

3.13.4 Implementation challenges for the demo

Scene 4.c requires an extension of the ONOS GUI and REST APIs. The extension of the GUI will provide the network manager the capabilities for interaction with the platform network resources. Additionally, an ONOS network application should be developed to provide the extended functionality described before.

3.13.5 Evaluation plan and validation metrics

The evaluation of this scene will be based on the time saved in managing the network. We plan to measure the time for user access, network operator management and recovery after SDN nodes failure.

3.13.6 Relevant 5G-PPP metrics

This scene is related with the scalable management framework that enables fast deployment of novel applications, and particularly with the reduction of the network management OPEX aimed to be at least 20% compared to today's networks.

4 Summary

In this section a summary of the scenes is provided, in order to have an overview of the use cases and the relevant Superfluidity technologies they utilise and demonstrate. In particular, a good coverage of the various technologies developed in Superfluidity is demonstrated through the overview diagram.

4.1 Map with all SF technologies

The Figure below summarises in a single figure all the Superfluidity technologies addressed by the selected use cases. It can be seen that an almost complete coverage is obtained, allowing a large number of the project developments to be evaluated, in a single and integrated manner via the common demonstrator.

Besides architectural and conceptual work developed in WP3, WP4, 5 an 6 concrete developments will be part of the integrated demonstrator, contributing actively for the execution of, at least, one use case. Since the use cases will run in a sequential way, integration is required and, thus, will be achieved.

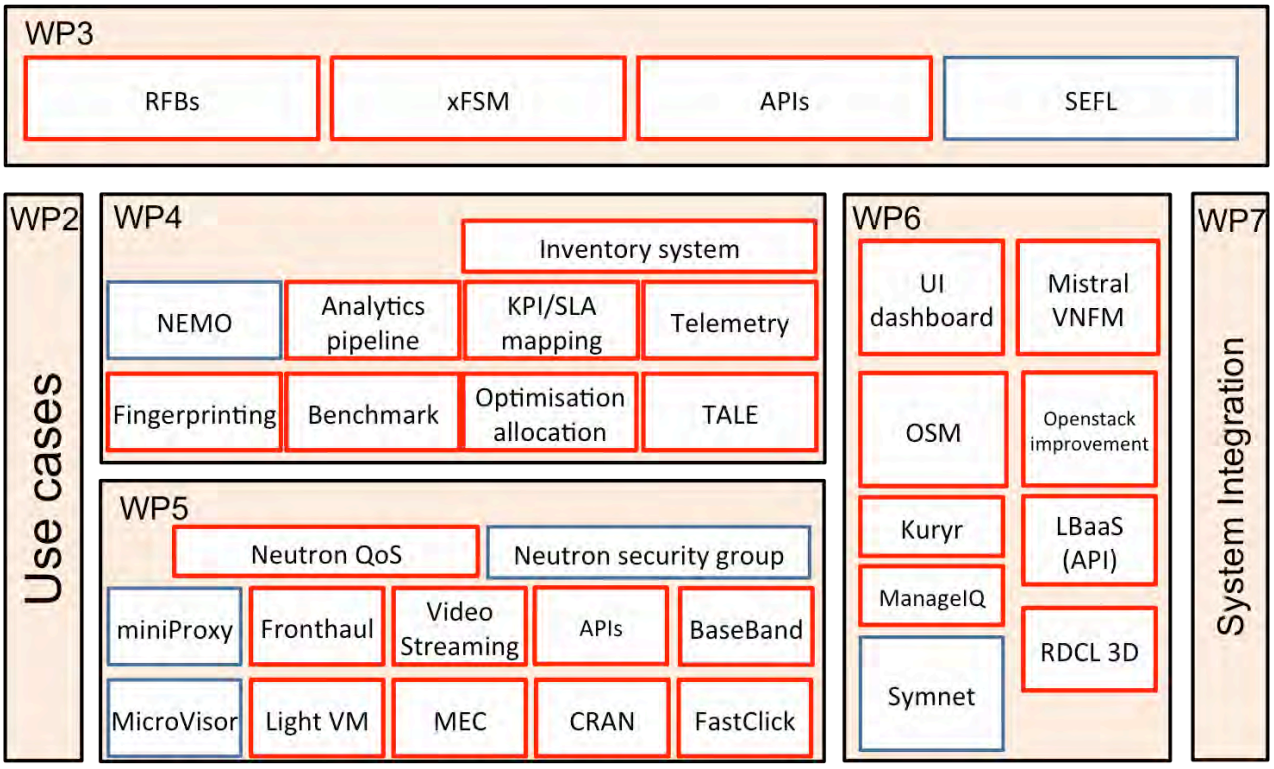


Figure 25: Summary of addressed Superfluidity technologies

4.2 Summary of implementation challenges

The following table brings together in a single place all the implementation challenges identified above. The project provides solutions to these challenges, which will be described in more detail in D7.2, and evaluated later in D7.3.

Table 2 – Summary of implementation challenges

Scene	Summary	Implementation challenges
0	Infrastructure set-up	Setup of a heterogeneous environment, spanning across multiple clouds, with support for multiple virtualisation technologies.
1.a	Superfluidity system design	Adopt and extend descriptors, suitable to the project component types; develop a graphical tool to organise and edit those descriptors; decouple design from deployment, providing required translators and interactions.
1.b	Initial components deployment (CRAN+EPC and MEC)	Decoupling RFB from nodes for dynamic deployments Provide a high level orchestrator, enabling multi-level orchestration for CRAN and MEC. Establish a datapath chain of dynamically deployed RFB.
2.a	Workloads offline characterisation	Characterisation of workloads using different virtualisation approaches; Telemetry framework supporting VM, containers and Unikernels; Scaling model implementation.
2.b	Workload (video streaming) / service deployment	Blocks based decomposition of the streaming server, allowing for distributed deployment and independent scaling.
3.a	Central cloud services automatic scaling	Fine-tune a suitable telemetry system; Selection of data to be monitored; Orchestration for a fine control loop.
3.b	Services relocation to edge cloud	Integration between service execution environment and service management and orchestration; Fast instantiation of services

3.c	Container based services deployment at the edge cloud	Fast transfer of container images from central to core clouds and fast instantiation
3.d	Unikernel based services deployment at the edge cloud	Coexistence at MEC of traditional and unikernel VMs; Implementation of DDoS related actions inside unikernel VMs
3.e	Services' optimisation at the edge cloud	Networking integration solutions between different virtualisation technologies. Implementation of additional traffic handling rules' types.
4.a	Alternative load balancing function at the central cloud	Use a commercially available load balancer, integrated with OpenStack.
4.b	Alternative edge offloading function	ToF description translation into RDCL; Efficient TOF implementation in FastClick.
4.c	Advanced network control and management	Achieve ONOS GUI and REST APIs extensions for integration with network managers.

These implementation challenges, which reflect the use case objective, have implementation solutions proposed by project, which will be deployed and tested, over the common testbed, made of the two central and edge clouds. This covers from network and services design, to monitoring and automatic correction, going through fast services instantiation and relocation.

4.3 Summary of proposed validation metrics

The following table captures all the metrics identified as relevant for the evaluation that will be carried out as part of D7.3.

Table 3 – Summary of proposed validation metrics

Scene	Summary	Identified metrics
0	Infrastructure set-up	Not applicable
1.a	Superfluidity system design	Service design time

		OpEx reduction
1.b	CRAN+EPC and MEC components deployment	Instantiation time Orchestrator efficiency Cost and Optimisation Latency and Throughput
2.a	Workload offline characterisation	No directly related metrics
2.b	Streaming service deployment	Throughput Latency
3.a	Central cloud services automatic scaling	Service provisioning time Energy saving User experienced data rate Reaction time 10 to 100 times more connected devices 10 times to 100 times higher typical user data rate 10 times lower energy consumption
3.b	Services relocation to edge cloud	Instantiation time Bandwidth savings Latency reduction Service continuity and deployment time Cache storage reduction
3.c	Container based services deployment at the edge cloud	Instantiation time Latency Service density User data rate Mobile data volume
3.d	Unikernel based services deployment at the edge cloud	Reaction time Instantiation time Traffic loss Reliability
3.e	Services' optimisation	Service Creation Time

	at the edge cloud	Network Latency Network Throughput Energy Consumption
4.a	Alternative load balancing function at the central cloud	See 3.a
4.b	Alternative edge offloading function	See 3.b
4.c	Advanced network control and management	Time savings

These evaluation metrics although highlighted as relevant for a particular scene may also be appropriate for other scenes and, as such, during the full run-through the metrics identified above, will likely not be an exhaustive list and will be further extended, in Tasks 7.2 and 7.3 scopes.

5 Conclusions

This document identifies and describes the selected Superfluidity use cases that once combined, contribute to the project integrated demo. The use cases have been organised as a series of scenes, each of which includes one or more pieces of Superfluidity technology. Overall, the obtained integrated demonstrator is intended to validate the project's developments by integrating, evaluating and showcasing the project work. The reason for having a single, integrated demo is to address the real engineering challenged needed to provide a working platform that represents a large set of stakeholders (the consortium partners) different objectives and requirements. A number of challenges to be addressed by each use case have been identified. Additionally, applicable validation metrics and an evaluation plan are already proposed to help guide the evaluation efforts. This will be reviewed and further extended in the scope of Task 7.2 (integration validation) and Task 7.3 (assessment).

In summary, for each use case/scene the document provides:

- A brief description of the scene
- The technical trigger that starts the scene
- A technical description explaining the involvement of Superfluidity technology
- Implementation challenges foreseen for the demo
- Ideas about evaluation and suitable validation metrics
- The potential impact on 5G-PPP metrics

In the first year review, four of the use cases were demonstrated as individual demonstrations. Annex A, included in this document, briefly outlines our four initial 'stand-alone' demos, which were prepared and run for the 1st project technical audit as they have not been described in any previous project Deliverable.

The content of this Deliverable serves as a starting point for the project integrated demonstrator and for the assessment that will be done.

6 References

[1] What is an Application Delivery Controller (ADC)?, https://www.citrix.com/products/netscaler-adc/resources/what-is-an-adc.html

[2] Citrix NetScaler ADC, https://www.citrix.com/products/netscaler-adc/

[3] Citrix NetScaler MAS, https://www.citrix.com/products/netscaler-management-and-analytics-system/

[4] ByteMobile Adaptive Traffic Management, https://www.citrix.com/products/bytemobile-adaptive-traffic-management/tech-info.html

[5] Cisco Visual Networking Index: Forecast and Methodology, 2016–2021 http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html, Updated on June 2017.

[6] Apple. (2016, 3). *HTTP Live Streaming*. Retrieved from HTTP Live Streaming: https://developer.apple.com/streaming/

[7] ISO/IEC JCT1/SC29 MPEG. (2014). *ISO/IEC 23009-1:2014: Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats.* ISO.

[8] Ioannis Prevezanos, Andreas Angelou, Christos Tselios, Alexandros Stergiakis, Vassilis Tsogkas and George Tsolis: "Hammer: A Real-world, end-to-end Network Traffic Simulator", IEEE CAMAD 2017, p 1-6, Lund, Sweden, 19-21 June 2017

[9] Rufael Mekuria, Michael McGrath, Christos Tselios, Dirk Griffioen, George Tsolis, Shahar Beiser: "KPI Mapping for Virtual Infrastructure Scaling for a Realistic Video Streaming Service Deployment", 8th International Conference on Quality of Multimedia Experience, Lisbon, Portugal, June 6-8, 2016

[10] Felipe A. Lopes, Marcelo Santos, Robson Fidalgo, Stenio Fernandes. "A Software Engineering Perspective on SDN Programmability". IEEE Communications Surveys & Tutorials (Volume: 18, Issue: 2, Second quarter 2016)

[11] Daniel M. Batista, Gordon Blair, Fabio Kon, Raouf Boutaba, David Hutchison, Raj Jain, Ramachandran Ramjee, Christian Esteve Rothenberg "Perspectives on software-defined networks: interviews with five leading scientists from the networking community". Journal of Internet Services and Applications, Oct. 2015, vol. 6, n. 1.

Annex A: Initial demos

A.1 Demo 1: Demand-driven orchestration for 5G deployments

a. Description

This demonstration highlights the activities around telemetry, monitoring and orchestration. It integrates into a single unified demo a set of components that a good proportion of partners have been working on in Superfluidity.

We use some of the core OpenStack projects as the underlying Virtual Infrastructure Manager (VIM) along with some modifications that enable the use case. The physical infrastructure has been set up as per the diagram shown in Figure 1, in the BT premises at Ipswich, UK. Access to the partners has been provided via VPN access. The servers S1 through S5 were then set up with Operating Systems; namely CentOS 7 for S1 through S4 and then a bare metal deployment of Citrix Hammer was deployed on S5. Two of the nodes (S1 and S2) are configured as OpenStack infrastructure nodes (e.g. Neutron, Horizon, Cinder etc). Two other nodes are configured as Compute Nodes (S3 and S4), with the final server dedicated as a Hammer node.

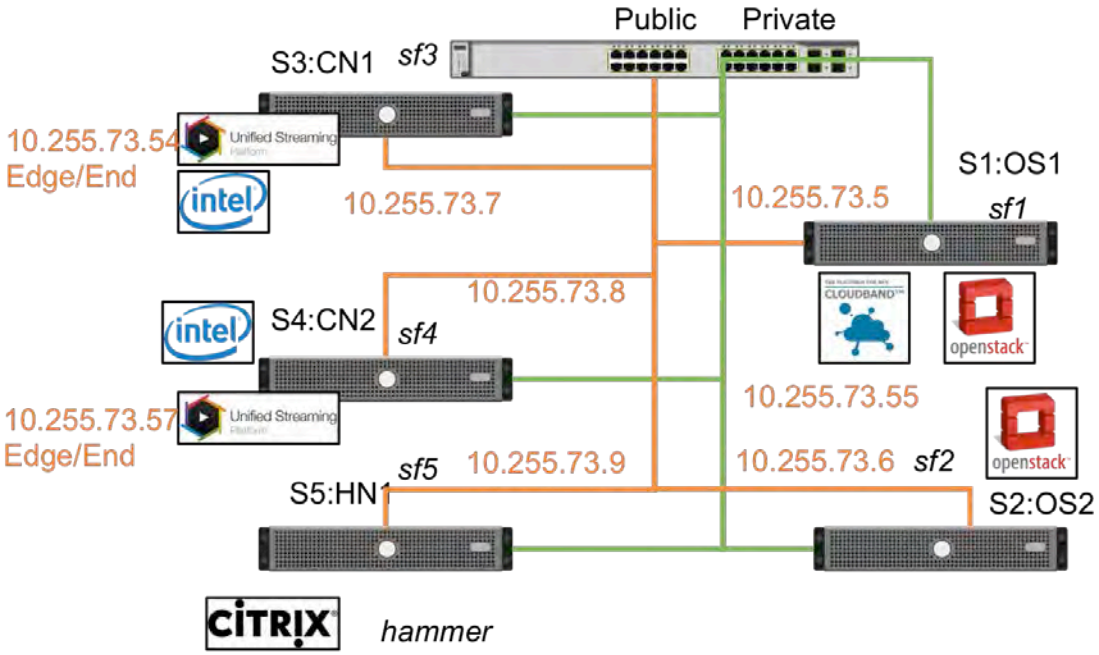


Figure 26: Superfluidity test-site configuration for this Demo

The general idea with this use case and associated demonstration is to show that for 5G networks we will need to leverage Cloud technology and also overcome some of the limitations that are present in current Cloud deployments. Currently Cloud deployments typically resemble that of

Figure 8, where an administrator sets up an infrastructure and then creates virtual machines or enables a mechanism for end-users to set up virtual machines on the underlying platform. Generally this system is then left in an uncontrolled and weakly monitored state. If there are alerts or problems, these are raised, submitted etc. and eventually actioned upon.

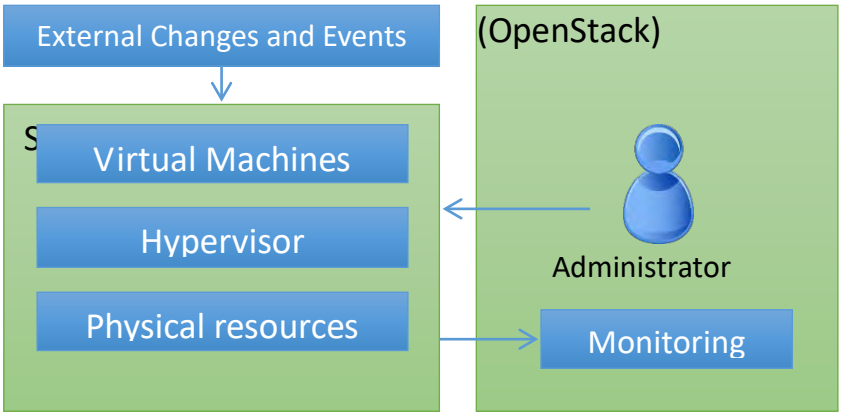


Figure 27: Before: Administrators used to manage workloads

In Superfluidity we argue that the control loop and decision logic should be performed by the platform itself, given sufficient understanding of what is going on in the platform and adequate descriptive systems, as shown in Figure 3. This then allows the system to react to most conditions in a pre-planned way that takes away the administrator from the critical path on performing steps to resolve certain conditions. This does not completely remove the administrator but allows for better day-to-day functioning of the system and also for Service Level Agreements (SLAs) and Key Performance Indicators (KPIs) to be used to provide sufficient levels of service per contracted agreements.

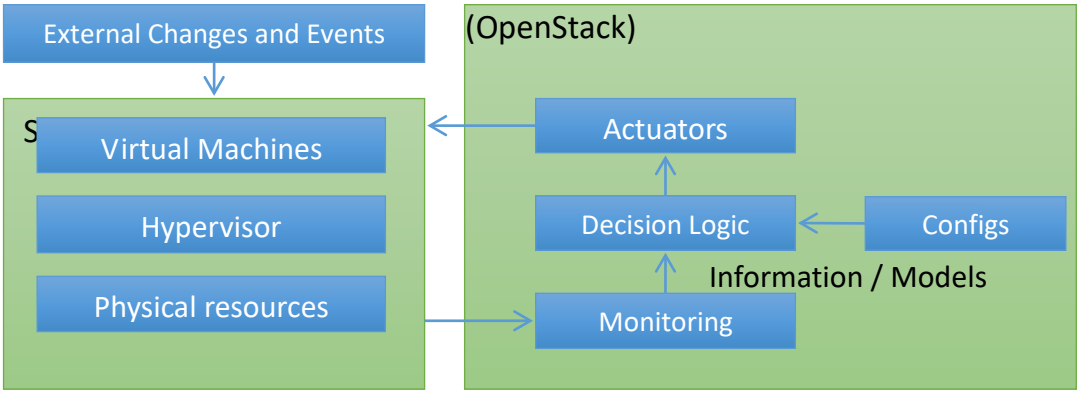


Figure 28: After: 5G deployments, need (semi-)automated orchestration frameworks

For the demonstration that was shown at the EC project review offline analytics were used to determine that latency and throughput were key parameters for the use case chosen which was that of playing video content via Unified Origin Streaming product. A high level of throughput would be detected by Ceilometer in two different scenarios. A) The case that noisy neighbours are present and create traffic between each other. B) Overall throughput is too high and we need to scale-out.

Once this was detected and passed to Mistral, it could then be reacted to via HEAT templates that changed parameters in the platform, in two possible ways:

a) Adding QoS rate limiting and changing the load balancer to another existing node;
b) Spawning a new VM, adding that new instance to the load-balancer pool and then changing the weighting to focus on the new instance.

b. Objectives

With this demonstration we intend to meet the demand for low-latency, high-bandwidth, always-on connections

- Currently have long provision times and waste resources (at the data centre)
- Video-streaming services are pre-provisioned (to try and decrease end-to-end latency)

The number of devices in 5G landscape is growing exponentially

- We expect 1000x volume of traffic of 3G/4G

Superfluidity wants to ensure that KPIs / SLAs are monitored and enforced.

The orchestration framework needs to meet the demands of the use cases.

The Superfluidity platform can create resources on-demand to react to telemetry allowing for automated, transparent service delivery mechanisms.

We undertake a model based approach for determining resource placement that can decide the best way to utilise the resources.

In parallel to the online monitoring we perform offline low-level analysis of KPI metrics for optimisations.

Superfluidity is designed for scale-independence, allowing for the addition of resources anywhere and is architected to work on heterogeneous hardware platforms.

- Offline service profiling using analytics technologies
- Real-Time Telemetry to monitor a service
- Application driven on-the-go resource scaling
- Cost reduction and quality improvement for video streaming
- Noisy Neighbour interference mitigation by QoS Policing
- Useful for Video Streaming and other bandwidth heavy applications
- Starting point for the Superfluidity architecture that features on-the-go scaling of compute and network resources

c. Demo Steps

Scenario A)

1) Video Streaming Starts (demonstrated using a custom JavaScript player)
2) Traffic generator starts to ramp up
3) A critical load at the server is reached and the user experience could start degrading soon
4) Ceilometer telemetry picks this up and raises an alarm to trigger a scaling based on pre-trained scaling model
5) Scaling is performed by Heat and Mistral
6) Load balancer updated to balance video traffic

Scenario B)

1) Video Streaming scenario with noisy neighbour and no rate policing
2) Video Streaming scenario with noisy neighbour and rate policing (QoS)

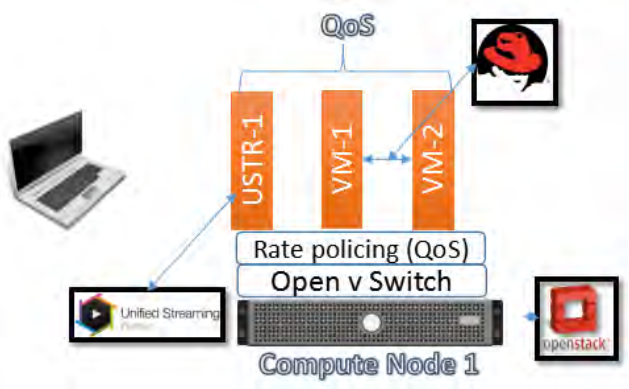


Figure 29: Demonstrating the noisy neighbour scenario

We intend to improve the decision logic feedback part of the demonstration, such that it is based less on human generated input, but rather through templates and enforced behaviours. This will require tooling to allow administrators to more quickly set up sets of actions to undertake.

We also intend to further evaluate the platform and understand how to make different parts of the system react at different rates, suitable for the heterogeneous workloads that are expected.

We also intend to fit the orchestration element as an overarching part of the fully integrated demonstration where the platform can react and meet the demands of users in real-time.

A.2 Demo 2: Software defined wireless network

a. Description

This demonstration aims to build a modular hybrid fixed and wireless front-haul system that enables high level of programmability. It integrates front-haul in a fully automated cluster based provisioning system for RFB's deployment. It mainly provides instantaneous deployments and instantiation of RFBs to have a minimum downtime.

b. Objectives

It also demonstrates how to adapt resource usage to the users' needs in near real-time. it builds a full software wireless network with a common REST based interface for simpler and better programmability. We demonstrate the flexibility of RFBs (RFBs can be redeployed independently over time).

c. Demo Steps

We demonstrate the deployment and operation of an end-to-end network in a one click – with a joint NFV and SDN control using RFB principles:

1) SDN controller instantiation
2) EPC based RFBs instantiation
3) Front-haul registration
4) Front-haul first flow (red flow) instantiation
5) BBU instantiation & registration
6) UE connection
7) Front-haul first flow (red flow) deletion
8) Front-haul second flow (green flow) instantiation
9) UE connection

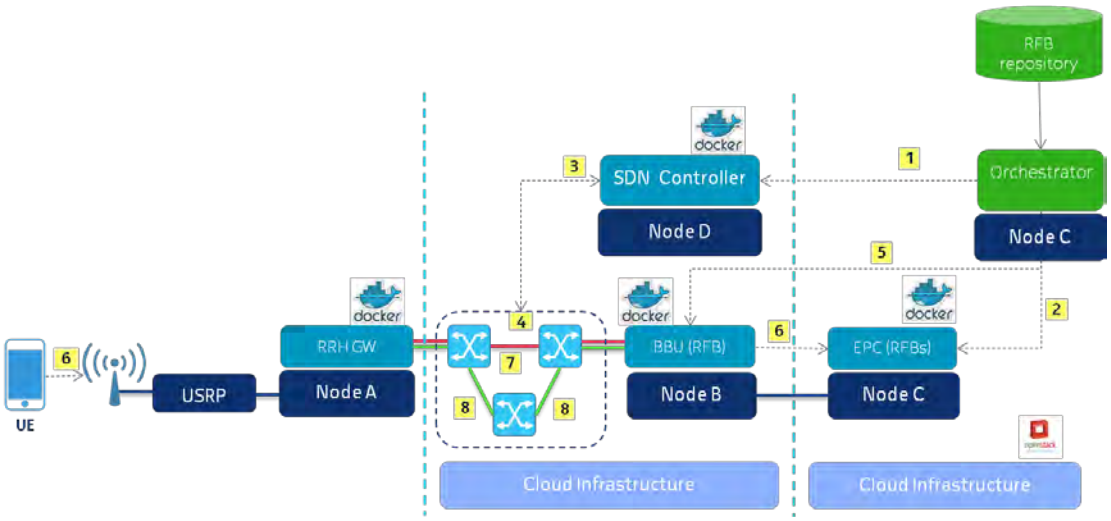


Figure 30: Demo: Software defined wireless network

A.3 Demo 3: Network configuration verification

a. Description

Static verification of network behaviour is already proving its value in a wide range of scenarios [Stoenescu-Sigcomm16, Kazemian-NSDI12, etc]. Most of the current approaches exhibit a fundamental shortcoming because verification takes place after deployment. In particular, whenever the network control plane changes a new dataplane is deployed and the verification process is started: first, a model of the network dataplane is built, and the verification process takes place on this model; finally, the results are yielded back to the network operator that decides if any corrections have to be applied or not. Such an operation mode is suboptimal for two reasons:

- Potential bugs can be deployed in production and create effects before the verification process captures them and they are repaired by the operator
- The model of the dataplane may be inaccurate invalidating the results of the analysis.

b. Objectives

Our contribution is to perform verification before deployment and to ensure that the deployed dataplane is equivalent to the model. A data plane update is deployed only if it was proven to respect the network policy in place. In order to enable such an approach the network operator has to be provided with a flexible policy specification language that the verification tool can then use as an input in order to verify the validity of the network. The high-level approach is detailed below: given RFB specifications written in SEFL and the desired interconnections, our tool will build a dataplane model that will be verified by Symnet for consistency with the operator's policy. If that is the case, an equivalent dataplane implementation in the P4 language will be automatically generated and deployed.

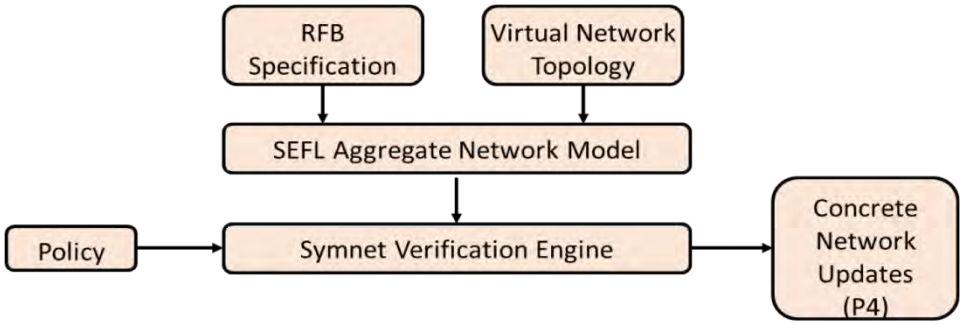


Figure 31: Network verification approach

c. Demo Steps

To demonstrate our work we will use a setup consisting of a P4-enabled data plane deployed within a mininet network. The input will be a SEFL description of RFBs and their connections in Symnet format (as shown in the first year demo too), together with a desired high-level policy specified in the CTL language.

a) The verification cycle begins with the policy specification phase. This is the only phase which requires human intervention. The requirements are specified using a dialect of CTL (Computational Tree Logic). The demo will explain the CTL policy and also show how changes can be made to specify different requirements.
b) Symnet will then be invoked to verify whether the policy holds; the answer will be either yes, or examples of packets/nodes where the policy does not hold.
c) If the policy holds, a parser from SEFL to P4 will be invoked and generates runnable P4 code automatically. The dataplane will be deployed in mininet, and the demo operator will show that the dataplane behaves as expected by injecting traffic at different ports.
d) The last part of the demo will examine the relationship between the SEFL model and the P4 code, highlighting interesting parts and showing how equivalence is achieved.

A.4 Demo 4: Video transmuxing and MEC

a. Description

This demonstration focuses on the usage of *Mobile Edge Computing* (MEC) for efficient video delivery, using *late transmuxing* (LTM) technologies at the edge. LTM corresponds to the action of encapsulating differently the raw video/audio content, according to the receivers' environment. Doing it at the edge reduces caching requirements, since only the raw content is stored (single format) and only a single stream is transferred in the backhaul, as only the raw content is required at the transmuxing point.

For demonstration purposes, Altice Lab's MEC implementation has been integrated in a virtual deployment of an EPC (OpenEPC) and used to run Unified Origin's LTM. The integration is performed at the S1-U interface, between the eNB and SGW elements. This way, by opening the GTP-U tunnels, the MEC realisation is independent from E-UTRAN and EPC functions. In a real environment, the MEC solution would be deployed physically at the same data centre PoP as the eNB.

b. Objectives

This demonstration shows the execution of services at the edge, using elements being implemented in line with the "early" MEC standardisation work. It adopts SDN and NFV related technologies (OVS, OpenFlow, Openstack, etc.) and is aligned with the NFV architecture. The way the MEC Host is implemented guarantees isolation among ME Apps (networking and compute), since they can belong to different 3rd parties (App providers).

In this context, caching gains are significant due to the reduction of different video formats transferred*. For that purpose a *Smart Transmux Edge Cache* with byte range caching implementation was executed.

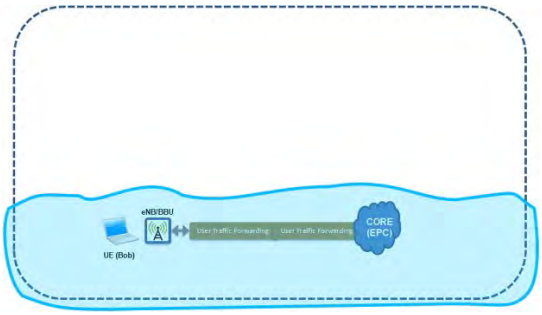
** For simplicity, this demo only uses a single user, not fully demonstrating the actual possible gains of this solution.*

c. Demo Steps

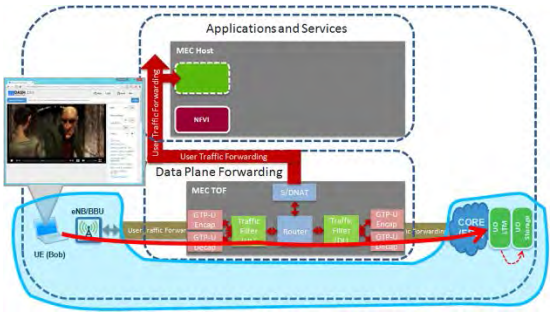
1. A user willing to watch a video, types the corresponding URL in the browser
2. The respective video, without any other action, starts being retrieved from a central server, since the typed FQDN/URL corresponds to an IP address located there (behind the SGi interface)

3. At some point in time during the video streaming, and triggered by some operator policy, it is decided to start providing the video from the closest edge to the user (at this stage, these policies are not implemented)
4. Acting on the MEC Host, the LTM application, which was previously on-boarded in the system, is instantiated (in the OpenStack environment)
5. After the LTM instantiation is fully operational, MEC traffic rules at the TOF are configured in order to redirect (offload) the video request to the new LTM at the edge
6. Due to the service protocols in use (DASH) and the way video streaming works, the user will, transparently, start obtaining the video service from the edge LTM, which in turn accesses the central storage for the raw video contents, in a connection outside the GTP-U tunnelling
7. Later, due to another trigger in the operator policies, it is decided to provide the video service back from the central cloud and the traffic flows again between the UE and the streaming server at the central cloud, without being noticed by the user
8. The LTM application instance in the edge is then disposed

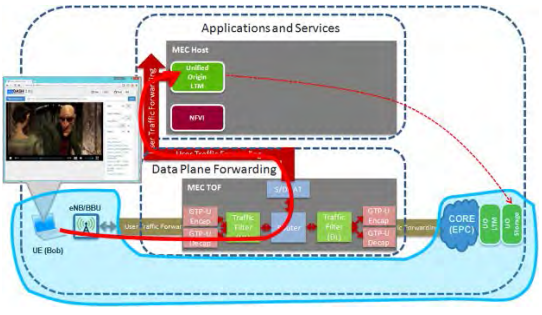
Some of the steps are shown in the following figures.



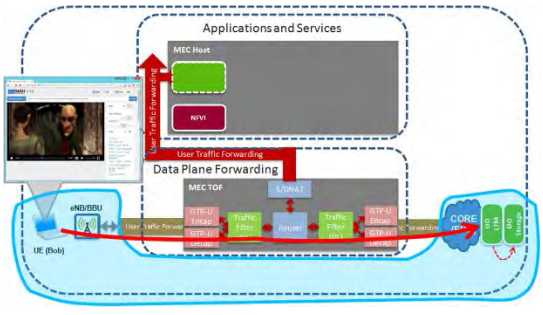
A. Network scenario, without MEC



B. Video being streamed from the central cloud, with MEC already inserted in the data path (step 2 above)



C. After LTM is instantiated and traffic rules are configured, video flows from the backend to the LTM and is streamed from there to the UE (step 6 above)



D. (step 8 above)

Figure 32: Demo of video streaming from MEC

This demonstration will be integrated with Demonstrations 1 (Orchestration) and 2 (CRAN) in the scope of the overall Superfluidity integrated demonstrator. In that scenario, CRAN and MEC functional blocks will be deployed using suitable orchestration decisions and operations, with ME Applications (e.g. the LTM) Lifecycle Management (LCM) also being performed by orchestration. Mobility support, associated to UEs' mobility, will also be experimented.