SUPERFLUIDITY

a super-fluid, cloud-native, converged edge system

Research and Innovation Action GA 671566

Deliverable D1.3:

Project Vision and Roadmap, v2

Deliverable Type:	Report
Dissemination Level:	Public
Contractual Date of Delivery to the EU:	31/03/2018
Actual Date of Delivery to the EU:	15/05/2018
Workpackage Contributing to the Deliverable:	WP1

Editor(s):	Stefano Salsano (CNIT)
Author(s):	Nicola Blefari Melazzi, Giuseppe Bianchi, Luca Chiaraviglio, Stefano Salsano (CNIT), Philip Eardley (BT), George Tsolis (Citrix), Walner Saint-Aubin (EBlink), Michael J. McGrath (Intel), Bessem Sayadi (NOKIA-FR), John Thomson (OnApp), Pedro A. Aranda (Telefónica, I+D)
Internal Reviewer(s)	George Tsolis (CITRIX)

Abstract: This deliverable reports the final version of the Project Vision and Roadmap of the project. The document takes into account results generated within the project, other relevant scientific, technological or market developments, and the long term strategies of the EU and project partners. Its aim is to concisely describe the project as a whole, and publicly.

Keyword List: SUPERFLUIDITY, Project Vision, Roadmap

Executive Summary

This deliverable reports the final version of the Project Vision and Roadmap, delivered at the end of the project. The document starts from the Description of Work document, and takes into account results generated within the project, other relevant scientific, technological or market developments, and the long term strategies of the EU and project partners. Thus, it includes parts of the original proposal, updated when needed, concise description of work performed, and new/updated concepts.

In some cases this document duplicates information provided in other deliverables, but its aim is to more concisely describe the project as a whole, and publicly. More details can be found in project deliverables (http://superfluidity.eu/results/deliverables/), which are quoted in this document by their official abbreviation, Dx.y for final deliverables and Ix.y for internal, intermediate deliverables. All deliverables are available for reviewers, while only those marked as “P(ublic)” are publicly available.

As for the organisation of the document, after a brief introduction, in Section 2 we report the vision of the project, detailing major features and expected benefits, potential market, and comparison with other solutions. In Section 3, we describe how we worked to reach our vision, by: i) listing specific objectives; ii) specifying scientific/technical, market and societal aspects of the overall vision; iii) discussing the relevant KPIs. In Section 4, we summarize the work performed so far, including both design and demonstration activities and related expected impact. Finally, in section 5 we provide the reference to the partners’ exploitation actions and plans.

INDEX

List of Figures

List of Tables

List of Abbreviations

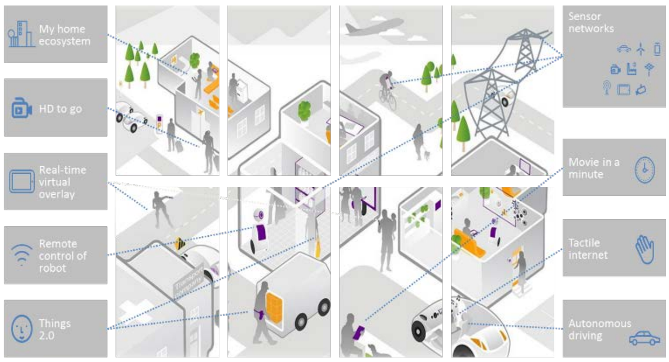
API	Application Program Interface
CDN	Content Distribution Network
DPI	Deep Packet Inspection
DRM	Digital Rights Management
ETSI ISG	European Telecommunications Standards Institute Industry Specification Group
FTTX	Fiber To The x
GGSN / P-GW	Gateway GPRS Support Node / Packet Gateway
IoT	Internet of Things
JBOA	Just a Bunch Of Accesses
KPI	Key Performance Indicator
LTE	Long Term Evolution
LTM	Late TransMuxing
M2M	Machine to Machine
MEC	Mobile Edge Cloud
MIMO	Multiple-Input Multiple-Output
NFV	Network Function Virtualization
NFVI	NFV Infrastructure
NIC	Network Interface Card
NS	Network Service
ONF	Open Networking Foundation
OSS	Operational Support Systems
OTT	Over-the-top
PNF	Physical Network Function
PoC	Proof of Concept
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network

RFB	Reusable Functional Block, for a definition see section 3.5.1 and I3.1
RRH	Remote Radio Head
RRM	Radio Resource Management
S/Gi	Reference point defined by 3GPP between the mobile packet core and PDN (Gi is between GGSN and PDN; SGi is between P-GW and PDN)
SDN	Software Defined Networking
SDO	Standard Defining Organisation
SFC	Service Function Chaining
SLA	Service Level Agreement
SON	Self-Organising Network
UBB	Ultra-Broadband
UE	User Equipment
vCS	virtual Convergent Services
vHGW	virtual Home GateWay
VM	Virtual Machine
VNF	Virtual Network Function
VNFC	Virtual Network Function Component
VNFM	Virtual Network Function Manager

1 Introduction

The vision of the SUPERFLUIDITY project emerges from powerful drivers that are shaping our society.

The first driver is the increase of population and the still growing globalisation and physical and virtual mobility: more people (2 billion and a half in 1950, almost 7.5 billion today, half of them living in cities), and more interconnections among them.



Source: Nokia

The second driver is the proliferation of new or improved applications and services that need network connectivity: social networks, video (high definition), IoT (metering, smart home, connected cars), industry 4.0 (or the fourth industrial revolution, the current trend of automation and data exchange in manufacturing technologies), low latency services (games, virtual reality, autonomous vehicles), advanced services (face recognition and speech translation, cognitive and expert systems, big data exploitation). The white paper on “View on 5G Architecture” published on July, 13, 2016 by the 5G PPP Architecture Working Group (https://5g-ppp.eu/white-papers/) identifies three groups of use cases: i) Massive broadband (xMBB) that delivers gigabytes of bandwidth on demand; ii) Massive machine-type communication (mMTC) that connects billions of sensors and machines; iii) Critical machine-type communication (uMTC) that allows immediate feedback with high reliability and enables for example remote control over robots and autonomous driving. The “5G Manifesto for timely deployment of 5G in Europe”, presented by “key players in the telecoms industry and sectors which will use 5G technologies and services in the future” on July, 7, 2016 (http://ec.europa.eu/newsroom/dae/document.cfm?action=display&doc_id=16579), adopts another classification, grouping use cases according to so-called verticals: i) connected automotive scenarios; ii) connected eHealth scenarios; iii) connected planes, railway and high-speed transportation; public safety; iv) smart grids; v) smart City; vi) media and entertainment.

Whatever the classification, the network connectivity requirements of such applications and services are more and more demanding in terms of: i) data rates; ii) latency; iii) ubiquitous coverage.

The Digital Agenda Europe requires that all European citizens should access the Internet at bit rates greater than 30 Mbit/s by 2020, but in reality Ultra-Broadband (UBB) access performance is even higher and growing (FTTx, VDSL2-vectoring, G.fast, 4G, 4G+, 5G). The target latency is in the order of very few milliseconds. Network access is required everywhere.






Simply put, we have more connected devices, with each requiring higher data rates, lower latency, and ubiquitous coverage, with very high densities of users possible.

In addition, the importance of network connectivity and networked applications in our society and economy has the consequence of requiring significant improvements also in terms of: i) faster deployment of applications and services, so reducing their time to market and easing their evolution; ii) lower energy consumption; iii) enhanced security and privacy; iv) better reliability and dependability.

Furthermore, processing needs will be exacerbated in high capacity, dense networks. Current cloud computing solutions are not suitable for dynamic, real-time, high-bandwidth, low-latency applications because of issues such as granularity, localisation and configurability; service processing nodes should be distributed and located close to users or to routers or in local data-centres and not only in traditional data-centres.

As a matter of fact, the upcoming 5G network has two important characteristics, sometimes overlooked: the increasing role of cloud computing, and the fact that this generation of networks is concerned with the whole network and not only with its cellular part. The first generation of cellular networks was about portability; 2G marked the advent of digital technologies; 3G introduced widespread data services, in addition to voice services; 4G focused on Internet integration. 5G will be the whole network, including fixed sections and core, and will continue to disrupt the original Internet architecture, fully adopting cloud computing in its paradigm. The Internet will become a network of data centres, providing more often than not a 1-hop access to cloud, with a thin access section targeting infinite bandwidth/zero latency performance. Smartphones will increasingly provide access to artificial intelligence/cognitive services, with a computer-to-cloud-to-computer communication model, rather than person-to-person.

However, in spite of, or maybe because of, the great achievements that we witnessed in ICT, revenue growth for telecom operators is expected to halve from now to 2020. This means that demand cannot be satisfied by simply increasing network capacity, especially in networks that are becoming always more diverse, dense, mobile and changing unpredictably.

The answer to the challenging and sometimes contradicting necessities summarised above, consists in reducing capital and operating costs, by using low cost technologies, reducing energy consumption, sharing and optimising resources utilisation by dynamically allocating them in time and space, and in general resort to virtualisation techniques as much as possible. Benefits of a full virtualisation of network devices, at all layers, include [JAI13]: i) sharing: resources divided into multiple virtual pieces used by different users; ii) isolation: sharing of a resource does not endanger security and privacy of users; iii) aggregation: if resources are not big enough to accomplish a task, they can be aggregated; iv) dynamics: reallocation of resources in space and time on demand; v) ease of management: software-based devices are easier to manage and update.

In addition, it is necessary that the network be programmable, as a function of the needs of the services that it provides. An example of the capabilities of a virtualised and programmable network is the concept of a network slice; a virtual, end-to-end network, deployed in software, which runs in parallel to other slices on a common hardware infrastructure. A network slice also allows the isolation and support of different classes of services/customers.

The overall vision is thus the one of a software network with an application/service-centric network control able to dynamically share and allocate virtualised resources, allowing to: i) reduce costs, simplify network management, increase flexibility and ease evolution, and dynamically deploy network services.

5G will be, then, a fully "softwarised" network providing fixed and mobile UBB access to a distributed cloud infrastructure.

The SUPERFLUIDITY project contributes to the vision of a "superfluid" network, which will have the ability to instantiate services on-the-fly, run them anywhere in the network (core, aggregation, edge) and shift them transparently to different locations. Such capabilities are a key part of the converged cloud-based 5G future - they will enable innovative use cases in the mobile edge, empower new business models and allow almost instant roll-out of new services, and reduce investment and operational costs.

As for the organisation of this document, we first present the specific vision of the project, then how we intend to realise this vision and finally where we are now, and what is the work performed so far.

2 Vision Statement

The SUPERFLUIDITY project tackles crucial shortcomings in today's networks: long provisioning times, with wasteful over-provisioning used to meet variable demand; reliance on rigid and cost-ineffective hardware devices; daunting complexity emerging from three forms of heterogeneity: heterogeneous traffic and sources; heterogeneous services and needs; and heterogeneous access technologies, with multi-vendor network components.

The SUPERFLUIDITY solution is based on: a decomposition of network components and services into elementary and reusable primitives; a native, converged cloud-based architecture; the virtualisation of radio and network processing tasks; platform-independent abstractions, which permit the reuse of network functions across heterogeneous hardware platforms, while catering to the vendors' need for closed platforms/implementations; and high performance software optimisations along with leveraging of hardware accelerators.

As a result, the 5G network will benefit from: i) location-independence: network services deployable in heterogeneous networks; ii) time-independence: near instantaneous deployment and migration of services; iii) scale-independence: transparent service scalability; and iv) hardware-independence: development and deployment of services with high performance irrespective of the underlying hardware.

2.1 Major Features and Key Benefits

The ever increasing traffic volume is expected to push the limits of 5G access infrastructures, with some experts expecting up to 1,000 times the volume of traffic than what current 3G/4G deployments carry. This increased volume is to be accompanied with stringent low-delay requirements, all the while trying to reduce investment and operational costs in an increasingly competitive network operator market. Technological enhancements in the wireless domain are of course pivotal to facing such performance and scalability challenges, and mandate for the ability to fully harness capillary distributed antennas (for massive MIMO, beamforming, interference cancellation) via cooperative multi-point processing and control primitives. The ability to virtualise signal processing tasks, and to execute them inside centralised computational environments directly deployed within the network infrastructure itself (i.e., by trading network capacity with properly localised CPU resources, placed not only in the edge cloud but also remotely placed near the antenna) will enable such technologies to profit from the rate of innovation of typical software-based solutions. Furthermore, well-designed basic processing primitives and application programming interfaces would allow deployment of advanced algorithms which are today blocked by the proprietary nature of the involved components.

5G networks will face issues beyond performance and scalability. SUPERFLUIDITY plans to offer a converged solution to counter the complexity emerging from three forms of challenging heterogeneity:

1. Heterogeneous data traffic and end-points make proper planning and prediction of loads incredibly hard. The ability to effectively cope with, and dynamically adapt to, different, suddenly emerging, and ever mutating conditions requires a substantial leap in the level of flexibility and agility with respect to today's 3/4G networks; we argue that a much more fluid network architecture is called for.
2. Heterogeneity in services and processing needs: operators have largely recognised the need to transform the wireless access network from a bit pipe to a "smart" pipe. A network that is able to instantiate platform-agnostic software-based processing when needed, where needed, would open up seemingly endless possibilities. Paying businesses could run blazingly fast location-aware processing for their clients in the Radio Access Network. Virtual CDN operators could deploy virtualised content caches at edge networks, growing their infrastructure as their business grows. Customers could pay for on-demand ad removal in order to improve their browsing experience without draining their battery power, or could use aggressive traffic compression at the edge when the cellular load is high.
3. Heterogeneity in access technologies and their scale. 5G networks should become access-agnostic: specific wireless or wired technology should be treated as "just a bunch of accesses" (JBOAs), and seamlessly exploited (switched) so as to offer an "always best served" model down to a per-application level of granularity (rather than users). This process should be

autonomously driven by service level agreements and congestion conditions and should be scale-agnostic, i.e., handled in the very same way irrespective of the size and scale of the network nodes handling traffic/application flows.

All of these challenges point to the need for a new convergence model (see Figure 1) well beyond the “everything-on-IP” convergence that paved the way in past research efforts and that has already been implemented, to a large practical extent, in current 4G network deployments.

Concretely, we will accomplish this through a “decoupled” approach which identifies and abstracts re-usable network primitives, so as to hide complexity inside: i) their (vendor-specific) adaptation to the specific technologies and hardware, and inside ii) specifically devised cloud-like technologies providing the relevant (distributed and edge-based) dynamic programming, scaling, execution and provisioning environment.

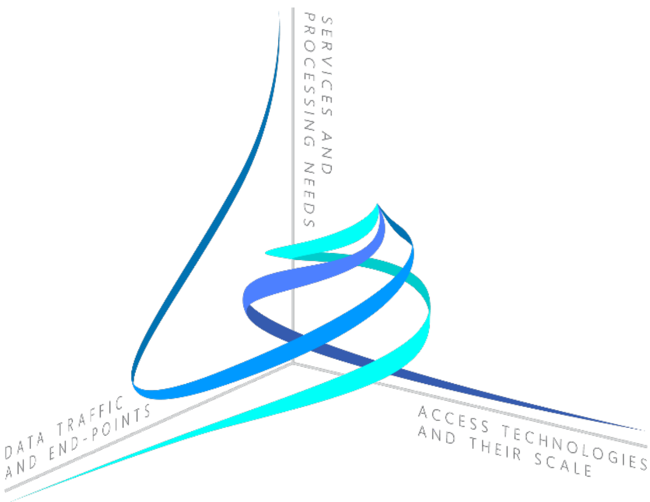


Figure 1: Threefold Convergence

What we target is the identification of elementary and reusable network and processing “primitives”, as well as provisioning “elements”, and their high performance software-based implementation for maximum portability, so as to permit us to rapidly deploy widely different network services where they are needed. As this concept is vital to understanding SUPERFLUIDITY’s position, it is elaborated in the next section.

2.1.1 Towards a Different Architectural Model

In the past, and still in most of today’s deployments, physical diversification of hardware-based network equipment has been the mainstream approach to providing an increased range of network functionalities, including developing different equipment to cater to the specific scale of target deployments. The crucial drawback resides in the proprietary nature of these highly heterogeneous devices, which brings about obvious provisioning and upgrading expense issues, as well as major management complexity in large scale multi-vendor networks.

More recently, the Network Functions Virtualisation (NFV) trend of implementing network services as (virtualised) software running on commodity hardware has gotten a lot of traction. NFV is an important part of a convergence solution for 5G networks but SUPERFLUIDITY goes beyond it by not limiting itself to exploiting NFV in an otherwise ordinary components/interfaces architecture, but tries to foster programmability as a main architectural feature of future 5G networks.

We believe NFV alone is insufficient to solve the convergence problems of 5G networks. NFV comes with the mantra of running everything as software on commodity hardware, which has two key implications: a) the hardware is homogeneous (x86) and b) all in-network processing can be efficiently implemented in software and run on x86. Both these implications do not hold in the real world: custom ASICs are still the way to implement fast network processing, and heterogeneous components such as TCAMs, FPGAs, GPUs can be added to x86 machines to dramatically increase the performance of certain workloads; any solution aiming for convergence should accommodate heterogeneous hardware. Secondly, there are cases where some functions are much more efficiently run in hardware (e.g., packet switching, or wireless protocol implementations with strict packet timings); in such cases, the best option is to create an API that abstracts this complexity and allows external control and innovation while still benefiting from the speed afforded by the available hardware.

Summing up these considerations, our concept to accommodate and leverage this hardware heterogeneity is a “divide and conquer” architectural paradigm for 5G networks. We want to depart from the usual 2/3/4G architectural approach based on monolithic network components/entities and their interfaces, and promote an approach where components are “constructed” via the programmatic composition of elementary “building blocks”. In our view a specific 5G network deployment should comprise the combination of:

1. Elementary radio, packet, and flow processing primitives and events, formally specified and described independently of the specific underlying hardware, but implemented and automatically selected/instantiated so as to match the underlying hardware facilities while taking advantage of the relevant accelerators (e.g., GPUs or FPGAs) if/when available.
2. Platform-agnostic node-level and network-level “programs” describing how these primitives interact, communicate, and connect to each other so as to give rise to specific (macroscopic, and formerly monolithic) node components, network functions and services.
3. A computational and execution environment, supporting the execution (and deployment) of such “programs” and the relevant coordination of the signal/radio/flow/network processing primitives.

With this, operators will formally describe a desired network service as a composition of platform-agnostic, abstract elementary processing blocks; vendors will be in charge of providing efficient software implementations of such elementary blocks, possibly casting them to underlying hardware accelerated facilities; and the (cloud based) infrastructure will provide the brokerage services to match the design with the actual underlying hardware, to control and orchestrate the execution of the elementary blocks comprising the designed service, and to permit dynamic and elastic provisioning of supplementary dedicated computational, storage, and bandwidth resources to the processing components loaded with peak traffic. In short, we believe that one of the project’s major contributions will be applying this architectural approach to 5G networks to create the “glue” that can unify heterogeneous equipment and processing into one dynamically optimised, superfluid, network.

2.1.2 A Realistic Open Programming Paradigm

We aim to dedicate a large part of our effort towards enabling software processing on heterogeneous hardware platforms. To this extent, we will design and prototype selected building blocks as virtualised, software-based network processing tasks, which can be run on heterogeneous commodity hardware anywhere in the network: in the data centre, in edge networks including mobile ones [MEC], in micro data-centres at Points-of-Presence [MICRODC] all the way out to access points, customer premises equipment, or even end user devices. Such processing will need to be instantiated on-the-fly and will be also available to third parties: entities who do not necessarily own the underlying infrastructure such as mobile application developers or businesses.

While the project will make its architecture, APIs and output as open as possible (several partners in the consortium have an excellent track record of releasing major software packages as open source), a pragmatic analysis shows that in reality we need to also accommodate the need for vendors to keep certain parts of their platforms closed: device manufacturers have spent time and money to develop their products internally, and ignoring this or their need to stay profitable would doom a solution to non-relevance.

To address this issue and emerge with a viable and pragmatic approach, our programming abstraction relies on the identification of a set of open reusable elementary primitives and processing tasks. These primitives can be openly available software run on commodity hardware, but this is not compulsory. As long as proprietary building blocks perform processing that is well defined by the API, they can be used to create higher-level, more complex solutions.

Finding the right mix of elementary primitives (so that they can be reused in several different more comprehensive network functions), which are sufficiently complex to permit vendors to differentiate from competitors, and sufficiently complete to permit operators to deploy desired services, is certainly not trivial. To our support, we note that in trying to accomplish our goal we will not start from scratch, but will be able to leverage, recent successful experiences in both the software defined community at large and in pioneering initiatives directly run by project partners in the wireless access and the wired network core domains.

Finally, although we target reuse of functionalities at all levels of the 5G hierarchy, e.g., from radio primitives in small cells and access points, SUPERFLUIDITY's practical approach will be to differentiate radio and signal processing tasks in the cloud-RAN from packet/flow processing network tasks in the rest of the network. Further, the project will target the convergence of control and management primitives needed to ensure the interoperability and integration of multiple tasks (e.g., triggering of per-flow differentiated radio behaviour) in the same network component, as well as similar programmatic composition and orchestration of elementary and reusable building blocks.

2.1.3 Security by design

Last but not least, SUPERFLUIDITY will pay great attention to security and robustness. Anecdotal reports from network operators show that, in many cases, admins do not really know exactly what processing a deployed monolithic network appliance is performing, and are outright hesitant to make any major changes to its configuration or even disable it, for fear of breaking their networks. In other words, network processing today is not only difficult to scale but it actually ossifies the network structure. In future 5G converged networks, flexible in-network processing that is instantiated on the fly has the potential to significantly disrupt network operations unless care is taken to understand the effects changes will have before they are applied. This is very important for the network operators' processing, and crucial for processing that is run on behalf of third parties, where security concerns are paramount.

SUPERFLUIDITY pioneers an approach where the processing blocks will be described in a way that is amenable to automatic offline verification. Given such a high-level description of processing, it is possible to use a technique from compilers called symbolic execution to reason about the effects network processing will have on traffic beyond simple packet reachability. We will contribute with a tool that can be used to check converged 5G networks, together with theoretical models of traditional building blocks that are proprietary (e.g. NATs, firewalls, DPI boxes). Additionally, we will propose mechanisms to help derive and in some cases automatically infer the model from source code given the public availability of such code.

2.1.4 Sample Use Cases Enabled by the SUPERFLUIDITY Architecture

To give a brief flavour of the possibilities of SUPERFLUIDITY, here are the few use cases that it could enable, as envisioned in the proposal:

- Minimum-Delay Cloud storage: Cloud storage has the potential for finally allowing people to throw out all of the clumsy hard drives, memory cards and USB sticks cluttering homes and travel bags. Unfortunately, the high data volumes and relatively low throughputs and high delays to the core/data centre mean that there is still a difference in the experience between local and cloud storage; deploying cloud storage services at the network edges would finally close this gap.
- RAN As A Service: individual functions constituting a Cloud RAN would be readily deployed, following a dynamic life cycle (creation, attachment to core network and antenna site / RRH, hot upgrade, etc.) involving optimised placement decisions about CPU, NIC, memory, hardware-acceleration capabilities; Moreover, a RAN could be flexibly build and adapted to the context using different types of schedulers, different physical layer blocks pointing to various waveforms, etc.
- Localised services: Many services that require some sort of mixing server (e.g., video conferencing, online gaming, to name a few) often end up using a distant one in terms of delay. Instead, such a virtualised server could be deployed on-the-fly at the edge, drastically reducing delays and improving user experience.
- Pooling: User-specific functions attached to various cells (and even baseband computation units [Wer13]) can be pooled in a same host so as to maximise the host load / minimise the required number of hosts; intra-cluster live migration of functions would optimise system KPIs (pooling gain, total radio capacity, energy efficiency, etc.), and would comply with RRM handover requirements (e.g. the current intra LTE handover < 50ms would be readily attained by our technology);
- Edge offloading: One of the drawbacks of mobile devices is their short battery life. Many services (e.g., firewalling, anti-virus software, ad blockers) could be offloaded to the edge to reduce battery consumption.

- Portable signal processing: platform independence would permit portability of signal processing tasks between the edge cluster and the antenna site so as to minimise front-hauling requirements and maximise radio capacity (as front-hauling requirements increase for larger radio bandwidth using carrier aggregation, more massive MIMO, more network MIMO…).
- On-the-fly Monitoring: The owner of the infrastructure could deploy a monitoring service in order to track usage of its tenants' services, or to for instance instantiate a DPI service on a particular suspicious flow.
- Virtualised CDN operators: It is well known that the performance of CDNs improves the closer that content is from users. This, however, is an expensive proposition, and so restricts all but the biggest players from the market. Instead, newcomers could deploy (virtualised) content caches at network edges, effectively renting out infrastructure and growing it as their business grows.
- And many others: For instance, context-aware services that take advantage of location information, low-delay services such as augmented reality (e.g., Google glass) or SIRI, edge-based video analytics, and application-aware performance optimisations, as described in an ETSI white paper on Mobile-Edge Computing [MEC].

However, during its first year of work, the project elaborated a more comprehensive set of 23 use cases, covering themes such as wireless access, mobile edge computing and on-the-fly monitoring. Each use case is described, along with its business and technical requirements in D2.1; the main outputs of D2.1 are also reported in this document in section 4.1.

2.1.5 Expected Benefits

As anticipated in the vision statement, SUPERFLUIDITY aims to achieve four key characteristics that together will enable dynamic processing in 5G networks: (1) location-independence, so that the processing can be placed in a number of different places and networks along the end-to-end path, as deemed optimal by the beneficiary of that processing. (2) time-independence, whereby processing can be deployed or moved near instantaneously, without end-users noticing or traffic being affected; (3) scale independence to achieve seamless scaling by decoupling network services from their scaling, and (4) hardware-independence, such that the processing can run efficiently irrespective of the different kinds of underlying, possibly heterogeneous, both commodity and proprietary hardware.

The resulting superfluid network is a network where multi-tenant, virtualised software-based network services co-exist with proprietary network functions, allowing network operators and third parties to quickly stitch together complex functionality that achieves high performance. Software processing will run on common, shared heterogeneous commodity hardware infrastructure deployed throughout the network. Operators will have the ability to instantiate such services on-the-fly, whenever needed (i.e., in milliseconds), and run potentially thousands of them on a single inexpensive platform (thus supporting a large number of concurrent users), migrating them near-instantaneously (in milliseconds, allowing for transparent adaptation to changing requirements and network conditions) and deploying them across a wide range of hardware and locations, ranging from base station and multi-cell aggregation sites all the way to data centres in the core network.

2.2 Potential Market

The advances in this project can influence and create new potential markets, such as in Internet of Things, and Big Data computing and Entertainment (gaming, multimedia).

For instance, a particular market that could benefit from the advances in SUPERFLUIDITY is the media industry that is increasingly moving to over the top (OTT) video delivery. In this case of OTT video content, owners can stream video directly over the current Internet infrastructure. According to a recent report the current over the top video streaming market amounts to around 26 billion USD in 2015, expected to nearly double to 51.1 billion in 2020 [HOLLY]. Famous OTT delivery includes companies like NETFLIX, but many operators and content owners are deploying OTT delivery. Large scale OTT video on demand streaming systems, often demand massive and duplicate storage of video content. The media industry typically encrypts its content for different users, and uses different delivery protocols for different devices. The location independence paradigm offered by the SUPERFLUIDITY platform enables video streaming functions to operate closer to the users. This can enable media operations on the edge such as Trans Muxing (protocol specific format generation), Digital Rights Management (encryption of video content) and/or ad insertion. Centralized execution of these highly personalized functions increases required backend bandwidth and storage. This inefficiency has made over the top video streaming (OTT) an unprofitable business for many content owners and distributors (due to high CDN bill, network bandwidth and storage). The Location independency introduced by the SUPERFLUIDITY architecture enables optimization in location of video processing operations (transcode, storage, DRM, personalization) leading to hugely reduced costs. By executing functions on the edge of the network, more efficient caching and reduced backhaul traffic can be achieved.

Further, SUPERFLUIDITY Orchestration and service Management and virtualization enables network programmability. This creates opportunities for telecom operators to increase their revenue by providing virtualized network capacities to OTT services. Further, the time independence (fast instantiation of services) and scale independence enable much better provisioning and adaptation of services that demand large amounts of bandwidth and computational resources. We expect that the SUPERFLUIDITY architecture can improve revenue and efficiency in the media industry, for content owner, distributor and telecom provider. Beyond this, users will benefit from increased bandwidth, reduced latency and so forth. In addition telecom operators can benefit by opening up API's to their network enabling charging for network and compute infrastructure, receiving a more fair share in OTT revenues than currently possible.

In addition, as mentioned above, a key differentiator between 4G and 5G is the idea of convergence of the whole network including the Cloud. Cloud solution providers can benefit from the converged vision of 5G. Cloud incorporates a lot of the core concepts that are of interest for 5G, including managing virtualised resources and varying workloads across a heterogeneous hardware environment. Other than the existing customer base, which will benefit from technology improvements that enable better or more efficient use of resources, the improvements that will allow greater convergence with the Telecommunications space will mean that there are more potential customers and in particular the larger, enterprise-focused businesses.

2.3 Relationships to current solutions and initiatives

SUPERFLUIDITY is one of the 19 projects belonging to Phase 1 of the 5G Infrastructure Public Private Partnership (http://5g-ppp.eu/), a 1.4 Billion Euro joint initiative between the European ICT sector and the European Commission to rethink the infrastructure and to create the next generation of communication networks and service. Such projects are described at https://5g-ppp.eu/5g-ppp-phase-1-projects/.

SUPERFLUIDITY is working together with the other projects to reach the aims of the 5G PPP. Important collaborative achievements of SUPERFLUIDITY include: i) contribution to both white papers on 5G Architecture, already available (https://5g-ppp.eu/white-papers/), ii) contribution to a second white paper on Software Networks, also already available; iii) work for achieving commonly agreed KPIs (see Section 3.5); iv) demonstration of exemplary functionality through testbeds (see Section 4.3.6) presented in joint events, such as the 5G Global event (https://5g-ppp.eu/2nd-global-5g-event/).

SUPERFLUIDITY is also linked to important international open source activities where several SUPERFLUIDITY partners are strong stakeholders: i) OpenStack, the world's fastest-growing open cloud platform and developer community, receives strong contributions from REDHAT and INTEL (part of the platinum members and technical committee), NEC (part of the gold members) and NOKIA and CITRIX (part of the corporate sponsors); ii) OpenDaylight, the open platform for network programmability, receives strong contributions from INTEL and REDHAT (part of the platinum members), CITRIX and NEC (part of the gold members), NOKIA (part of the silver members) and TELEFONICA (Dr. Pedro A. Aranda is a member of the User Advisory Board); iii) Xen is a collaborative projects of the Linux Foundation. INTEL and CITRIX are members of the XEN project and have representatives in the XEN advisory board, ONAPP has attended the latest XEN Hackathon held at ARM; iv) OPNFV, the new open source project focused on accelerating the evolution of Network Functions Virtualisation (NFV) in terms of Virtualised Infrastructure Management (VIM) and application programmable interfaces (APIs) receives strong contributions and steering from NOKIA, INTEL, NEC and REDHAT (part of the current board of directors); OSM orchestrator, supported by the ETSI OSM community, the Superfluidity RDCL 3D tool has been taken as the basis for the new web GUI of OSM lightweight build that will be released in May 2018.

In addition to collaborating with other 5G-PPP Phase 1 projects, as part of the relevant 5G-PPP working groups, Superfluidity has already started to disseminate project results towards 5G-PPP Phase 2 projects. This was accomplished in two ways: i) Superfluidity members continued to actively participate to 5G-PPP working groups, during the important time when Superfluidity overlapped with 5G-PPP Phase 2 projects. ii) Through participated to workshops, where project partners were invited to talk about reusable outcomes. Given the strong results of the project on the open source and standardization fronts, Superfluidity results are expected to be reused and leveraged in current and upcoming related R&D projects (NGPaaS, 5G-EVE, 5G-VINNI).

2.4 Progress beyond the SoA

In this section we outline the project's progress beyond the state of the art in a set of research areas that we believe are more of importance to SUPERFLUIDITY. This progress is the one expected from the whole duration of the project. More detailed and current information about the project's progress beyond the state of the art, as well as the project's innovation potential, can be found in deliverable D8.8 (Final Report on Innovation and Exploitation Actions).

Cloud Networking: SUPERFLUIDITY will aim to meet the stringent requirements imposed by future 5G networks by designing and implementing a superfluid, converged network architecture that is location, hardware and time-independent. The work will push the boundaries of what is currently possible with virtualised, software-based packet processing (10-40Gb/s and higher, extremely fast service instantiation and migration in milliseconds, massive numbers of concurrent virtualised services on a single platform, significant power reductions, etc.). The goal is to bring the advantages of cloud and software-based networking to 5G networks so that services can be deployed whenever and wherever they are needed, and to leverage the availability of inexpensive, off-the-shelf hardware in the process.

Network Services Decomposition and Programmability: SUPERFLUIDITY will devise programming abstractions specifically targeted to 5G functions. The API design work will address three programming levels: service, function, and processing levels, and will attempt to maximise viability by reusing existing standard (when applicable) or research community best practices. Work will on one side target the definition of 5G specific actions and events, and on the other side will address the specification of the constructs needed to combine and orchestrate a desired execution of such actions (conditioned on the arrival of events). Particularly promising and forward-looking is SUPERFLUIDITY's approach of combining block-based composition abstractions (such as those exploited in Click routers, in some software defined radio architectures, or emerging in the ETSI NFV work on service chaining) with event-driven programming paradigms such as basic match/action based approaches or more powerful stateful abstractions based on extended finite state machines.

RAN Cloud and Mobile Edge Computing: Beyond the current vision of a static RAN function fully located in one "edge computing" place, SUPERFLUIDITY will support the ability to modularly "hot" replace eNB functions (such as scheduling) and to permit migration of such functions between edge clouds and the antenna subsystem, so as to balance algorithmic complexity with front-haul capacity. SUPERFLUIDITY will also transcend current Mobile Edge Computing vision where non-RAN functions (local caching, CDN, etc.) are envisaged to be co-located only at the eNB by enabling their migration between the RRH and the edge cloud, to maximise their performance.

Automated Security and Correctness: SUPERFLUIDITY will provide a two-pronged, complementary approach to security. First, it will go beyond the state of the art, providing a pre-deployment checking system that will ensure that virtualised network services do not negatively affect the network nor other tenants; unlike approaches in the literature, the system will be both scalable and stateful, able to model most types of services. Second, SUPERFLUIDITY will implement a post-deployment system that will learn the behaviour of traffic and detect any anomalies, thus providing a further security mechanism in cases where the checking system does not have information about the processing performed by a network function, or when static analysis is inaccurate.

3 Creating the Vision

We believe that the time is ripe to tackle the next stage in the network's evolution, towards what we termed the superfluid network. A number of different trends have recently started to come together to render such architecture an incrementally deployable possibility. First, hypervisor-based virtualisation technologies such as Xen or KVM provide the necessary isolation needed to be able to share common infrastructure across different, possibly competing, tenants. In addition, other virtualization technologies, such as containers, offer alternative solutions that may fit different scenarios.

Second, the availability of an increasing range of commodity hardware at affordable prices means that network processing can be carried out with high performance on a number of different heterogeneous hardware (e.g., GPUs, TCAMs, SSDs, FPGAs, etc.) and platforms ranging from high performance blade servers and stand-alone x86 servers to small-size, low-energy footprint microservers (e.g., CubieTrucks, Raspberry Pis, fit-PC, etc.). The micro-servers open up the possibility of pushing software and IP-based network services to the very edge of access networks, in places where energy consumption or space constraints might render the deployment of traditional servers impossible.

Finally, the accelerating pace of deployment of such platforms not only in the core of the network, but also at the edge, at network PoPs [MICRODC] or even next to base stations with different access technologies [MEC, SMART], brings the concept of running network processing “anywhere” in the network one step closer to reality.

One of the aims is to open-up the proprietary and difficult-to-innovate equipment and protocols that are deployed in heterogeneous access networks by relying on SUPERFLUIDITY’s “basic blocks” concept to converge towards a common architecture and base protocol (IP) just like the rest of the network has done. While the RAN industry is yet to move towards open APIs, the success of Openflow has created large waves, and their ripple effect is likely to spread in the near future.

Figure 2 shows a conceptual view of the SUPERFLUIDITY 5G network. At the top of the picture we show a physical view with a set of SUPERFLUIDITY platforms (in red boxes) running on different types of hardware (microservers, small racks, larger x86 deployments). These platforms are set up at different points in the network: next to base stations and aggregation sites in access networks, at micro data centres at Point-of-Presence (PoP) sites in aggregation networks, and at full-fledged data centres in the core network. Each of these platforms is a multi-tenant and shared infrastructure, and network processing can be instantiated by third parties on-the-fly, when and where it is needed.

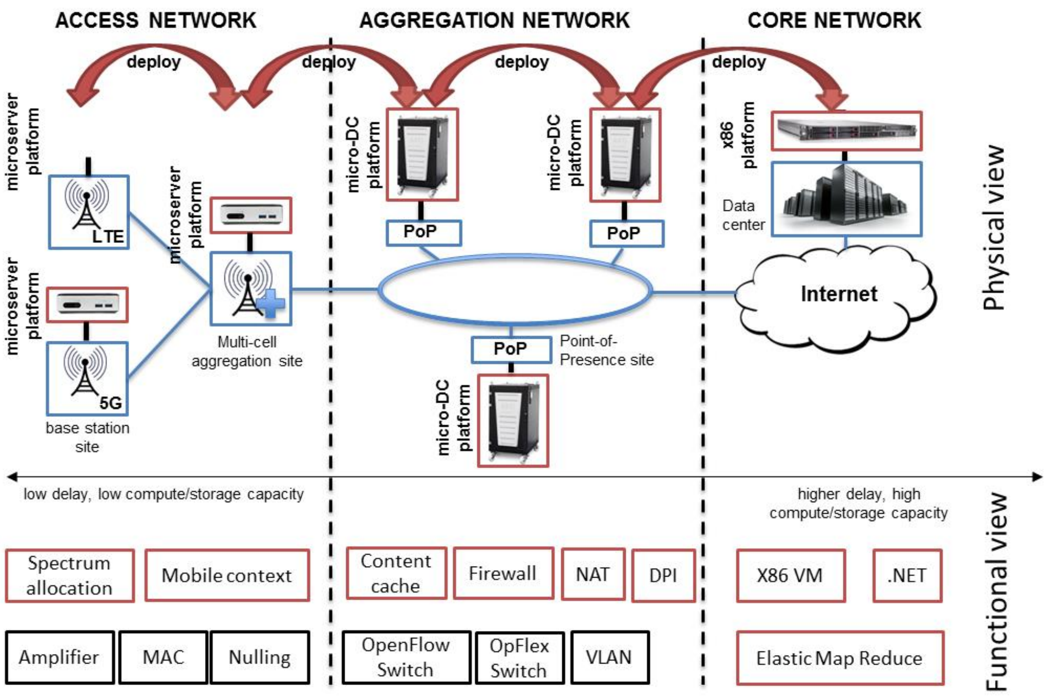


Figure 2: A conceptual view of the SUPERFLUIDITY 5G network

An orchestrator (not shown in the picture for the sake of clarity) is in charge of providing an API to platform users and of deploying the necessary network processing in a safe, high performance fashion. End-users, application developers and any SUPERFLUIDITY tenant can decide the trade-off between low-delay access near the edge (left-hand side of the figure) and high compute/storage capacity near the core (right-hand side).

At the bottom of the figure, we provide a functional view, based on decomposing processing into a series of functional blocks, within different heterogeneous environments. These functional blocks (that will be referred to as "Reusable Functional Blocks" – RFB, for a detailed definition see below in section 3.5.1 and in I3.1) may range from very basic and "small" functionality to complex and aggregated functions. As shown in the Figure, both hardware (black boxes) and software (red boxes) functions can be represented in the functional view. These functional blocks are then used by the operator and its customers to deploy more complex processing that relies on combinations of existing blocks. It is possible for the operator to apply static checking techniques to understand the impact of deploying certain processing.

To achieve this high-level architecture and vision, SUPERFLUIDITY will organise its work around five central pillars, presented here in a bottom-up form:

Pillar 1: Superfluid, Converged Architecture Design

The work will be driven by real-world requirements derived from the project's operator partners, but also from other EU operators, as well as from the definition of novel use cases such as the ones listed in D2.1. The aim is to provide a converged, uniform and superfluid network that allows virtualised, software and IP-based network services to be deployed as-needed and throughout the end-to-end path. For access networks in particular, the idea is to push IP and software processing all the way to the edge, leaving base stations to provide basic heterogeneous connectivity and for the rest of the network to operate in a converged, IP-based fashion; Deutsche Telekom's Terastream architecture, for instance, is already starting to push IP all the way to the edge in order to simplify management, reduce equipment costs, improve service deployment and remove ugly traffic breakout kludges [TERA]. The high-level goal is to provide an architecture that is location, time, and platform independent while providing high performance, ease-of-use, and security mechanisms to prevent the deployed services from harming each other or the network as a whole.

Pillar 2: Platform-Independent Block and Function Abstractions

SUPERFLUIDITY will seek to decompose (1) heterogeneous hardware and system primitives into block abstractions and (2) network services into basic functional abstractions. It will then implement a provisioning framework that will match function abstractions to the available hardware/block abstractions in order to automatically derive high performance and meet end-user SLA requirements, i.e., without forcing developers to have to understand low-level system details. Figure 3 depicts the process, where network processing is decomposed into function abstractions, and these are then matched to the underlying hardware. For instance, a simple firewall could be decomposed into functional abstractions to do rule matching, packet dropping or packet forwarding. These functions would then be matched to the available underlying hardware, for instance assigning the matching function to a GPU in order to parallelise and thus speed up this look-up.

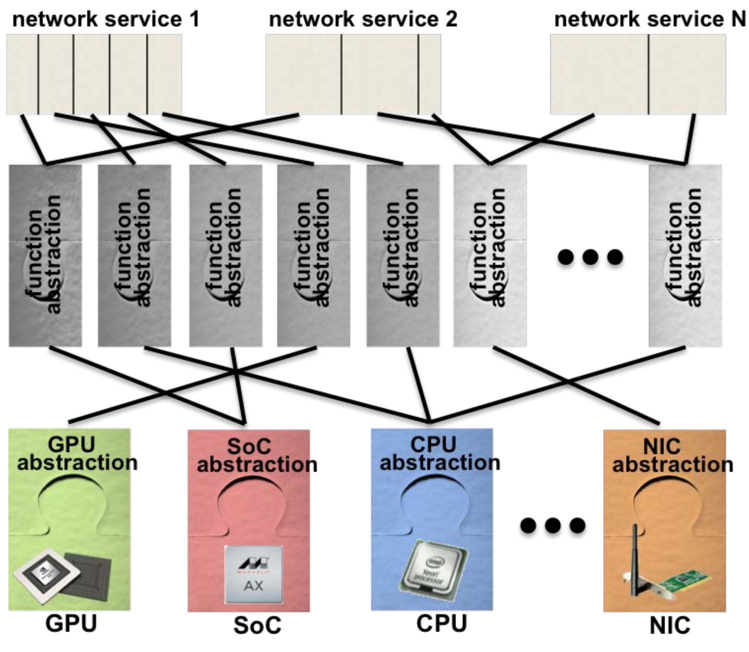


Figure 3: SUPERFLUIDITY function decomposition and matching process

Pillar 3: High Performance Block Abstractions Implementation

This pillar consists of deriving block abstractions for each of the underlying hardware components, and to optimise their performance. For instance, an abstraction could be defined for a System-on-Chip (SoC), describing its ability to perform 40Gb/s rule matching on IP's 5-tuple. A further abstraction could wrap the capabilities of an SSD drive: low-storage but high access speeds. Yet another one could abstract a GPU with high parallelisation potential but poor performance when using complex data structures (e.g., radix-trees, commonly used in IP forwarding). Beyond the abstractions, the work will involve an investigation and measurement study of the available commodity hardware, virtualisation schemes and packet I/O frameworks (Intel DPDK or netmap, to name a couple). This pillar will also look at revisiting commodity operating systems' concepts. For instance, processes and threads (which may not be congruent with the pipeline approach of packet processing) and the typical user/kernel split (which may result in unnecessary overheads given that the software already runs within an isolated virtual machine) may prove redundant.

Pillar 4: Superfluid Platform

This pillar will be in charge of putting together all of the work on abstractions into a single, coherent, SUPERFLUIDITY platform able to run virtualised network services for different tenants on shared infrastructure. The platform will implement an API for deploying services, and an algorithm for deriving the best way to match the necessary services to the underlying block abstractions such that SLAs can be met. Further, this pillar will look into optimising the number of network services that can be concurrently run, setting a target of as many as 10,000 virtual machines and/or containers on a single, shared, multi-tenant server; such high-capacity on an inexpensive server would make it easier to achieve the concept of running services anytime, anywhere. Finally, the work will include the ability to almost instantaneously instantiate services (as low as a few milliseconds is the target), as well as to migrate them quickly enough that the process is transparent to end-users and their connections (e.g., in hundreds of milliseconds or less). Clearly, some of these figures will depend on the underlying hardware, but the goal is to make the network I/O as efficient as possible so that even low-cost hardware (e.g., micro-servers running ARM processors) can act as SUPERFLUIDITY platforms.

Pillar 5: Orchestration Framework

The last pillar is in charge of gluing the different SUPERFLUIDITY platforms deployed throughout the access, aggregation and core networks into a coherent architecture. First, it will provide an API through which users can request the deployment of network services along with SLAs to which the platform(s) should conform; where possible, we will aim to integrate SUPERFLUIDITY's orchestration framework into existing management frameworks such as OpenStack. Second, this pillar will implement provisioning algorithms that will ensure that the required network processing is deployed at platforms and places in the various networks such that the SLAs are met (and should the network conditions change to migrate the processing in order to still comply).

Finally, in terms of security, the great flexibility derived from software-based network processing opens the door to introducing new problems and threats into networks. To address them, SUPERFLUIDITY will take a two-pronged, complementary approach. First, for pre-deployment security it will embrace static analysis to check network processing for safety before it is instantiated. Static analysis tools have shown great promise in checking programs for faults; SUPERFLUIDITY postulates that the same tools can be adapted to great effect for network processing. Static checking will remove the need for firewalls and other sandboxes, greatly reducing the overheads. For instance, pass-through virtualisation would be feasible to use for untrusted tenants, as long as their processing is deemed "safe" by static analysis. Second, as a backup to static analysis, this pillar will implement post-deployment security through anomaly detection algorithms that will ensure the correct functioning of the network.

3.1 Project Objectives

The objectives of the project as reported in the Description of Work are the following:

Objective 1	**Novel 5G data plane processing architecture** - Design a flexible, open and programmable 5G data plane processing architecture and relevant APIs for network functions' convergence.
Work carried out during the second year and measurable details	The **Superfluidity architecture** has been finalized. It includes the concepts of RFB (Reusable Functional Block), REE (RFB Execution Environment) and RDCL (RRFB Description and Composition Language). In the proposed architecture, RFB can be nested across heterogeneous REEs. SDN/NFV based ***Multi-access Edge Computing*** **(MEC) prototype**, building an edge IT environment, for terminating and pass-through applications, with different virtualization technologies support. **Kuryr networking to support heterogeneous VM/containers virtualization scenarios**: Thanks to the kuryr work, containers can take advantage of Neutron features already developed for VMs. Thanks to the Superfluidity work done at kuryr, now it is possible to have mixed VMs and containers environments, regardless of if containers are running in baremetal servers or inside VMs (for higher security and pure multitenancy), avoiding double encapsulation (therefore speeding up data path) and providing fast instantiation times. Note this also enables operator to move from VMs to containers at a different pace, i.e., not all the components needs to be migrated to containers, definitely not at once.
Overall status	The architectural concepts for the open and programmable data plane processing have been finalized. The Superfluidity architecture is described in deliverable D3.1. The design/implementation of the APIs is described in D5.3.

Objective 2	**Converged 5G platform** - Design, implementation, and evaluation of a unified and high performance distributed cloud platform technology for radio and network functions support and migration.
Work carried out during the second year and measurable details	**Cloudification of the RAN**. The RAN was re-factored using RFB principle where the RAN protocol stack was broken down into a set of elementary functions that can be combined together, assigned with target performance parameters, mapped onto the infrastructure resources, and finally delivered as a service. The prototype is based on container virtualization technology, offering performance comparable to bare metal. The usage of Container simplifies function portability as well as migration between different computing nodes. The Remote Radio Head (RRH) is also enhanced with functionality, which is able to send IQ samples over Ethernet. **Making the fronthaul network re-programmable**. A fully re-programmable fronthaul solution, which can select the routing path according to the requirement of the flow in terms of latency and throughput is designed and prototyped. It is based on an overlay network composed of Open Virtual Switches. With the use of an SDN controller (in our case ONOS), and the intent programming, we succeed in connecting the enhanced RRH and the EDGE cloud in a software manner and making it fully re-programmable. These results are described in I4.1b.

Overall status	The softwarization/cloudification of the RAN has been demonstrated as a prototype. A prototype for the fronthaul network supporting heterogeneous transmission technology is also available. The results have been described in deliverables D7.2 and D7.3.

Objective 3	**New Algorithms and functions** - Design, development and evaluation of algorithmic and design improvements for radio processing tasks, flow processing primitives, and service optimisation.
Work carried out during the second year and measurable details	**Flow/packet processing functions allocation to processors**. In the context of the FastClick packet processor, algorithms for low-level resource allocations have been developed. A Network Performance Framework (NPF) to benchmark network applications, and analyse the results using statistical analysis and machine-learning based tools has been designed and implemented. **Sharing data plane processors via endpoint congestion control**. A novel approach to resource allocation is proposed, based on the idea is to treat CPUs as if they are normal links and let CPU endpoint congestion control share them, instead of using resource reservations. We explored what happens if we apply this idea and finds that there is a modest drop in throughput because of Network Functions contention for CPU, and that the allocation of throughput is not fair. We explored possible solutions to get proper allocation based on weighted RED. **Modularized and scalable C-RAN baseband functions** with platform-aware optimization and adaptation capability employing dynamic task clustering and scheduling approach (T4.3, T5.1, T5.2)
Overall status	The project has achieved interesting results described in D4.3, D5.1 and D5.2.

Objective 4	**Ultra-fast and efficient virtualisation** – Design, implementation and evaluation of beyond the state of the art quickly instantiable, low memory footprint, and high performance virtualization technology
Work carried out during the second year and measurable details	**Unikernels**: The work comprises several tools. First, Tinyx builds minimalistic Linux distributions, considerably reducing their boot times and memory footprints. Second, Unilinux supports running existing application transparently within the Linux kernel, reducing overheads and improving metrics such as throughput. Third, Unicore allows for automatic building of specialised OSes and VMs, reducing boot times, memory footprint, and increasing throughput, among others. Tinyx is ready and finished and Unilinux/Unicore are under active development. **Superfluidity platform**: The work consists of a significant re-architecting of the Xen virtualization system to (1) remove its centralised database (known as the XenStore and comprising a single point of failure and an important source of overheads) and (2) implement a split toolstack. These two systems are complementary and combine to allow instantiation times of a few milliseconds, migration times of 10s of milliseconds and the ability to run thousands of guests concurrently on a single inexpensive server. **NFV platform:** Implements changes to the hypervisor to include the forwarding path of the back-end software switch in the hypervisor as well as parts of the drivers. The result is high

	throughput (including for long service chains) by 36%-83% depending on the NF and the ability to closely match required performance guarantees (with deviations of only 3.5%) **CPU sharing:** We have designed and implemented mechanisms for fair CPU sharing which rely on sending congestion control proportional to per-packet cost for each flow. This enables endpoint congestion control (e.g. TCP) to react appropriately and share the CPU fairly, thus improving cumulative throughput for the platform.
Overall status	The project has reached its goals as described in deliverable D5.2

Objective 5	**Hardware adaptation and abstraction** – design and development of technologies and interfaces to exploit and integrate customized hardware.
Work carried out during the second year and measurable details	**Design of a MIPS based PMP (Packet Manipulation Processor)** architecture (MIPS is a RISC instruction set processor architecture). Implementation of the PMP MIPS simulator. Simulation of the PMP MIPS version with 3 use cases: IPinIP encapsulation, Network Address Translation, ARP reply generation.
Overall status	The results on the PMP are described in D4.1

Objective 6	**Control and provisioning framework** – Extensions of existing and widespread frameworks for platform's management, control, and elastic provisioning.
Work carried out during the second year and measurable details	**Extensions to Manage IQ**. ManageIQ is a management project that enables managing containers, virtual machines, networks, and storage from a single platform. ManageIQ allows the inter-connection and management of different existing clouds: OpenStack, Amazon EC2, Azure, Google compute engine, VMware, Kubernetes and OpenShift. In Superfluidity we have extended ManageIQ with Ansible playbook execution support. This supports enables a more fine grained application lifecycle management as well as adding the possibility of not only managing different sites (one at a time), but to perform actions that spawn across several of them. **Throughput, Anomaly, Latency and Entropy (TALE) and Key Platform Indicator (KPI)**. Mapping methodologies have been developed in order to exploit system telemetry and improve the system performance and predictability. Application of those techniques have been demonstrated on a video transmuxing service, where TALE demonstrated an improvement in performance of 2.2X and KPI mapping shown a predictability of the workload behaviour with a confidence higher than 90%. **Unikernels orchestration**. The Superfluid platform technologies (e.g., ClickOS/LightVM) have been integrated into existing orchestration frameworks. In particular, we focused on the adaptation and optimization of VIM (Virtual Infrastructure Managers) for Unikernels. In greater detail, we have extended OpenVIM to support the ClickOS Unikernel (and more in general the XEN virtualization) in a backward compatible way, starting from the latest release of OpenVIM. We have also prototyped the support of Unikernels in OpenStack and Nomad and provided performance measurements of orchestration time for the three considered platforms (OpenVIM, OpenStack, Nomad).

	Automated scaling and load balancing mechanism for VNF. The Superfluidity platform supports fast scaling mechanisms of VNF, i.e., increasing/decreasing the number of VMs that are allocated to the VNF and the resources allocated to them can be done in a much faster manner utilizing the Superfluidity platform. In this study we devise mechanisms that are facilitated by the Superfluidity technologies which can learn the system behaviour and its expected performance, based on the past and can dynamically adjust (increase or decrease) the number of VMs allocated to a VNF based on the users' requirements. The scaling decision should minimize the consumed resources, while in keeping with the application required SLA. In addition, in order to be more efficient to changes in the demands, and more flexible to changes, in conjunction with the scaling decisions, we are also facing a load balancing challenge, steering the traffic flows to the different VMs. We note that, in this study, we did not consider the scale up and down operations (e.g., adding/removing CPU cores or memory allocation to a given VM), which are less common in NFV use cases.
Overall status	The project has achieved the results as reported in deliverable D6.1.

Objective 7	**Security framework** – Security abstractions and mechanisms will be developed to control the access to, and execution of, the network processing functions, and to prevent third-party network functions from having a negative impact on other clients' functions, the network, or the Internet at large
Work carried out during the second year and measurable details	**Static verification and provable correctness of 5G service configurations**. We have worked towards translating P4 programs to SEFL, and to allow operators to express their policies easily via the novel NetCTL language. We plan to run symbolic execution on the SEFL version to check desired properties such as isolation, while ensuring that the P4 and SEFL programs are equivalent; this means that properties that hold on SEFL also hold on P4 too. We have also continued our integration of Symnet into OpenStack to ensure both correctness and policy compliance at the tenant view of the network and at the network view. **Secrecy Rates and Outage in Multi-User Multi-Eavesdropper Systems**. We analysed the vulnerability of the C-RAN architecture to unknown eavesdroppers in the presence of a large number of legitimate users, developing codes to mitigate this vulnerability and assessing their performance. For example, we discussed the secrecy and outage rate probability when transmitting to a preferable legitimate user in the presence of multiple unknown eavesdroppers. **Monitoring the Context of System Calls for Anomalous Behaviour and Flow Classification**. We devised a novel universal anomaly detection algorithm, which is able to learn the normal behaviour of systems and alert for abnormalities, without any prior knowledge of the system model, or of the characteristics of the attack. In addition, we developed a novel fingerprinting technique that enables classification of ongoing flows according to the various applications, which created them with very high accuracy while requiring a minimal number of traffic flow sampling rules. Both techniques are tailored to virtualized environments, where network functions can be placed and replaced continuously.
Overall status	The project results have been reported in deliverables D3.1 and D6.1.

Objective 8	**Contribution to standardization** – Feed Superfluidity results into the relevant standards bodies and communities working on de-facto standard tools
Work carried out during the second year and measurable details	Project partners (TID, CNIT, UPB, NEC, BT, ALB, NOKIA-IL, USTR, NOKIA-XHAUL) contributed significantly to the activities of standardisation bodies (IETF/IRTF, ETSI NFV, ETSI OSM, ETSI MEC, OASIS TOSCA, ISO MPEG, ECPRI). The specific contributions have been documented in Deliverable 8.7 (Final Report on Standardization and Open Source Contributions).
Overall status	The project has fulfilled its objectives.

3.2 Scientific and Technological Vision

The explosive growth in mobile data traffic, together with the proliferation of 'smart' connected devices such as smartphone, personal fitness trackers etc. is forcing service providers to transform their networks from one which is static and is inflexible and is based on monolithic fixed appliances to one which is dynamic, flexible and fluid in nature. At the centre of this network transformation is pervasive deployment of NFV and SDN technologies. While NFV is being adopted and rolled out by many service provides today, its role in the context of 5G will be vastly expanded and will become a foundational technology.

The characteristics of 5G networks will be significantly different to the current 3/4G networks. 5G is expected to provide an exponential increase in network capacities, support a much wider variety of use cases, improved operational characteristics such as lower latency and jitter, ubiquitous connectivity through increase connection densities as well and increase reliability and resilience. 5G networks will be highly virtualised and support high levels of flexibility through a software driven programmable network environment. 5G networks will fully embrace open technologies (both open source and derivative commercial offerings) from a diverse ecosystem built using open standards. This new ecosystem will provide support for increased and more rapid innovation, new value creation and will allow the development of new business models and revenue opportunities.

The advent of 5G will herald a new connectivity paradigm. The emergence of the smartphone has significantly changed the way in which we use our phone moving away from the consumption and generation of voice traffic to data. In Cisco's recent Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015–2020 White Paper it is predicted that mobile data traffic is expected to grow from 2.5 Exabytes of traffic per month to more than 23.4 Exabytes per month by 2019. Mobile users are increasing using applications that have high-bandwidth requirements e.g. on-demand video streaming, thus generating increasing levels of data traffic over the network. In the 5G paradigm everything will be connected, mobility is pervasive, usage and connectivity varying significantly by location and time. 5G is set to realise a vision of a hyper connected world in which the network is highly heterogeneous with convergence across multiple access technologies (wired and wireless) across a significantly expanded range of licence and unlicensed spectrums making the access network much more adaptable to user contexts and needs. The access networks will be much more spectrally efficient allowing them to support much higher densities of device connections, higher data rates and with more predictable characteristics such as latency, jitter etc.

5G will drive the proliferation of the IOT devices. Cisco have predicted that there be 50 billion by 2020. This means there will be almost 7 IOT devices for every human on the planet. 5G will need to have appropriate scalability which need be achieved through technologies such as virtualisation to enable scaling in orders of magnitude and with minimal cost implications. 5G will have to deliver low latency which will an order of magnitude lower than in 4G networks. In some specialist 5G uses cases such as tactical feedback for robotic surgery a latency of 1ms is required. Due to range and scale of expected use cases in 5G, control becomes a significant challenge and requires a much larger control plane.

To meet the diversity of anticipated use cases/services, heterogeneous access mechanism, and quality of service requirements will require the ability to provide network slices enabled through SDN capabilities in order to ensure required capacity, throughput, security, latency etc. This requires a significant movement away from proprietary, vertical integrated and specialised hardware to open platforms that offer significantly more flexibility, support from programmability with a high level of automation and autonomy. The foundational adoption of NFV and SDN is fundamental to realising these capabilities. An important differentiator of these technologies is their utilisation of an open source approach. The approach leverage significant investment from industry players such as Cisco, RedHat, Intel etc., solution providers, and community developers to provide a unified approach to implementation. Projects such as OpenStack, OpenDaylight, OPNFV, OpenFlow, Open vSwitch, DPDK etc. will be instrumental in providing the technology building blocks for 5G. These software-based building blocks will enable service providers to better monetise their network assets.

The advent of Cloud RAN will be another key architectural change for 5G. The approach allows aggregated traffic from radios to be processed by virtualised applications running on standard high volume servers. Functions which have flexible timing constraints such as mobility, subscriber management etc. are suitable for centralisation and virtualisation in a C-RAN environment.

The adoption of open and non-propriety technologies in 5G will afford new opportunities for many companies. For example, network infrastructures can be realised on standard X86 high volume services which offer various capabilities to support virtualised environments. For example, Intel is providing its Intel Open Networking Platform (ONP) as a reference architecture that combines leading open source technologies and Intel technologies such as QuickAssist Acceleration as building blocks that can be used by the ecosystem to create optimised solutions for SDN and NFV workloads to meet the needs of 5G.

3.3 Market Vision

The market vision for 5G is a technology ecosystem that is built around open standards, open source technologies and standard high volume hardware platforms. This move towards a more open and standards-based environment will encourage new entrants into market both from the infrastructure technologies and service provisioning perspectives. Increased diversity in the ecosystem drives increased and more rapid innovation, new value creation by reducing barriers to entry and adoption, and will enable the realisation of new business models and use cases.

The initial target market for 5G is likely to be fixed wireless broadband. Verizon and AT&T have indicated they are targeting initial commercial deployments in the 2017 timeframe. KT Corp who are the telecom sponsor of the 2018 Winter Olympics located at Pyeong Chang in South Korea have plans to demonstrate 5G technologies supporting broadcasting and data services. This is expected to act as a catalyst for the launch of 5G service by Korean operators by 2020 provided the standards are in place by then. In a similar timeframe NTT DoCoMo plan to launch 5G services in time for the 2020 Olympics [JAN15].

From a standardisation perspective, the 3GPP started the standardisation process in September 2015. It is expected that the Release 15 should be available in the 2018 timeframe. The Release 15 work items should form the basis for the first phase of 5G in the 2020 timeframe.

3.4 Societal Vision

5G is envisioned to deliver an end to end ecosystem that supports a diverse range of users and use cases enabling a fully mobile and connected society. The vision of 5G is support to a diverse range of use cases which affect the daily lives of most citizens. These include smart grids, smart utilities and better water quality monitoring and domotics which will impact and improve people's domestic lives. In transport domain we will have smart mobility with autonomous cars and intelligent traffic management. From a personal perspective smart wearables will enable people to live fitter and healthier lives. E-healthcare will also improve people health outcomes and improve the democratisation of healthcare by giving people access to healthcare specialisations such as remote surgery, specialists consultations etc. which are independent of location.

3.5 Metrics / KPIs

In this section we report the key performance indicators defined by the 5G Infrastructure PPP; we also detail first what is the envisaged contribution of SUPERFLUIDITY to each KPI (see third column) and then we give more details on SUPERFLUIDITY's contribution to four KPIs, which are believed to be the most relevant for our project. The first three are those labelled as P1, S3, B3 in the following table; the fourth one is "End-to-End latency of < 1ms", and is not reported in the official 5G PPP table below, defined in 2013, but it was added afterwards to those of interest to 5GPPP (see https://5g-ppp.eu/kpis/).

3.5.1 Performance KPI

Table 1: Performance KPI

KPI		**Relevance** (High/Medium/ Low / N.A.)	**Details on planned project contribution towards achieving the KPI**
P1	Providing 1000 times higher wireless area capacity and more varied service capabilities compared to 2010.	High	- The RAN is deployed using container technology which offering us a low deployment time as it is captured in the following measured KPIs. All the values are available in D7.3 o *CRAN INSTANTIATION TIME* o *RRH INSTANTIATION TIME*

			- In D6.1, the resource allocation optimization e.g. vcpu, memory, core pinning, etc for SLA compliance. The different measurement are available in D6.1. - As MEC solution is deployed in the EDGE where the Cloud RAN components are instantiated, the latency is reduced. All the values are available in D7.3 o *CORE_RT_LATENCY* o *EDGE_RT_LATENCY* o *EXTRA_CORE_RT_LATENCY*
P2	Reducing the average service creation time cycle from 90 hours to 90 minutes.	High	- Service design time is reduced thanks to the usage of RDCL3D by taking advantage of both the intuitiveness of graphical based design and the expressiveness of the textual network service descriptors. - OPEX reduction since the descriptors are reusable at different times or in different locations - The network management time is reduced through the usage of SDN. To show this, the following KPIs are measured: o *TIME_MANAGE_NETWORK_WITHOUT_SDN*: o *TIME_MANAGE_NETWORK_WITH_SDN*: o *TIME_SAVED_MANAGING_NETWORK_WITH_SDN*: o *PERCENTAGE_TIME_SAVED*: - Automatic verification of service deployments before instantiation via static analysis to reduce the need for manual checks. Symbolic execution is a key enabler for this. - Automated characterisation of workloads using TALE, automated optimisation of workload on-boarding and management using KPI Mapping/Fingerprinting/Block Abstractions - Adoption of light-weight virtualisation approaches such as unikernels or

			container-based approaches to significantly reduce service instantiation times.
P3	Facilitating very dense deployments of wireless communication links to connect over 7 trillion wireless devices serving over 7 billion people.	Medium	The scaling feature is the Key technical feature to support device density. The scaling of cloud services is measured through two KPIs: - *NEW_INSTANCE_PROVISIONING_TIME* - *SCALING_REACTION_TIME* These KPIs are measured for the Cloud RAN and the video service. The service is optimized in the EDGE which is expressed through the following KPIs (refer to D7.3 for a complete measurement report): - *AVERAGE_DOWNLOAD_SPEED_WITH NETSCALER* - *AVERAGE_DOWNLOAD_SPEED_WITHOUT_NET SCALER* - *THROUGHPUT_WITHOUT_NETSCALER* - *THROUGHPUT_WITH_NETSCALER*
P4	Creating a secure, reliable and dependable Internet with a "zero perceived" downtime for services provision.	Medium	- The traffic loss and the Reaction time is measured to demonstrate the 'zero perceived' feature. In scene 4.c, we show that the time interval between the detection of an attack by the OPP and the time in which OpenVIM starts the instantiation of the Unikernel machines is in the order of milliseconds. - Use of symbolic execution to ensure robustness of the network, to check configurations and services before they are applied, and to ensure security by design, without sandboxing.

3.5.2 Societal KPIs

Table 2: Societal KPI

KPI		Relevance (High / Medium / Low / N.A.)	Details on planned project contribution towards achieving the KPI
S1	Enabling advanced User controlled privacy;	N.A.	
S2	Reduction of energy consumption per service up to 90% (as compared to 2010);	Medium	Reducing the Energy consumption in NFV data centres by improving infrastructure utilisation through more intelligent placement of workloads and continuous optimisation of placement decisions. Fast Service migration and shift or processing from one hardware to the other enables energy efficient scheduling (and having most of the platform in low energy consumption mode). Use of offloading to dedicated and more energy efficient hardware. For example, offloading OVS functionalities to NICs reduces the CPU utilization dramatically, and thus the energy consumption. More in general, cloud networking helps to adapt resources allocation, increasing efficiency by matching the workload characterisation to optimised allocation of resources through resource aware intelligent orchestration.
S3	European availability of a competitive industrial offer for 5G systems and technologies;	High	Consortium partners are an early adopter of SUPERFLUIDITY's innovation. Interface definition and contribution to standards. Working with the OpenStack and Kubernetes community to improve its functionalities as an NFV infrastructure manager and bridging the gaps with the ETSI NFV requirements. Contributions to industry initiatives such as OPNFV to accelerate the adoption and

			deployment of open platform solutions to drive the adoption of NFV. Creation of high fidelity prototypes to demonstrate feasibility of the approach developed by SUPERFLUIDITY.
S4	Stimulation of new economically-viable services of high societal value like U-HDTV and M2M applications;	Medium	Front-End cloud and Edge cloud concept facilitates delay-sensitive services and crowd-targeted video services (e.g. multi-camera video streaming apps of events). Easing the development of M2M markets via virtualised infrastructure.
S5	Establishment and availability of 5G skills development curricula (in partnership with the EIT).	N.A.	

3.5.3 Business-related KPIs

Table 3: Business-related KPIs

KPI		**Relevance** (High / Medium / Low / N.A.)	**Details on planned project contribution towards achieving the KPI**
B1	Leverage effect of EU research and innovation funding in terms of private investment in R&D for 5G systems in the order of 5 to 10 times;	Medium	Exploitation of SUPERFLUIDITY results and prototypes by industrial partners into internal R&D programs. This is illustrated by the exploitation plan of SUPERFLUIDITY partners Contribution to the open source community.
B2	Target SME participation under this initiative commensurate with an allocation of 20% of the total public funding;	Medium	Budget allocated to SMEs=€ 1.206.375, corresponding to 15,28% of the total public funding.
B3	Reach a global market share for 5G equipment & services delivered by European headquartered ICT companies at, or above, the reported 2011 level of 43% global market	High	Use of SUPERFLUIDITY collateral with customers to demonstrate feasibility of 5G use cases and solution paths to major 5G technical challenges. Investments by industrial partners.

	share in communication infrastructure.		

4 Work performed

4.1 Project achievements list

In this sub-section we provide a list of key project achievements, first by listing them an then providing a short summary of each one.

1. Selection and detailed description of use cases / scenarios for the integrated project testbed[1]
2. Setup of two interconnected testbeds, elaboration of an integration plan, components deployment and integration[2], integration and validation results[3]. Final demonstrator up and running.
3. Kuryr: enabling neutron-networking ecosystem for containers[4], regardless they are running on baremetal servers or inside OpenStack VMs. The work done avoid double encapsulation for the nested case[5], as well as speeds up the containers boot up in Neutron networks[6].
4. RDCL 3D tool for the design of NFV based services on heterogeneous infrastructure. The tool supports the modelling and design of: 1) services with nested RFBs like regular VMs and ClickOS Unikernels, respectively deployed on a traditional NFV virtualization infrastructure and on a XEN platform supporting ClickOS Unikernels; 2) services with mixed VMs and containers; 3) softwarized C-RAN configurations.

4.1. The OSM GUI for the OSM lightweight build [7] is based on the RDCL 3D tool. It is part of OSM Release Four to be released in May 2018.

5. Integration of decomposed C-RAN solution prototyped with RDCL 3D tool[8]
6. Unikernel technology: LightVM[9] re-architects the Xen virtualization system and uses unikernels to achieve VM boot times of a few milliseconds for up to 8,000 guests (faster than containers).
7. Unikernel orchestration[10]: OpenVIM extensions to support Unikernels. Performance evaluation of different Virtual Infrastructure Managers extensions (OpenVIM, OpenStack and Nomad) to support Unikernel orchestration.

7.1. OpenVIM extensions for Unikernels merged in the mainstream OpenVIM.[11]

---

[1] Deliverable D7.1
[2] Deliverable D7.2
[3] Deliverable D7.3
[4] Several blog posts about kuryr features/installation/usage are available at https://ltomasbo.wordpress.com/
[5] https://review.openstack.org/#/c/361993/, https://review.openstack.org/#/c/402462/
[6] https://blueprints.launchpad.net/kuryr-kubernetes/+spec/ports-pool
[7] https://osm.etsi.org/gitweb/?p=osm/LW-UI.git;a=summary
[8] Eucnc 2017 demo: "**Superfluid Orchestration of heterogeneous RFBs (Reusable Functional Blocks) for 5G net**"
[9] Felipe Huici et al. NEC: **"My VM is Faster (and Safer) than your Container"**, SOSP 2017.
[10] P.L. Ventre, C. Pisa, S. Salsano, G. Siracusano, F. Schmidtz, P. Lungaroni, N. Blefari-Melazzi: **"Performance Evaluation and Tuning of Virtual Infrastructure Managers for (Micro) Virtual Network Functions"**, IEEE NFV-SDN 2016, 7-9 November 2016, Palo Alto, California, USA.
[11] https://osm.etsi.org/gerrit/5850

8. Intent based reprogrammable fronthaul network infrastructure[12]
9. Algorithms and tools for flow/packet processing function allocation to processors in Fastclick[13]
10. Design of a Packet Manipulation Processor (PMP) based on a RISC architecture for extended Data Plane Programmability[14,15]
11. Dissemination and demo of Citrix Hammer, the Traffic Generator that was used in WP4 and WP7 activities [16,17]
12. A novel approach to resource allocation in 5G networks data plane that extends the packet switching principle to CPUs.[18]
13. OSM adoption and integration in the MEC environment as a Management and Orchestration (MEO) solution for applications Life Cycle Management (instantiation and disposal)
14. Platform-aware C-RAN baseband RFB adaptation and optimization using dynamic task clustering and scheduling approach [19] [20]
15. Scalable and predictable 5G baseband architecture employing dynamic task scheduling and guaranteed service inter-task connection allocation [21, 22]
16. Evaluation of cost of software switching for NFV[23]
17. MDP (Markov Decision Process)-Optimized VM scaling and load balancing mechanism for NFV[24]
18. Automating workload fingerprinting, trying to identify the hardware subsystems on a compute node that are most significantly affected by the deployment of a workload.[25]

---

[12] Eucnc 2017 demo: "**Superfluid Orchestration of heterogeneous RFBs (Reusable Functional Blocks) for 5G net**"
[13] Deliverable I5.1b
[14] S. Pontarelli, M. Bonola, G. Bianchi: **"Smashing SDN "built-in" actions: programmable data plane packet manipulation in hardware"**, IEEE NetSoft 2017, Bologna, Italy, July 3-7, 2017
[15] M. Bonola, R. Bifulco, L. Petrucci, S. Pontarelli, A. Tulumello, G. Bianchi: **"Implementing advanced network functions for datacenters with stateful programmable data planes"**, IEEE LANMAN 2017 – The 23rd IEEE International Symposium on Local and Metropolitan Area Networks, Osaka, Japan, June 12-14, 2017
[16] I. Prevezanos, A. Angelou, C. Tselios, A. Stergiakis, V. Tsogkas and G. Tsolis: **"Hammer: A Real-world, end-to-end Network Traffic Simulator"**, best paper award @ IEEE CAMAD 2017, Lund, Sweden, 19-21 June 2017.
[17] I. Prevezanos, C. Tselios, A. Angelou, M. McGrath, R. Mekuria, V. Tsogkas and G. Tsolis: **"Evaluating Hammer Network Traffic Simulator: System Benchmarking and Testbed Integration"**, IEEE GLOBECOM 2017 – Global Hub: Connecting East and West, Singapore, 4-8 December 2017.
[18] L. Vasilescu, V. Olteanu, C. Raiciu: **"Sharing CPUs via endpoint congestion control"**, IEEE SIGCOMM KBNets 2017, Los Angeles, US-CA, August 21-25, 2017
[19] E. Matus: "**Flexible Signal and Data Processing Platforms for Wireless Communications**", IEEE 5G Dresden Summit, September 2016, Dresden
[20] E. Matus, "**Design And Optimization of 5G Signal Processing System**", IEEE 5G Summit Dresden, 2017
[21] Y. Chen, E. Matus and G. Fettweis, "**High Performance Dynamic Resource Allocation for Guaranteed Service in Network-on-Chips**", IEEE Transactions on Emerging Topics in Computing (TETC), 2017
[22] G. Fettweis and E. Matus, "**Scalable 5G MPSoC Architecture**", Asilomar Conference on Signals, Systems, and Computers (ACSSC), Pacific Grove, 2017
[23] Deliverable D4.1
[24] M. Shifrin, E. Biton, and O. Gurewitz. "**Optimal control of VNF deployment and scheduling**". IEEE International Conference on the Science of Electrical Engineering (ICSEE), 2016.
[25] Deliverable D4.1

19. A novel automated approach for the implementation of block abstraction models was developed.[26]
20. Debugging P4 Programs with Vera[27]
21. Resource allocation for network functions in Click-based environments: a joint characterization, modeling and optimisation approach based on machine learning.[28]

[26] Deliverable D4.1

[27] Radu Stoenescu, Dragos Dumitrescu, Matei Popovici, Lorina Negreanu and Costin Raiciu, "**Debugging P4 Programs with Vera**", to appear in SIGCOMM 2018

[28] Deliverable D5.1

4.2 Project achievements details

In this sub-section we provide a short summary for each of the of key project achievements.

1. Selection and detailed description of use cases / scenarios for the integrated project testbed[29]
The project has selected 12 use cases to demonstrate most of the technology developed in the project, matched to the planned contributions of partners according to the DoA. In deliverable D7.1, we have shown the relation of the use cases with the technological blocks developed by the project. Then we have identified the "scenes" within a unified demonstration story, showing how the components needed to implement the use cases map into the integrated project demonstrator.
2. Setup of two interconnected testbeds, elaboration of an integration plan, components deployment and integration[30]
The integrated Superfluidity system is based on two interconnected testbed, one in BT premises representing a central cloud infrastructure and another one in Nokia FR premises, hosting the MEC/Edge Cloud/C-RAN infrastructure. The two testbeds are interconnected by means of a VPN, providing a single logical networking infrastructure. Several services and components, implemented with different technologies and having different requirements, needed to be deployed and integrated in this distributed environment, making the integration a challenge. This work (documented in the deliverable D7.2) was the basis for the subsequent work on the project unified prototype validation and assessment. A total of 28 components were integrated and run together, showing the 14 selected use cases /scenes.
2.1 Final integrated system up and running. Validation and assessment of the integrated system[31].
The project has followed the ambitious approach of integrating as many components as possible into the integrated Superfluidity system. We selected and implemented a sequence of use cases/scenes, conceived to mimic the lifecycle of a real network, from the initial planning, design and configuration stages, to network and service deployment and operation, including multiple alternative solutions, to assess optimal performance. The validation and assessment of this unified prototype has been reported in the deliverable D7.3.
3. Kuryr: enabling neutron-networking ecosystem for containers[32], regardless they are running on baremetal servers or inside OpenStack VMs. The work done avoid double encapsulation for the nested case[33], as well as speeds up the containers boot up in Neutron networks[34].
Kuryr is an OpenStack project that targets to bring OpenStack networking capabilities to containers. The idea behind Kuryr, is to be able to leverage the abstraction and all the hard work that was put in Neutron and its plugins and services and use that to provide production grade networking for containers use cases. In Superfluidity, we have contributed to Kuryr project by: (1) Enabling nested containers, i.e., allowing to create containers inside OpenStack VMs by making use of the VLAN-Aware-VMs Neutron capability. This enables containers to be in different networks than

[29] Deliverable D7.1
[30] Deliverable D7.2
[31] Deliverable D7.3
[32] Several blog posts about kuryr features/installation/usage are available at https://ltomasbo.wordpress.com/
[33] https://review.openstack.org/#/c/361993/, https://review.openstack.org/#/c/402462/
[34] https://blueprints.launchpad.net/kuryr-kubernetes/+spec/ports-pool

their containing VM, providing not only Neutron networking features, but also an extra security layer, as well as avoiding double encapsulation issues when running Kubernetes cluster on top of OpenStack VMs. (2) Adding pools of Neutron resources, in this case, Neutron ports, to speed up containers boot up time when using OpenStack networking. This approach minimizes the interaction with Neutron during containers boot up, as well as make a more efficient use of Neutron REST API by performing bulk requests instead of single ones. (3) Kuryr integration with Neutron load balancers, making both LBaaSv2 and Octavia load balancers available for containers, even being able to load balance between VMs and containers Note that these contributions are already available upstream and the community is already making use of it. It has been even integrated with other SDN solutions, such as DragronFlow and OpenDaylight.
4. RDCL 3D tool for the design of NFV based services on heterogeneous infrastructure. The tool supports the modelling and design of: 1) services with nested RFBs like regular VMs and ClickOS Unikernels, respectively deployed on a traditional NFV virtualization infrastructure and on a XEN platform supporting ClickOS Unikernels; 2) services with mixed VMs and containers; 3) softwarized C-RAN configurations.
The RDCL 3D tool is a web framework that can be adapted to support the modelling and design of different types of NFV services and the interaction with different orchestrators. It is modular and it is designed to facilitate the introduction and the support of different models. The tool has been released as Open Source under the Apache 2.0 license.
4.1 The OSM GUI for the OSM lightweight build [35] is based on the RDCL 3D tool. It is part of OSM Release Four to be released in May 2018.
The project has worked to improve the OSM orchestrator with a new web GUI. The work has been performed in close cooperation with the TID team leading the OSM development in the context of the ETSI OSM community. Thanks to the flexibility of RDCL 3D tool, it has been relatively easy to support the functionality of OSM lightweight, providing a powerful GUI for accessing the OSM functionality.
5. Integration of decomposed C-RAN solution prototyped with RDCL 3D tool[36]
RDCL3D allows a high level of abstraction of any network or a service to be deployed on a heterogeneous cloud infrastructure. The network or the service will be described via a graph where the different RFBs are connected. Once the graph is built, an RFB descriptor file available in YAML and JSON format is generated. In this way, RDCL 3D enables Infrastructure as Code. For the Cloud-RAN, three components are defined: RRH, BBU, and a fronthaul. The C-RAN will be connected to a core EPC. After building the graph, a button's click allows us to trigger the deployment of the designed network. Behind the scene, an orchestrator takes the generated file, translates it into another file or language that will be used by the actual controller to run the deployment. For instance, if we want to deploy C-RAN components using Kubernetes, then the RFB descriptor file would be translated into Kubernetes yaml file. The first thing that we deploy is a set of Open vSwitch bridges that form the fronthaul network spanning multiple host. Next, the EPC docker container starts, followed by the RRH and finally the BBU.

[35] https://osm.etsi.org/gitweb/?p=osm/LW-UI.git;a=summary

[36] Eucnc 2017 demo: "**Superfluid Orchestration of heterogeneous RFBs (Reusable Functional Blocks) for 5G net**"

6. Unikernel technology: LightVM[37] re-architects the Xen virtualization system and uses unikernels to achieve VM boot times of a few milliseconds for up to 8,000 guests (faster than containers).
Containers are in great demand because they are light- weight when compared to virtual machines. On the down- side, containers offer weaker isolation than VMs, to the point where people run containers in virtual machines to achieve proper isolation. In this work, we examine whether there is indeed a strict trade-off between isolation (VMs) and efficiency (containers). We state that VMs can be as nimble as containers, as long as they are small and the toolstack is fast enough. We achieve lightweight VMs by using unikernels for specialized applications and with Tinyx, a tool that enables creating tailor-made, trimmed-down Linux virtual machines. By themselves, lightweight virtual machines are not enough to ensure good performance since the virtualization control plane (the toolstack) becomes the performance bottleneck. We present LightVM, a new virtualization solution based on Xen that is optimized to offer fast boot-times regardless of the number of active VMs. LightVM features a complete redesign of Xen's control plane, transforming its centralized operation to a distributed one where interactions with the hypervisor are reduced to a minimum. LightVM can boot a VM in 2.3ms which comparable to fork/exec on Linux (1ms), and two orders of magnitude faster than Docker.
7. Unikernel orchestration[38]: OpenVIM extensions to support Unikernels. Performance evaluation of different Virtual Infrastructure Managers extensions (OpenVIM, OpenStack and Nomad) to support Unikernel orchestration.
We have adapted and optimized three VIMs (Virtual Infrastructure Managers) to support the ClickOS Unikernel, namely OpenVIM, OpenStack and Nomad. In particular, we have extended OpenVIM to support the ClickOS Unikernel (and more in general the XEN virtualization) in a backward compatible way. We have also prototyped the support of Unikernels in OpenStack and Nomad and provided performance measurements of orchestration time for the three considered platforms.
7.1 OpenVIM extensions for Unikernels merged in the mainstream OpenVIM.[39]
Our work has been presented to the OSM community that is maintaining OpenVIM. We received feedback and reacted accordingly. Then we have submitted a revised code contribution that has been merged upstream in April 2018.
8. Intent based reprogrammable fronthaul network infrastructure[40]
The fronthaul network offers the connectivity between the RRH or the Front-End Unit and the EDGE cloud. The process running in each location will depend on the type of split we apply which are discussed in D4.3. To connect the antenna site (RRH) to the EDGE cloud, a CPRI link is normally used. Our objective was to build a fully re-programmable fronthaul solution, which can select the routing path according to the requirement of the flow in terms of latency and throughput following a declarative model. For that purpose, an Ethernet based solution was developed using an overlay network composed of Open Virtual Switches to deliver the required flexibility. Using a SDN controller (in our case ONOS), and the

[37] Felipe Manco, NEC: **"NoXS: Death to the XenStore"**, talk at Xen Project Developer and Design Summit, Budapest, Hungary, July 11-14, 2017

[38] P.L. Ventre, C. Pisa, S. Salsano, G. Siracusano, F. Schmidtz, P. Lungaroni, N. Blefari-Melazzi: **"Performance Evaluation and Tuning of Virtual Infrastructure Managers for (Micro) Virtual Network Functions"**, IEEE NFV-SDN 2016, 7-9 November 2016, Palo Alto, California, USA.

[39] https://osm.etsi.org/gerrit/5850

[40] Eucnc 2017 demo: "**Superfluid Orchestration of heterogeneous RFBs (Reusable Functional Blocks) for 5G net**"

intent programming, we succeed in connecting the RRH and the EDGE cloud in a software manner and making it fully re-programmable. The orchestrator instructs an ONOS controller to install host-to-host intents to create paths between the BBU and RRH, and between the BBU and EPC. The BBU initiates communication with the RRH and EPC, and once all is set up, a UE can be attached.
9. Algorithms and tools for flow/packet processing function allocation to processors in Fastclick[41]
FastClick allows a higher level of abstraction than Click, thanks to its automatic resource allocator. The user only needs to specify the logic of the network function, and FastClick will automatically handle low-level details such as core allocation. FastClick has also been extended with support for flow processing in addition to packet processing. This extension supports writing flow-aware middleboxes in a simplified manner, such as an HTTP reverse proxy, directly inside FastClick. We also improved service chaining efficiency by sharing classification work, which is now done only once (using hardware offloading when possible), even if several components in the chain need classification. Finally, using a machine-learning-based approach, we can now predict the performance (e.g. throughput) of some network functions based on information about the workload and allocated resources. This work will be used to guide resource allocation and function placement towards optimality.
10. Design of a Packet Manipulation Processor (PMP) based on a RISC architecture for extended Data Plane Programmability[42,43]
Programmable dataplanes are emerging as a disruptive technology to implement network function virtualization in an SDN environment. Starting from the original OpenFlow's match/action abstraction, most of the work has so far focused on key improvements in matching flexibility. Conversely, the "action" part, i.e. the set of operations (such as encapsulation or header manipulation) performed on packets after the forwarding decision, has received way less attention. With PMP,we move beyond the idea of "atomic", pre-implemented, actions, and aim at having programmable dataplane actions while retaining high speed multi-gbps operation. The Packet Manipulation Processor (PMP) is a domain-specific HW architecture, able to efficiently support micro-programs implementing such actions.
11. Dissemination and demo of Citrix Hammer, the Traffic Generator that was used in WP4 and WP7 activities [44,45]
Hammer is a real world, end-to-end network traffic simulator, capable of simulating complex and dynamic network, user and server behaviours. The focus of this tool is to primarily facilitate investigations related to product stability, for instance different aspects of capacity, longevity, memory leaks, cores and to handle customer content testing that will reveal the behaviour of the device under test in realistic network conditions. Hammer has a modular design, which offers excellent scalability, rendering the platform

[41] Deliverable I5.1b

[42] S. Pontarelli, M. Bonola, G. Bianchi: **"Smashing SDN "built-in" actions: programmable data plane packet manipulation in hardware"**, IEEE NetSoft 2017, Bologna, Italy, July 3-7, 2017

[43] M. Bonola, R. Bifulco, L. Petrucci, S. Pontarelli, A. Tulumello, G. Bianchi: **"Implementing advanced network functions for datacenters with stateful programmable data planes"**, IEEE LANMAN 2017 – The 23rd IEEE International Symposium on Local and Metropolitan Area Networks, Osaka, Japan, June 12-14, 2017

[44] I. Prevezanos, A. Angelou, C. Tselios, A. Stergiakis, V. Tsogkas and G. Tsolis: **"Hammer: A Real-world, end-to-end Network Traffic Simulator"**, best paper award @ IEEE CAMAD 2017, Lund, Sweden, 19-21 June 2017.

[45] I. Prevezanos, C. Tselios, A. Angelou, M. McGrath, R. Mekuria, V. Tsogkas and G. Tsolis: **"Evaluating Hammer Network Traffic Simulator: System Benchmarking and Testbed Integration"**, IEEE GLOBECOM 2017 – Global Hub: Connecting East and West, Singapore, 4-8 December 2017.

capable of being installed on commodity hardware. In addition, it is designed in a resource-savvy manner, thus requires limited computational resources to generate significant traffic loads. Instead of operating on a packet level, Hammer offers application-layer workload interaction along with inherent data plane acceleration, delivering brisk performance with unparalleled flexibility and ease of use. Our tests show that Hammer's operation is linearly linked to the underlying hardware resources, however, even when the simulator is installed in a resource-bound environment, it can still deliver traffic loads that correspond to thousands of interconnected users each with real-world behaviour per session. When installed in cutting-edge contemporary servers with the latest generation CPUs, Hammer performs significantly better indicating the software's suitability for high-demanding tasks. Under the auspices of the project, we have partially presented and benchmarked Hammer, while demonstrating some of its capabilities on testing actual platforms deployed in production-like networking environments.
12. A novel approach to resource allocation in 5G networks data plane that aims to extend the packet switching principle to CPUs.[46]
Software network processing in 5G relies on dedicated cores and hardware isolation to ensure appropriate throughput guarantees. Such isolation comes at the expense of low utilization in the average case, and severely restricts the number of network processing functions one can execute on a server. We propose that multiple processing functions should simply share a CPU core, turning the CPU into a special type of "link". We use multiple NIC receive queues and the FastClick suite to test the feasibility of this approach. We find that, as expected, per core throughput decreases when more processes are contending; however the decrease is not dramatic: around 10% drop with 10 processes and 50% in the worst case where the processing is very cheap (bridging). We also find that the processor is not shared fairly when the different functions have different per packet costs. Finally, we implement and test in simulation a solution that enables efficient CPU sharing by sending congestion signals proportional to per-packet cost for each flow. This enables endpoint congestion control (e.g. TCP) to react appropriately and share the CPU fairly.
13. OSM adoption and integration in the MEC environment as a Management and Orchestration (MEO) solution for applications Life Cycle Management (instantiation and disposal)
The Open Source MANO (OSM) is an ETSI-hosted project, created to develop an open source NFV Management and Orchestration (MANO) software stack aligned with ETSI NFV. As the MANO NFV functions are significantly similar to the MEO (Mobile/Multi-access Edge Orchestrator), a MEC component, after some evaluation work of different MANO solutions, we decided to use OSM. MEO is responsible for managing and orchestrating ME Applications (ME Apps) which are hosted at the Edge (ME Host). Management features include the lifecycle operations such as instantiation, scaling, and termination of ME Apps in a ME Hosts. The orchestration functionality has a global view of all ME Hosts and is able to take decisions about where a particular ME App should run, or about the migration of a particular ME App between two different ME Hosts, e.g. triggered by an end-user movement. It is important to understand that MEO, implemented using OSM, is part of the MEC solution and devoted to manage MEC Apps. Previously the MEC solution and other VNFs (e.g. C-RAN) needed to be deployed so that they can start this operation. For this purpose, the project uses ManageIQ.

[46] L. Vasilescu, V. Olteanu, C. Raiciu: **"Sharing CPUs via endpoint congestion control"**, IEEE SIGCOMM KBNets 2017, Los Angeles, US-CA, August 21-25, 2017

14. Platform-aware C-RAN baseband RFB adaptation and optimization using dynamic task clustering and scheduling approach [47] [48]	
Typically, implementation of communication signal processing algorithms employs static approaches for application definition, mapping and scheduling. In Superfluidity. We developed dataflow runtime system and reconfigurable scalable baseband C-RAN application that enables dynamic application generation (TTI specific task graphs), dynamic graph transformation using task clustering and dynamic mapping to underlying hardware. Dynamic generation-transformation-mapping approach allows for runtime application structure optimisation and adaptation according to hardware platform and performance requirements, which make it suitable for flexible deployment on various HW platforms.	
15. Scalable and predictable 5G baseband architecture employing dynamic task scheduling and guaranteed service inter-task connection allocation [49, 50]	
The huge diversity of 5G application requirements and associated modem protocols impose high demands on radio platforms in terms of scalable latency, reliability, and computation performance. Within the Superfluidity project, we scope a scalable multi-processor platforms that can handle a plurality of parallel sliced wireless links. The challenge in this regards is to efficiently allocate and use shared computation resources according to workload requirements of particular slice, and to minimize inter-task congestion by provisioning adequate communication resources. We propose dedicated unit core-manager to dynamically map, schedule and prioritize the application tasks to computation resources. Moreover, we propose linear complexity trellis-search path algorithm and associated network-manager allowing to dynamically find an optimum network path between communicating tasks to provide guaranteed service connection in terms of throughput and latency.	
16. Evaluation of cost of software switching for NFV[51]	
An important enabler for the NFV paradigm is software switching, which should satisfy rigid network requirements such as high throughput and low latency. However, software switching comes with an extra cost in terms of computing resources that are allocated specifically to software switching in order to steer the traffic through running services (in addition to computing resources required by VNFs). This cost depends primarily on the way the VNFs are internally chained, packet processing requirements, and accelerating technologies (e.g., packet acceleration such as Intel DPDK). The use of Fast Dataplane technology in network interface cards as an alternative to SR-IOV and DPDK was evaluated. The cost of software switching (in terms of consumed resources) and its expected performance are key inputs for Superfluidity orchestration and particularly placement algorithms. The task has evaluated the performance of software switching using different configurations. There was a particular focus on two key acceleration technologies, namely, OVS DPDK and FD.io VPP. OVS DPDK benchmarked and its performance and resource usage compared against standard OVS as well as with SR-IOV. The task also evaluated the emerging Fast Dataplane input/output (FD.io) switch approach. The performance of FD.io was compared against the performance of an OVS-DPDK enabled soft switch.	

[47] E. Matus: "**Flexible Signal and Data Processing Platforms for Wireless Communications**", IEEE 5G Dresden Summit, September 2016, Dresden
[48] E. Matus, "**Design And Optimization of 5G Signal Processing System**", IEEE 5G Summit Dresden, 2017
[49] Y. Chen, E. Matus and G. Fettweis, "**High Performance Dynamic Resource Allocation for Guaranteed Service in Network-on-Chips**", IEEE Transactions on Emerging Topics in Computing (TETC), 2017
[50] G. Fettweis and E. Matus, "**Scalable 5G MPSoC Architecture**", Asilomar Conference on Signals, Systems, and Computers (ACSSC), Pacific Grove, 2017
[51] Deliverable D4.1

17. MDP (Markov Decision Process)-Optimized VM scaling and load balancing mechanism for NFV[52]
Dynamically adjusting the amount of resources allocated to a certain VNF based on the demand (performance) is a highly important problem. We devised a decision mechanism that dynamically increases (via scale out operation) or decreases (scale in operation) the number of VMs that are allocated to the VNF, according to the demand from the network service (e.g., number of flows a firewall handles). Our scaling decision mechanism minimizes the consumed resources, while maintaining the application required SLA. In conjunction with the scaling decisions, our dynamic mechanism also handles load balancing, steering the traffic flows to the different VMs and balancing the load between them. In this study, we tackle both the scaling decision and the load balancing strategy as a single problem, formulated as a Markov Decision Process.
18. Automating workload fingerprinting, trying to identify the hardware subsystems on a compute node that are most significantly affected by the deployment of a workload. [53]
A characterisation approach developed was based on the automated identification of platform metrics that most significant influence the behaviour of service KPI's. In addition a profiling approach based on automated workload fingerprinting was developed which can identify the hardware subsystems on a compute node that are most significantly affected by the deployment of a workload. The approach developed has utility for informing workload placement decisions.
19. A novel automated approach for the implementation of block abstraction models was developed[54].
A novel automated approach for the implementation of block abstraction models was developed. The block abstraction model provides a logical representation of a workloads affinity for infrastructure resource allocations and features. The block abstraction model also encapsulated the effect of the deployment topology across a heterogeneous resource landscape on a workload's KPI performance. The block abstractions modelling approach was applied to enabling reasoning over performance versus cost deployment options. The approach developed enabled infrastructure cost reductions when favouring cost over performance for a test service
20. Debugging P4 Programs with Vera[55]
Truly programmable switches allow replacing the dataplane algorithm too, not just the rules it uses (like in Openflow). To ensure programmability and performance, 5G networks will adopt switches that support the P4 language, or other ones (e.g. OpenState from Univ. of Rome Tor Vergata). We present Vera, a tool that exhaustively verifies P4 programs using symbolic execution. Vera automatically uncovers a number of common bugs including parsing/deparsing errors, invalid memory accesses, loops and tunneling errors, among others. To enable scalable, exhaustive verification Vera automatically generates all valid header layouts, it uses symbolic table entries to simulate a variety of table rule snapshots and uses a novel data-structure for match-action processing optimized for verification. These techniques allow Vera to scale very well: it only takes between 5s-15s to track the execution of a purely symbolic packet in the largest P4 program currently available (6KLOC) and can compute SEFL model updates according to table insertions and deletions in milliseconds. We have used Vera to analyze all P4 programs we could find

[52] M. Shifrin, E. Biton, and O. Gurewitz. "**Optimal control of VNF deployment and scheduling**". IEEE International Conference on the Science of Electrical Engineering (ICSEE), 2016.
[53] Deliverable D4.1
[54] Deliverable D4.1
[55] Radu Stoenescu, Dragos Dumitrescu, Matei Popovici, Lorina Negreanu and Costin Raiciu, "**Debugging P4 Programs with Vera**", to appear in SIGCOMM 2018

including the P4 tutorials, P4 programs in the research literature and the switch code from https://p4.org. Vera has found several bugs in each of them in seconds.
21. Resource allocation for network functions in Click-based environments: a joint characterization, modeling and optimisation approach based on machine learning.[56]
We tackled the optimisation of resource allocation for network functions in Click-based environments. We devised a joint characterization, modeling and optimisation approach based on machine learning. The proposed approach is very general and can also be applied to other environments, such as the deployment of VMs, which was also promisingly investigated using this method. As this approach requires a network function performance dataset, we first devised a tool, the Network Performance Framework (NPF), to help with such experiments and we began the generation of an open dataset to foster further research into this topic, both in the networking and in the machine learning communities. While this research work is not fully completed, the achieved results are very promising.

[56] Deliverable D5.1

4.3 Work by Work-packages

In this sub-section, we report a more detailed description of the work performed in each WP.

4.3.1 WP2 Use Cases, System Requirements, and Functional Analysis

A "superfluid" network will have the ability to instantiate services on-the-fly, run them anywhere in the network (core, aggregation, edge) and shift them transparently to different locations. Such capabilities are a key part of the converged cloud-based 5G future - they will enable innovative use cases in the mobile edge, empower new business models and allow almost instant roll out of new services, and reduce investment and operational costs. Specifically, there are a large number of potential use cases for a superfluid network, covering themes such as wireless access, mobile edge computing and on-the-fly monitoring. By studying these use cases we can better understand both business requirements, such as service agility and cost savings, and technical requirements, for example quality of experience and scalability.

Whilst use cases are interesting in themselves, they also help guide the technical work of the project. We have not attempted a strict 'top down approach', where the architecture would be derived in a formal process from the requirements of the various use cases. Instead we are using an iterative approach where the requirements analysis takes into account our on-going architecture work. We include business requirements, such as service agility and cost savings, and technical requirements, for example quality of experience and scalability, as well as architectural concepts like reusable functional blocks.

To instantiate services on-the-fly, run them anywhere in the network and shift them transparently to different locations, SUPERFLUIDITY introduced the concept of Reusable Functional Block (RFB). This capability is a key part of the converged cloud-based 5G future. Specifically, during the first year of the project we have started to understand the benefits of the concept and identify its most important applications. More in depth, SUPERFLUIDITY proposes to decompose the architecture into elementary radio and network processing primitives and events, which can then be exploited as basic modules of more comprehensive (and traditionally monolithic) network functions and services. The decomposition of monolithic functions into RFBs permits a flexible placement, as well as the incorporation of adequate virtualisation techniques like Containers (e.g. Docker), UniKernels, and full Virtual Machines, by considering real-time constraints.

SUPERFLUIDITY has focused on a set of different contexts, such as: Cloud Radio Access Networks, Mobile Edge Computing platform, generic NFV environments, fixed networking equipment and packet processing state machines. In each of this context a preliminary functional analysis and decomposition into RFBs has been performed.

4.3.2 WP3 Architecture and Programming Interfaces Specification

The vision of the SUPERFLUIDITY project is to move from the current architectural approaches based on monolithic network components/entities and their interfaces, to an approach where network functions can be programmatically composed using "building blocks" and the deployment of these "building blocks" over the underlying infrastructure is highly dynamical, allowing a continuous real-time optimisation. The SUPERFLUIDITY architecture is based on the following main pillars: (i) standardisation convergence, (ii) Reusable Functional Blocks (RFBs), (iii) programmable and portable interfaces, (iv) a security framework.

Standardisation Convergence

The SUPERFLUIDITY project aims to design a unified, high performance and distributed cloud platform for radio and network functions support, as well as their migration. In our vision, CRAN, MEC and cloud technologies are integrated, by adopting an architectural paradigm able to create the glue that can unify heterogeneous equipment and processing into one dynamically optimised, superfluid, network. Figure 4 depicts a high-level view of the overall architectural framework that has been investigated during the first year of the project. The top layer of the Figure includes the different components involved (CRAN, MEC, virtual core and Data Centres (DC)), while in the bottom layer the different types of physical DCs are shown (namely Cell-site, Local, Regional and Central). This classification is somehow arbitrary and the infrastructure of different operators can be structured in different ways. The next layer down is a traditional Operational Support System (OSS), whose main goal is to deal with all the components in order to create services for end-users.

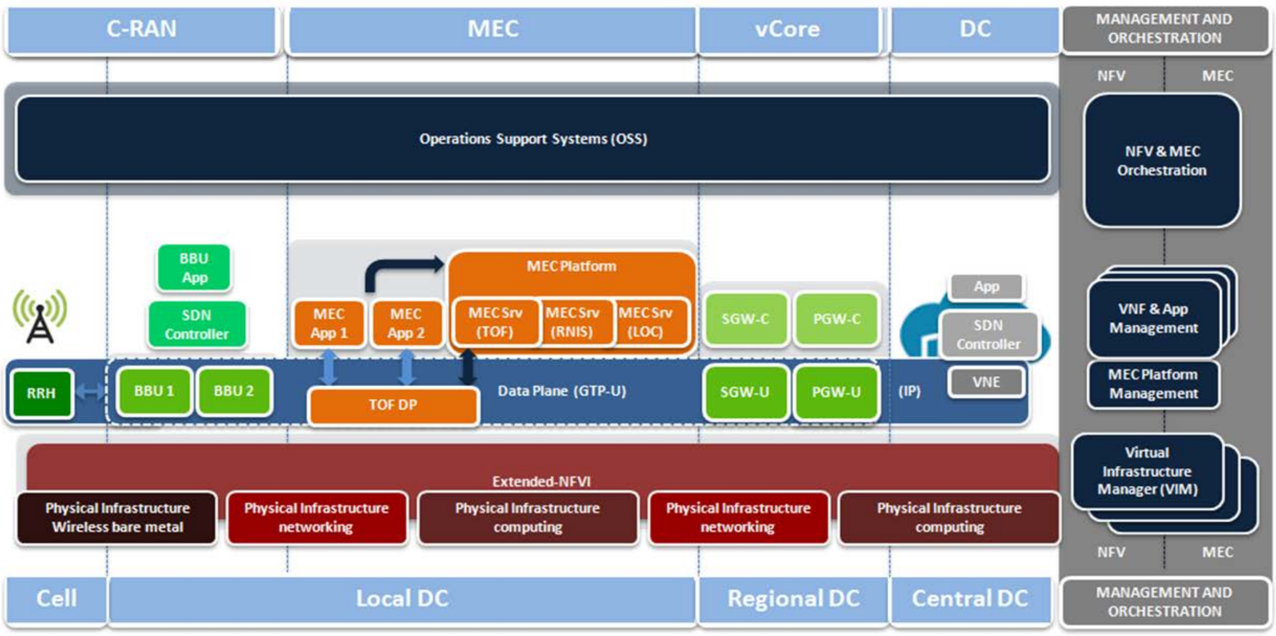


Figure 4: Architectural framework for SUPERFLUIDITY with an example mapping into the physical Data Centres

The underlying Extended-NFVI, located at the bottom of the proposed architecture, represents an evolution of the ETSI NFVI concept. The current NFVI focuses on supporting VMs or containers to run VNFs; the E-NFVI also considers heterogeneous execution environments (see I3.1 for more details). This extended-NFVI is common to all components, simplifies resource management and allows agile (superfluid) orchestration of services. The mapping of the components into the different physical DCs (Cell, Local, Regional) shown in Figure 4 and described hereafter is the one considered for the SUPERFLUIDITY testbed implementation. However, we stress that the dynamicity of the architecture, and the concept of an Extended-NFVI, allows support for different solutions, derived by considering the various trade-offs between performances and efficient utilisation of resources.

Starting from the left, the CRAN component is split into two blocks, corresponding to the Remote Radio Head (RRH) and the Base Band Unit (BBU) components. The RRH is placed in the cell-site, while the BBU is located in a specialised local DC, adapted for telecommunications. It is assumed that the local DCs will control a small number of cell-sites and that they are geographically distributed, but in general proximity to the cell sites. The BBU functionalities will be virtualised and thus they will benefit from the centralisation to scale in/out, according to the load and to be fully re-programmable under a holistic NFV/SDN controller. Figure 4 also shows the MEC components. In our architecture the MEC component is deployed in the same location (local DC) as the CRAN, given the fact that it utilises the exact same infrastructure with the latter or other VNFs. This endogenous characteristic of MEC significantly increases efficiency and ease of use. The next component shown in Figure 4 is the virtual Core (vCore), comprising the central nodes of a cutting-edge mobile network. The vCore runs on the common E-NFVI, usually located in regional DCs. In this way, both agility and fluidity of the overall architecture are improved, especially when live nodes need to be migrated and/or scaled. The Figure shows the expected evolution of mobile core networks towards the SDN model, where the control plane and data plane components are completely separated, with the former fully controlling the latter on data processing tasks. In particular, the components shown are the ones resulting from splitting the 4G/LTE Core elements. The DC component corresponds to the traditional datacentre segment, where a large number of services are deployed. These services are located at central points and deal with significant compute/storage/network resources. Beyond traditional services, central DCs also implement NFV and SDN technologies, in order to control the network components inside the DC. For this reason, a generic VNE (Virtual Network Element), a SDN controller and a generic App element are included in the architecture.

The last part of the architecture is the common management and orchestration vertical layer (right part of the Figure), which is responsible for providing the system intelligence. It includes a NFV-like set of functions, which supports an integrated view of the overall architecture, including networks, services and DCs. In this way, it is possible to take advantage of a common extended-NFVI and achieve an end-user centric view of the ecosystem. This layer is responsible for resource management over the different DCs illustrated at the bottom of Figure 4, thus building a federated environment. Moreover, it orchestrates VNFs to create complex services, taking the best decisions for services to be deployed, while considering customer needs.

Reusable Functional Blocks

The decomposition of high-level monolithic functions into reusable components is based on the concept of *Reusable Functional Blocks* (RFBs), which has been introduced during the first year of the project. An RFB is a logical entity that performs a set of functionality and has a set of logical input/output ports. In general, a Reusable Functional Block can hold state information, so that the processing of information coming in its logical ports can depend on such state information. RFBs can be composed in graphs to provide services or to form other RFBs (therefore a Reusable Functional Block can be composed of other RFBs). The RFB decomposition concept is applied to different heterogeneous environments.

In order to achieve the vision of superfluid composition, allocation and deployment of "building blocks" (RFBs), the SUPERFLUIDITY project focuses on the following pillars:

- *Decomposition of monolithic functions into Reusable Functional Blocks in different heterogeneous environments* - SUPERFLUIDITY targets to decompose: network services into basic functional abstractions; node-level functions into more granular packet processing primitives; heterogeneous hardware and system primitives into block abstractions.
- *Measurements based RFB operations* - This pillar consists in the definition and realisation of tools for real time monitoring of RFB performances and of the status of the environments (e.g. resource utilisation). The information provided by such tools can be used to drive the RFB operations (e.g. allocation of resources to RFBs).
- *Semantic specification of Block Abstractions* – A crucial aspect in composition-based approaches is to provide the "programmer" with a clear and unambiguous understanding of what each block is expected to do, and how it processes input data and produces an output.
- *High Performance Block Abstractions Implementation* - This pillar consists of deriving block abstractions for each of the underlying hardware components, and to optimise their performance.

Programmable and Portable Interfaces

RFBs needs to be characterised and described, and we need platform-agnostic node-level and network-level "programs" describing how the RFBs interact, communicate, and connect to each other so as to give rise to specific (macroscopic, and formerly monolithic) node components, network functions and services. A language that supports the description and the interaction of RFBs is referred to as RFB Description and Composition Language (RDCL). The heterogeneous computational and execution environments, supporting the execution (and deployment) of the RDCL scripts and the relevant coordination of the signal/radio/packet/flow/network processing primitives are referred to as RFB Execution Environments (REE).

The SUPERFLUIDITY architecture shown in Figure 5 describes the relation among the RFB, RDCL and REE concepts. The figure highlights that the model (with due technical differences) recursively apply at different levels.

Considering the architecture proposed in Figure 5, we identify the logical entities that are involved in the realisation of services. We consider that the architecture can recursively be applied to a number of levels, therefore we define an approach that can be applied at all levels. At each level, we identify a REE Manager and a REE User. The REE User requests the realisation/deployment/execution of a service/service component described using a RDCL script to the REE manager. The REE Manager is in charge of deploying/executing the RCDL script using the resources in its REE. Within a REE, the REE Manager interacts with REE Resources Entities that are needed to support the realisation of the RCDL script.

Hence, we can identify two main APIs. The first one is the API used by the REE User that wants to deploy a service or a component into an RFB Execution Environment. We refer to it as the User-Manager API (UM API). The second one is the API used by the REE Manager to interact with the resources in its REE. We refer to it as the Manager-Resource API (MR API).

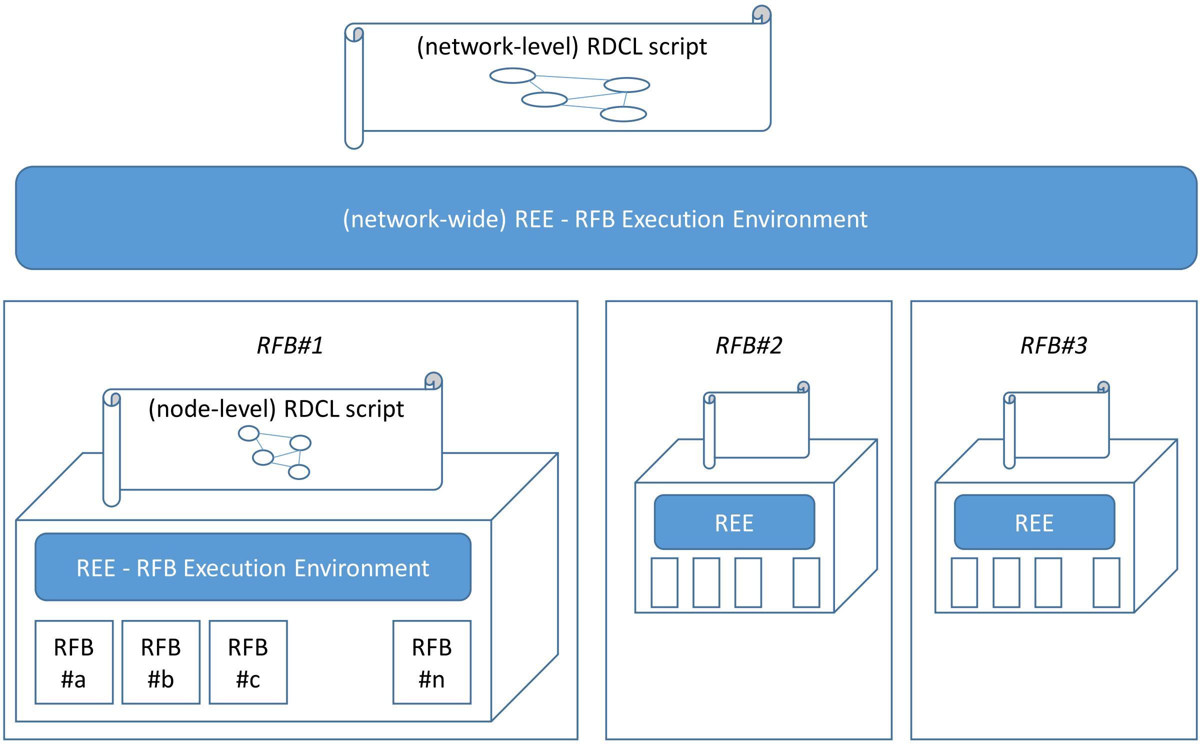


Figure 5: SUPERFLUIDITY conceptual architecture

Figure 6 illustrates these concepts, showing how the APIs can be mapped to different levels. Note that we refer to the APIs considering information exchange in both directions for each APIs.

The APIs for the different levels may use different modelling approaches, languages and tools, considering that SUPERFLUIDITY will not build them from scratch but will rely on existing work. Wherever possible, the project will try to relate the APIs at different level and proceed towards a harmonised vision.

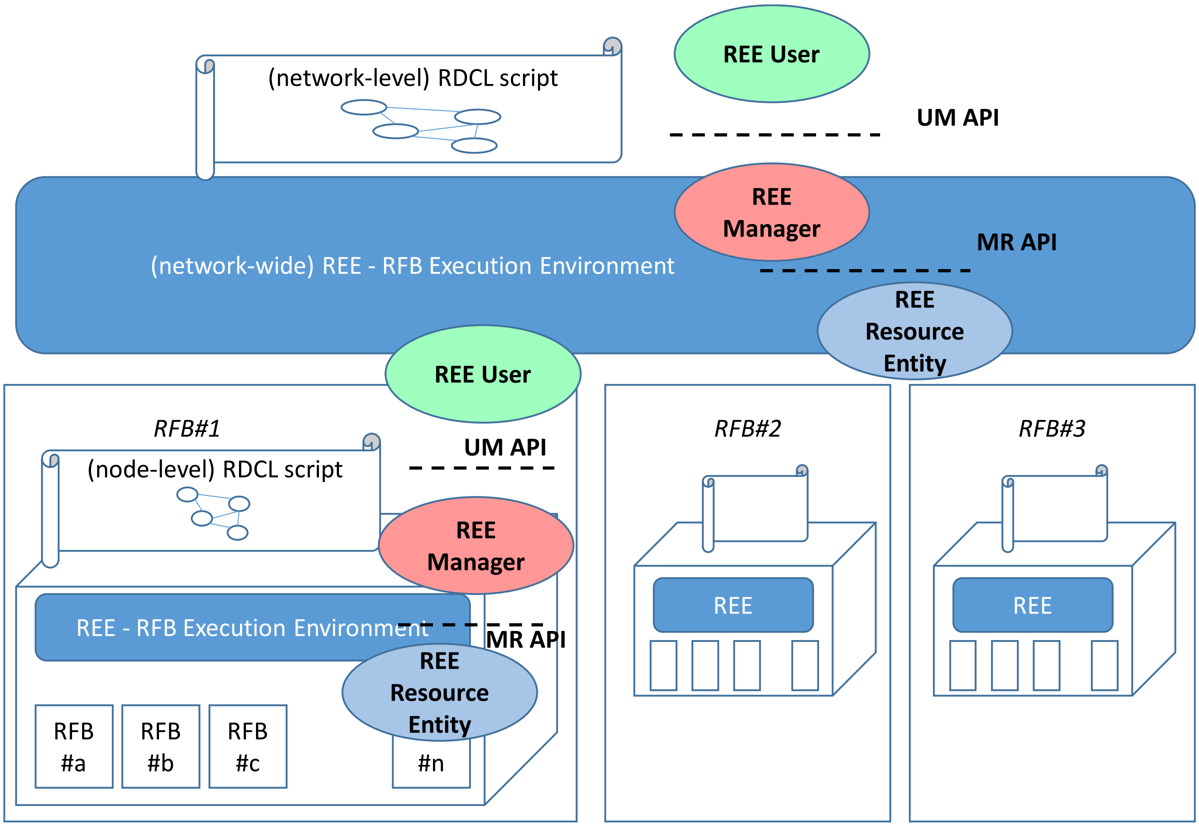


Figure 6: SUPERFLUIDITY Architecture APIs

Security Framework

Any architecture that relies on a real-time cloud infrastructure is susceptible to security hazards. This applies more to those relying on virtualisation of network functions. Accordingly, any such architecture is facing great challenges identifying and taking into account these security hazards. During the first year of the project we have started to study the security challenges and hazards that the SUPERFLUIDITY architecture need to address. Specifically, we have studied the issues concerning general cloud computing and virtualisation. Then, as a second step, we combine these two aspects and formulate the potential security challenges. Our preliminary results show that new security challenges and hence new security threats arise. These issues arise from to the migration of previously conventionally secluded application families to the virtual public environment. Moreover, pure security services can be deployed as isolated VNFs, providing services to other VNFs. Nonetheless, in some cases, the functional architecture should take into account and should be implemented in a symbiosis with accompanied security functions. Furthermore, we show that in different cases some security functionality should be integrated within the VNF design.

4.3.3 WP4 Heterogeneous Infrastructures and Abstractions

WP4 was focused on the infrastructure related elements of the Superfluidity system with a primary focus on the commodity hardware elements. The workpackage developed methodologies such as Throughput/Anomalies/Latency/Entropy (TALE) and testbed capabilities to support the modelling and profiling of general-purpose and specialised hardware and software resource types including accelerators, etc. In addition to the compute resource elements, the WP investigated the performance of I/O, network and acceleration sub-systems including caches in the context of the use cases identified in WP2 such as late stage transmuxing. Profiling in the context of workloads and service scenarios enable the infrastructure layer to expresses the available landscape in parameters that are meaningful to a reservation and provisioning decision at run-time. Approaches to expressing the key characteristics of resource elements were investigated such as the use of Block Abstractions (BA). A general-purpose data model called the Superfluidity Information Model (SIM) was developed. Specific workloads in Cloud RAN were investigated for innovation opportunity, such as means to express runtime specifications and performance targets, which would then boost the impact of the block abstraction information by allowing very fine-grained placement at runtime.

Objectives

WP4 has three key overall objectives:

1. Model and establish profiles, key metrics and a supporting data model for the landscape resource elements and key metrics for the workloads selected from the WP2 use cases.
2. Define and implement a block abstraction model including API to express the relevant parametric details of the infrastructure profile to the other layers in WP3, which can interpret and reason over the available resources.
3. Explore innovative approaches to key radio and network processing functions, targeting scalability, performance, low latency, online (real time) operation.

Work Done

Workload Characterisation and Profiling

The focus of these activities characterising the workloads as appropriate to the use cases identified in WP2. The characterisation and profiling involves the application of full stack telemetry coupled with the development of novel methodologies and analytics to investigate the behaviours of workloads under various operational conditions. The characterisation of workloads/services has been an area of continued focus in Task 4.1 in year 2 and year 3.

TALE Methodology

The characterisation approach developed by the task was formalised into the TALE methodology. This workload characterisation methodology focuses on four key aspects of workloads in a virtualised cloud environment, namely Throughput, Anomaly, Latency and Entropy (TALE). TALE enables the description of a system in terms of the T/L relationships within a single resource and among distributed resources within the same system. The **Throughput and Latency** elements for a workload focus on what is being processed per unit of time (e.g. second) and the time each element takes to be processed. Standard categories of metrics, which apply in this context are: Compute, Network, Storage, Memory and Hypervisor. Investigation of **Anomalies and Entropy involves the** construct baselines based on each unique configuration and building statistical models that can predict anomalous behaviour within certain confidence intervals. For example when characterizing the Unified Origin workload anomalous behaviour was found with respect the behaviour to the number of simultaneous TCP connections that could be supported. As a result, the Apache Multi-Processing Module, which implements a hybrid multi-process multithread server was enabled. By using threads to serve requests, it is able to serve a large number of requests with fewer system resources than a process-based server, which is important in the context of the Unified Origin service and its ability to handle large volumes of user requests

Predictive KPI Mapping

The KPI mapping work carried out in year 1 was extended to provide a predictive KPI mapping capability. Using a machine learning approach, a model was developed that demonstrated the ability to predict the latency or throughput KPI's for the Unified Origin workload for a future time epoch with an accuracy of >90% provided the behaviour of the relevant metrics identified can be estimated. However, the results are only valid for this particular use case, as the confidence intervals were found dependent on the data quality, system stability and configuration, making it difficult to fully generalise.

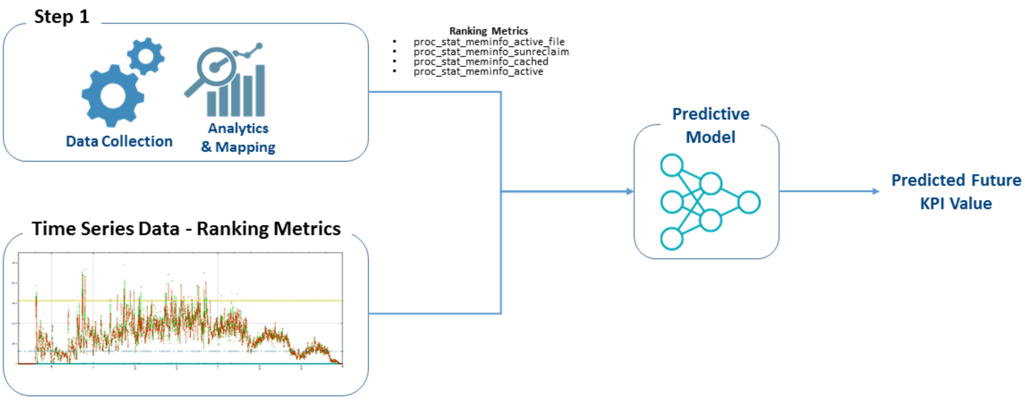


Figure 7: Predictive KPI Mapping Process

Dynamic Fingerprinting

The Dynamic Fingerprinting activities have been focusing during year 1 on analysing insights related to the behaviour of a service in terms of both static allocation of the infrastructure resources (quantity and type of resources) and the real dynamic usage of those resources from a service perspective. The methodology previously defined has been generalized in order to allow the collection of this information for a generic service. A single node testbed has been implemented. A video transmuxing service was designed, deployed and tested on the heterogeneous infrastructure testbed. An automated methodology has also been implemented which deploys the transmuxer instance. The deployment methodology also features full stack telemetry collection. The testbed has been connected to a landscape monitoring technology to collect information about the Virtual Machines and the physical resources assigned to them. A methodology for data gathering after the experiment execution has been implemented in order to collect information from both the sources (telemetry and landscape/topology) and automated to support the fingerprinting methodology.

Workload Modelling

The workload modelling activities have focused in developing an understanding and insights into approaches, which express the affinity of a workload/service or a specific host or set of ingredients.

These approaches considered the context of cost and service assurance compliance in order to inform workload placement decisions across heterogeneous resource environments.

Block Abstractions

The block abstractions research work has focused on defining, implementing and evaluating a scalable and efficient approach to exposing platform ingredients/features available within a heterogeneous data centre, in a manner that enables an orchestrator to improve the decisions related to each specific service when deploying complex services. The improvement in decision targets reducing the overall total cost of ownership (TCO) for a given service deployment. Secondly the approach targets the service fulfil in the context of achieving KPI's for a given level of confidence.

A heterogeneous testbed has been implemented. A video transmuxing service was designed, deployed and tested on the heterogeneous infrastructure testbed. An automated methodology has also been implemented which deploys the components of the service as virtual machines i.e. VPNs, load balancer and transmuxer instances. The deployment methodology also features full stack telemetry collection via Snap and an analytics module. An initial experimental campaign has been completed which investigated resource deployment costs versus KPI performance indicators (Latency and Throughput). The campaign involved deploying the VMs in different combinations over the testbed and interrogating the performance of each deployment. Block abstraction were generated which relate specific cost points and KPI performance levels. The next steps in the research are focused on developing a reinforced learning approach, which can determine how the number of deployments can be reduced which providing sufficient resolution of block across cost versus performance curve.

An alternative block abstraction method that focuses on how to work with heterogeneous hardware deployments has also been developed for the MicroVisor technology. This block abstraction uses an OnApp proprietary data model for mapping the resources and the dynamic telemetry information through the MicroVisor APIs. The telemetry information from heterogeneous hardware resources can thus be collated and reasoned on for determining the workload placement that best matches the policies that have been enforced either programmatically or through the UI. For the final demo, effort will be undertaken to attempt to get the telemetry information from the OnApp MicroVisor source interacting with the remote telemetry information in order to use the remote decision logic.

Data Models and Information Model

The Superfluidity Information Model (SIM) based on a hierarchical modelling approach was finalised. The high level model comprised of three individual information models, namely RDCL and RDCL3D data model, the NFVI orientated Infrastructure landscaper data model, and the CRAN service deployment data model. Each of these cover different domains of discourse although they have some similar overlapping concepts. Building on the foundational model implementations the overarching information model for Superfluidity was defined.

The SIM relates the concepts of each of the three areas together to provide a unified, Superfluidity information model. The model was generated in OWL-RDF and has been provided as an open source contribution from the project. The work in the task fed into the implementation in WP7. The overall information model helps to relate concepts from the three separate areas such that developers working on the three modelling systems can use the information to integrate the separate tools into a unified, integrated system.

Innovative Radio and Network Processing Functions

C-RAN and Fronthauling

One of the key objectives in task 4.3 is the design and implementation of an end-to-end Cloud RAN solution. Two aspects are specifically addressed: cloudification of the RAN and making the fronthaul network re-programmable.

The RAN was re-factored using a micro-service architecture and RFB principle where the RAN protocol stack was broken down into a set of horizontal functions that can be combined together, assigned with target performance parameters, mapped onto the infrastructure resources, and finally delivered as a service. The work was based on the OpenAirInterface (OAI) LTE implementation.

To reduce the latency, which may be added by using hypervisor technology, the Cloud RAN solution was implemented using Docker, as the virtualization technology, offering performance comparable to bare metal. The usage of Docker simplifies function portability as well as migration between different computing nodes. The Remote Radio Head (RRH) is enhanced with functionality, which is able to send IQ samples over Ethernet. This functionality is called RRH-GW and run as a Docker container.

To connect the antenna site (RRH) to the EDGE cloud, a CPRI link is normally used. In the task, having a fully re-programmable fronthaul solution which can select the routing path according to the requirement of the flow in terms of latency and throughput is a key focus.

An Ethernet based solution was developed using an overlay network composed of Open Virtual Switches to deliver the required flexibility. Using a SDN controller (in our case ONOS), and the intent programming, we succeed in connecting the enhanced RRH and the EDGE cloud in a software manner and making it fully re-programmable.

Another fronthaul technology investigated in the task is microwave transmission which has the potential to lower the cost of the deployment, particularly in challenging areas.

Packet Manipulation Processor

Significant progress has been made on the implementation of a Packet Manipulation Processor (PMP) to provide complex packet modifications at line rate is described. A detailed architecture for the PMP, along with the relevant design choices in terms of memory arrangement and interaction with an SDN switch pipeline has been proposed and documented.

Dynamic Dataflow Application Transformation for C-RAN Baseband Processing

Finally, progress on the implementation of a dynamic dataflow application of for C-RAN baseband processing is reported together with a description of the necessary abstraction models and API support required for dataflow graph transformation. Implementation of graph transformation techniques have also been completed by the task. . According to the proposal, the activities related to radio functions and their runtime support has been divided into WP4 (T4.3 - modular C-RAN baseband application) and WP5 (T5.1&T5.2 – dataflow runtime engine and dynamic scheduling and allocation algorithms). However, due to the very high dependency between activities in both areas the developments have been done in close cooperation and small iteration steps. The results are described in deliverable I4.1b and due to outlined reasons were included as part of activities of WP5.

In the previous project period, the highly flexible C-RAN baseband functions were developed according to the dataflow paradigm. The baseband functions are represented by dataflow graphs supporting the expression of application structure by means of tasks producing/consuming data and inherent parallelism for parallel processing on multi-core platforms. The granularity of dataflow models affects the deployment flexibility i.e. the capability of efficient mapping of tasks to parallel computing resources. However, it negatively affects performance scalability due to the overhead associated with task management. Therefore, in the second year of the project, the motivation was to address the management overhead by investigating and implementing methods for task graph transformation, employing a dynamic task clustering approach. Based on this approach, optimized baseband functions with reduced structural complexity have been developed and evaluated.

In order to support dynamic task clustering, the dataflow engine API and runtime have been extended by associated data structures and locking mechanisms that allow task clustering at any stage of processing i.e. task generation, task-graph analysis, task scheduling, task execution. This approach enables splitting the clustering process into critical and non-critical parts and, consequently, to move the non-critical parts from the critical path of dataflow engine to task generation stage. Moreover, using this approach it is possible to optimize the graph structure manually in the task generation stage and/or automatically by implementing task clustering algorithms in the graph analysis and/or task scheduling stages. The principle of task clustering is illustrated in 8. On the left hand side is the clustering of task T3 with T2 in task generation stage while right hand side shows graph clustering in in the graph analysis stage. Both approaches are supported by transformation API implemented in the dataflow engine.

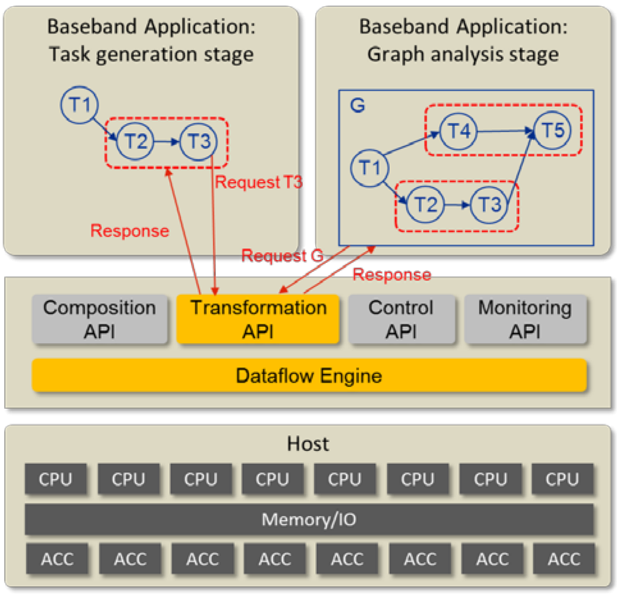


Figure 8: Principle of task-clustering approach and the associated system components and API.

In order to allow scheduling and execution of both simple and clustered tasks, the dataflow engine runtime has been extended. In particular, the dynamic list task scheduler has been modified allowing transparent handling of single and clustered tasks. Accordingly, the controller of PE-threads has been extended to support clustered tasks. Real-time monitoring and offline analysis tools have been also updated.

Based on the task-clustering support, the existing C-RAN baseband function have been re-designed and optimised regarding their structural complexity. Primarily static and dynamic linear clustering approaches have been implemented for baseband user signal processing parts in task generation phase. The evaluation and analysis of this approach for various deployment and operational scenarios demonstrated significant overhead reduction. Later the algorithms for automated clustering will be implemented into the task analysis stage, which is important for automatic application adaptation to underlying HW platform.

4.3.4 WP5 Virtualisation Platform Implementation and Network Dynamics

This WP builds on the hardware profiling and high performance block abstractions developed in WP4 in order to design and implement a converged, multi-tenant, virtualized high-performance platform that complies with the principles of the Superfluidity architecture: (1) platform-independence, meaning that a common API and optimal algorithms for resource allocation should hide, as much as possible, the fact that the actual network services may be run on a range of different commodity hardware and accelerators, from powerful Intel Xeon-based x86 servers with FPGAs and GPUs all the way down to low-frequency, low-power ARM-based microservers; (2) time-independence, meaning that the virtualized services can be deployed near instantaneously (e.g., in milliseconds); and (3) location-independence, whereby such services can be run on any Superfluidity platform, and migrated between them without the end-user noticing the change (e.g., in hundreds of milliseconds or less). On the high-end of the hardware possibilities (e.g., a multi-core, x86 server), the target is to be able to provide a platform that can cope with data rates of 10Gb/s and higher, and can host up to hundreds or perhaps even thousands of virtualized network services on a single server (the latter would help ensure that there is, with high probability, always room for one more network service to be run, and to reduce energy costs through massive consolidation of virtualized processing).

Objectives

WP5 has the following key overall objectives:

1. Implementation of novel allocation algorithms that match the functional abstractions derived from network services in T4.3 to block abstractions (T4.2), to automatically derive optimal configurations and allocations without requiring the programmer to understand low-level system details.
2. Support for near transparent, cloud-like dynamics for virtualized network services, including close to instantaneous instantiation (milliseconds), massive consolidation (thousands of functions on a single server) and transparent migration (without the end-points noticing).
3. Integration of the provisioning framework and the mechanisms of the previous objectives into a common, unifying Superfluidity platform, and an API to allow management of the platform and the network services running on it.
4. High performance processing on heterogeneous hardware at speeds of 10Gb/s and higher.

Work done

Efficient Resource Allocation

This year the project has seen important progress regarding the problem of efficiently allocating services (whether applications, VMs or containers) onto the available hardware, while taking into account changing network dynamics and changing loads placed on the platform by the services. Towards this goal, the project has and is working on a number of different approaches. The first of these is based on Multi-Attribute Utility Theory (MAUT), which supports minimisation of platform resource usage while ensuring that the selected resource allocation and placement option meets the KPIs requirements for the service and SLAs; we are now in the process of analysing the results.

The second approach relies on machine learning techniques, and in particular reinforcement learning, to build an efficient, dynamic resource allocator. In the past, in order to make the problem tractable, resource allocators have overly simplified the problem, purposely ignoring issues to do with changing network matrices, changing services, effects of changing network traffic on those services (e.g., a DPI suddenly having to execute an expensive execution path on a particular flow) and noisy neighbour effects. Machine learning models tend to thrive in such complexity, so the insight is that they should be a good match for this particular problem. More concretely, we have started creating predictive models of applications and services based on telemetry gathered by the Intel's open source Snap framework as well as other sources of performance data. Based on this, we are starting to build a resource allocator that will include a reinforcement learning mechanism that will improve allocations as it runs. We have a very early implementation of this and should have results in the coming months.

Related to this effort, we are working on predictive throughput models based on machine learning techniques for Click-based configurations. The work includes deep learning methods, and more specifically convolutional learning methods for graphs (Click configurations are basically directed graphs). This is particularly relevant to the project since several of its technologies (FastClick, LightVM/ClickOS) are based on Click. As with the previous items, we expect results within the coming months.

Superfluid Platform

In this period, the project has made great strides towards designing and implementing the project's platform. The main component behind the work is called LightVM. The motivation behind the work was to provide lightweight virtualization, the kind that is commonly found in containers and the reason why these have become so popular, without having to sacrifice isolation/security (as containers tend to do). In this work, we find that there are no fundamental limits to making VMs as nimble as containers, as long as the VMs are small and the toolstack is fast enough.

We achieve lightweight VMs by using Unikernels for specialized applications and with Tinyx, a tool that enables creating tailor-made, trimmed-down Linux virtual machines. By themselves, lightweight virtual machines are not enough to ensure good performance since the virtualization control plane (the toolstack) becomes the performance bottleneck. LightVM is a new virtualization solution based on Xen that is optimized to offer fast boot-times regardless of the number of active VMs. LightVM features a complete re-design of Xen's control plane, transforming its centralized operation to a distributed one where interactions with the hypervisor are reduced to a minimum. LightVM can boot a VM in 2.3ms, comparable to process startup times (1ms), and two orders of magnitude faster than Docker. Further, it is possible to pack thousands of LightVM guests on modest hardware while having memory and CPU usage comparable to that of processes.

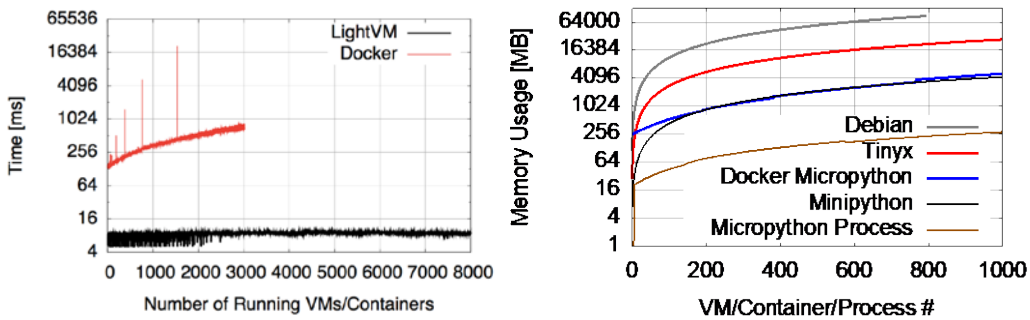


Figure 9: Superfluid Platform performance

[Left] Massive consolidation showing the platform's ability to boot up to 8,000 guests concurrently (LightVM curve) in shorter times than Docker containers. [Right] Memory usage for up to 1,000 guests compared to Docker.

A future deliverable will contain full details about LightVM, its implementation, and the results. As a preview though, the graph above (left) shows boot times when running large number of LightVM virtual machines, up to 8000, compared to Docker containers. In this massive consolidation scenario, LightVM is able to boot virtualized, isolated services in as little as 8.6 milliseconds even for the 8,000th VM. The graph on the right shows that LightVM consumes very little memory (Minipython is a LightVM virtual machine specialized for running Python), certainly no more than containers do. Our platform has lower memory requirements than Docker when using Unikernels (Minipython curve) and only slightly higher when using specialized, Tinyx distributions.

Another, complementary piece of the Superfluid platform has concentrated on obtaining high throughput even when facing changing network conditions. The first part of this consists of HyperNF, a high performance NFV framework aimed at maximizing server performance when concurrently running large numbers of NFs. To achieve this, HyperNF implements hypercall-based virtual I/O, placing packet-forwarding logic inside the hypervisor to significantly reduce I/O synchronization overheads. HyperNF improves throughput by 36%-83% depending on the NF, is able to closely match required performance guarantees (with deviations of only 3.5%), and to efficiently cope with changing traffic loads, yielding an improvement of up to 86% in cumulative throughput. The second part of this work is ONAPP's Microvisor, which implements features such as Ethernet drivers directly in the hypervisor. Finally, we have designed and implemented mechanisms for fair CPU sharing which rely on sending congestion control proportional to per-packet cost for each flow. This enables endpoint congestion control (e.g. TCP) to react appropriately and share the CPU fairly.

Platform API and Orchestration

The Platform API activity aims to merge all different components of Superfluidity under a coherent, uniform platform and to expose an API to it, to be used by the project's components and applications. This work (which is reported in Deliverables I5.3 and D5.3) aims to include the integration of the network service dynamics and allocation algorithms, as well as the security considerations developed in T3.3.

Before any integration activity could begin, a careful analysis of the state of the art was necessary, for properly evaluating the available components and tools that may be utilized by Superfluidity. In particular, the latest advances relevant to OpenStack were analysed and presented in Deliverable I5.3, namely OpenStack Compute and Neutron, HAProxy and LBaaS, Heat, Mistral as well as the OpenStack Telemetry. In addition to OpenStack, an analysis of the current status of the Architecture and Reference Points of ETSI NFV was presented, emphasizing on the necessary network interfaces for the management and configuration of Network Services and VNFs, the NFVI functions and the virtual resource interfaces that should be implemented towards an end-to-end solution.

Last but not least, Deliverable I5.3 presented a detailed gap analysis of certain NFV Orchestration frameworks, initially Open Source MANO, along with the Superfluidity mechanisms, algorithms and innovations that will further enhance Superfluidity's vision of an efficient 5G architecture, such as RDCL 3D, SymNet together with SEFL and also NEtwork MOdelling (NEMO) language.

Using the above SotA/gap analysis as a basis, deliverable D5.3 reported on the remaining work of this task (T5.3). Specifically, D5.3 covers the Superfluidity extensions to the ETSI NFV Information Models. These were not only proposed for inclusion in standards, but were prototyped as well: Both as part of the project (i.e. in RDCL 3D), but potentially also in an upcoming ETSI Open Source MANO (OSM) release.

D5.3 also included the integration of a second NFV Orchestrator, ManageIQ, enhanced further with the ability to execute Ansible playbooks. D5.3 moreover introduced support for containers, using Kubernetes, also covering the VM-container inter-networking aspects, using Kuryr.

Finally, D5.3 covered the integration of an exemplary SDN orchestrator, ONOS, as well as the implementation and integration of the project's Multi-Access Edge Computing (MEC) platform.

The above effectively combined all the outcomes of WP3, WP4, WP5 and WP6 into the Superfluidity platform, which is further validated in the context of WP7.

Regarding orchestration, in this period the project has concentrated a significant amount of effort into integrating the Superfluid platform technologies (e.g., ClickOS/LightVM) into existing orchestration frameworks. The work consisted of work on VIM (Virtual Infrastructure Managers) optimizations for Unikernels. In detail, we have extended OpenVIM to support the ClickOS Unikernel in a backward compatible way, starting from the latest release of OpenVIM. In addition, we have added support for heterogeneous hypervisors in OpenVIM, including support for XEN virtualization (HVM/Qemu) and for XEN virtualization for the ClickOS Unikernel. The patch for this work will be proposed to the OpenVIM community. Finally, we have conducted a series of performance optimizations targeted at improving boot times and other metrics when dealing with lightweight virtualization guests such as Unikernels.

4.3.5 WP6 System Orchestration and Management Tools

This WP is devoted to the provisioning and control framework, including automated security verification of network processing code. We then report the main achievements in the following tasks: i) control framework design, ii) tools and algorithms for SLA based deployment, iii) modelling and design for symbolic execution and monitoring tools.

Design of Control Framework

In order to tackle the challenge of designing a control framework, our approach was split into several steps. As a first step we started by analysing the use cases from WP2 as our input. The objective was the identification of shared attributes and the identification of common requirements that the use cases shared. Such requirements come from NFV, MEC, C-RAN technologies. After doing so, we had the next step ready – investigation of the aforementioned requirements' support in existing orchestration solutions. As a last step, we need to identify the gaps between the requirements and each solution capabilities. The following table reports the options considered for each scope, as well as the preferred ones. Finally, part of the activities considered in this task has been devoted to the investigation of management software tools.

Table 4: Considered options for the control framework

Scope	Options	Comment
Cloud Infrastructure	One NFVI per Service	Inefficient and Complex
	Common NFVI for all Services eventually locations	Preferred
Cloud Infrastructure Management	One local VIM per NFVI	Acceptable
	Single centralised VIM for all NFVIs	Acceptable
	Hybrid solution between the previous two	Preferred
Cloud Management and Orchestration	One Orchestrator for all services and locations	Not realistic
	One Orchestrator per service	Preferred
Orchestration Layer	Northbound and Southbound Interfaces	Acceptable
	Eastbound and Westbound Interfaces	Difficult
	Hybrid solution between the previous two	Preferred

Initial Design for SLA based deployment (I6.2)

We have first presented NEMO – a language to be used as SLA descriptor, for an application owner to declare the QoS requirements of her application. NEMO is a human-readable command language used for Network Modelling. NEMO provides basic network commands (Node, Link, Flow, Policy) to describe the infrastructure and controller communication commands to interact with the controller. We are proposing to extend the Node definition command to import TOSCA or OSM based descriptors as Node definitions. Since Node models can make use of previously defined node models, the resulting language would be recursive and therefore support our notion of (recursive) reusable function blocks.

Then we considered resource allocation and placement problems in different contexts: i) service chains in NFV deployments, ii) MEC applications, iii) generic formulation of a service scaling problem in NFV environment.

For service chains, we study two possible placement strategies – one that gathers all components of each chain into the same host, and one that distributes them between different hosts. This is performed both in the presence of hardware acceleration (DPDK), and without it. Following the comprehensive evaluation of the two placement strategies of service chains, we also model the cost of network switching. Given an arbitrary number of service chains, our model accurately predicts the CPU cost of the network switching they require.

In the context of MEC applications, we discuss the management and orchestrator and the resource allocation problem. We also examine service migration for MEC applications.

As for the generic service scaling problem, we focus on the scale in/out operations in VNF allocation. In particular, we model the decision whether to increase the resources allocated to a certain VNF type (scale out operation) or to release some of these resources (scale in operation) based on the current and expected demand from the network service and the required QoS as a Markov Decision Process (MDP). Our formulation also incorporates the load balancing challenge steering the traffic flows to the different VMs and balancing the load between them.

Modelling and Design for Symbolic Execution and Monitoring Tools

As reported in the WP3 description, SUPERFLUIDITY proposes the concept of RFBs as a modelling tool that supports APIs for 5G networking. RFBs have a specification described in a higher level language that allows their users to compose them correctly, and can be implemented in multiple ways as long as they obey their specification: software (as a monolithic block), hardware (ASICs), or a composition of other RFBs. How can we secure the resulting 5G networks? Enforcing network security has two major phases: the network operator specifies higher-level policies and then implements them using (low-level) networking functionality typically provided by third-party vendors. High-level policies could include access control lists (who can talk to whom), firewall rules, routing protocol configurations, and so forth. Networking functionality includes switches, routers, middleboxes, etc. In traditional networks, high-level policies are fairly static and thus easy to manually check and deploy infrequently. Traditional networking hardware (switches, routers, simple firewalls) processes packets on custom-made ASICs that are thoroughly verified and seldom updated; such implementations give a fairly strong low-level security guarantee. In 5G networks, network functions will be instantiated dynamically, and the network will run services configured not only by the operator, but also by third parties. Running third-party processing will be a major revenue source for operators and is thus very attractive; however it can expose operator networks to many security risks that must be addressed. The great benefit of software network functions is that it can be easily upgraded and it avoids vendor lock-in. On the downside, diverse software with increasingly complex functionality and developed by many vendors is much more susceptible to low level exploits such as buffer overflows.

To enforce security in 5G networks SUPERFLUIDITY takes the following main directions:

1. Describe operator policies in a high-level specification language.
2. Describe RFBs in a way that is amenable to static analysis.
3. Perform static analysis of RFB configurations to ensure policy is obeyed before deployment.
4. Ensure that the implementation conforms to the specification at deployment time.

Specifically, there is a gap between the abstract model of the network that we can statically analyse (steps above) and the actual implementation. This gap appears because it is impossible to verify large C implementations of networking code in useful time.

Even if we assume the implementations of networking functionality are correct, similar gap appears when multiple abstractions are applied in a network, for instance network virtualisation.

To bridge this gap, we have relied on multiple techniques, some of which are known and tested but have limited coverage, and some which are novel and are still subject of on-going work. We list them here:

- Active testing. Intuitively, once the static analysis results are known, packets are generated for each path resulting from static analysis and injected in the real network; the outcome is then checked to see if it matches that predicted by static analysis. As all other active techniques, this is lightweight and useful, but it is not sufficient on its own because it has poor coverage and cannot give any type of strong guarantees.
- Monitoring and anomaly detection. This is another runtime technique that applies machine learning to understand the standard behaviour of the network and detect attacks when the behaviour deviates from standard.
- Static analysis at lower layers. Symbolic execution can be run on the model, and on the lower level implementation of the model; if the resulting outputs are equivalent, then the implementation is correct. Defining equivalence is not easy, and different definitions capture different parts of the problem we want to solve.
- Automatic generation of C implementations from SEFL code, which is memory-safe and easy to symbolically execute, thus allowing to prove it satisfies the high level properties of the operator.

4.3.6 WP7 System Integration and Validation

Superfluidity WorkPackage 7, "System Integration and Validation", started on project month 12 (M12, Jun/2016). The WP was organized along three Tasks:

- T7.1, "Selection of Use Cases for Deployment and Field Trials": as the name states, this Task has as main role, the definition of a set of use cases to be used to demonstrate and assess project results, in an integrated demonstrator.
- T7.2, "System Integration": selected use cases were run in the scope of an integrated project demonstrator, joining together the required project components, constituting an integrated prototype.
- T7.3, "Use Cases Validation and Assessment": taking the integrated system achieved in T7.2, this Task validated the obtained system and assessed it, considering functionality, performance and security aspects, whenever they apply.

At the end of the project (M33), all three Tasks run successfully. Results were documented respectively in D7.1, D7.2 and D7.3, with intermediate versions I7.1, I7.2 and I7.3.

To support the project developments, especially related to Task 7.2 activities, the project deployed a distributed private cloud (central and edge clouds), hosted at BT and Nokia FR premises.

To fulfil the established objectives, the WP identified and documented a set of 14 use cases. This is documented in D7.1. Those use cases have been organized as a sequence of scenes, thus emphasizing the integration focus of the work, and the run through all operator services and network lifecycle. A total of 28 project components were adopted to implement the 14 scenes/use cases. The details are provided in deliverable D7.2, Figure 10 shows the integration of all the components in the demonstrator.

Besides the integrated testbed, the project also supported the execution of PoC and performed other assessment work, complementing the main work related to the use cases execution. The assessment made, shows how Superfluidity technology contributes to identified business and technical requirements, set forth in D2.1. Together, with the functional, performance and security assessments made, it is all documented in D7.3.

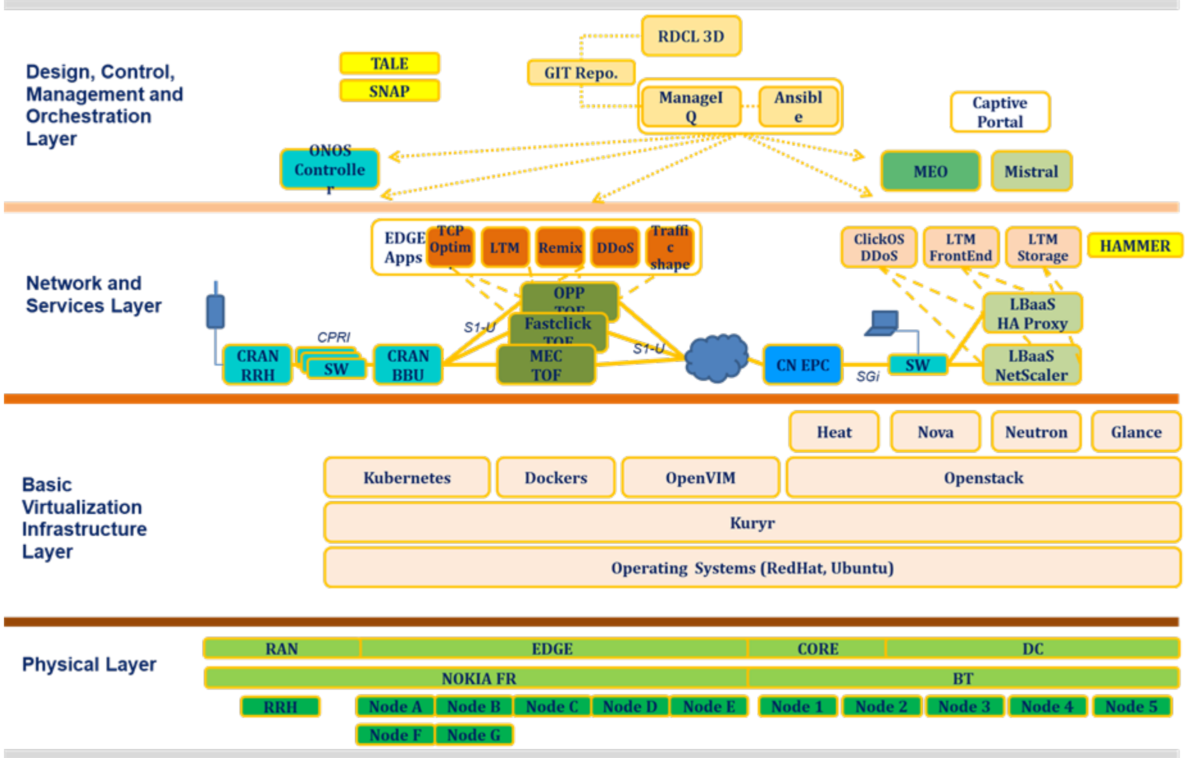


Figure 10: Overall Integration of Components in the Integrated Demo

4.3.7 WP8 Communication, Dissemination, Standardisation and Exploitation

The following objectives have been pursued by the project and supported by WP8: i) open source contributions, ii) initial proof of concepts, iii) contributions to standardisation organisations related to 5G, iv) organisation of special sessions at events like OpenStack Summits and Xen Hackathons, v) “Marketing-oriented” presentations at industry events, vi) Discussions with policy makers, social and environmental organisations regarding project results, vii) Cooperation with other peer 5G-PPP projects in the framework of the 5G-PPP overall programme, viii) Scientific publications at top tier conferences, journals and workshops, ix) Wiki-style website allowing easy retrieval of main project data (including software repositories), as well as social networks presence, x) Press releases targeting the public at large to allow for wide dissemination of project concepts and results.

Moreover, an innovation and management plan has been pursued, including the following activities:

- Overall project responsibility for identifying and driving the innovation and commercial potential for the technical work and goals of the project.

- Ensure of a common approach for Innovation and Commercial Exploitation in all work packages while maintaining the overall vision of the Project.
- Identification of potential innovation and commercial exploitation inhibitors in and between the work packages while implementing the technical work.
- Preparation of proposals for the Management Board on innovation concepts. Preparation of summaries for the periodic reports and annual reports, and for the reviews.
- Delivering of relevant presentations both internally and externally to the industry.

The deliverables D8.6, D8.7 and D8.8 document this work. Their contents are not reported here, as they are already written in the form of a report and are public.

4.4 Scientific and Technological Impact

At the time of writing is rather early to assess SUPERFLUIDITY's impact; thus, this section and the two following ones will be suitably expanded in the next version of this document. However, the work performed during the first year produced some visible and tangible results.

As regards scientific publications, SUPERFLUIDITY is targeting top tier research conferences and journals. These leading venues shape future research in this area and as such provide maximum visibility for the project results. Several partners have an established presence and a strong track record in such venues. The current list of publications is reported in http://superfluidity.eu/results/dissemination/.

Another important activity to exert impact is of course standards. Standards are a key way of ensuring that SUPERFLUIDITY has a real impact, in terms of actually achieving the "softening" of the network. The standards process distils the best technical approach, as well as helping to ensure interoperability and that operators can source components from different / multiple vendors. With critical input from SUPERFLUIDITY people, ETSI have just established a new work item on 'end to end process descriptions' (this builds on some of our ideas about recursion and common interfaces - work is just starting), and there was a meeting between many SDOs and industry groups in order to discuss their various Information Models (to start converging them and understanding how far convergence needs to go). Several SUPERFLUIDITY people play leading roles in standardisation activities: at the ETSI Industry Steering Group on NFV, BT's Andy Reid is Vice Chair of Open Source MANO (OSM), Chair of the End User Advisory Group and Rapporteur for the new Work Item on 'end to end process descriptions'.

4.5 Market Impact

At the time of writing, there are no products that have been developed that benefit from SUPERFLUIDITY. Based on events and discussions there would appear to be a high interest in the potential outcomes of SUPERFLUIDITY. Most of the demonstrations are tied in with expected commercial needs of the partners involved in those demonstrations and as such it is expected that in the revision of this document there will be a significant change in this section. In the research field, the papers that have been submitted have had an impact in their relevant areas, with some of the challenges posed providing new areas for investigation.

4.6 Societal Impact

At the time of writing it is too early to provide measurable results on the societal impact other than in the areas of skills related to 5G. Through research, having active participation in the 5G whitepaper and related 5G phase I discussions, the partners involved in SUPERFLUIDITY have been involved at the forefront of 5G technology discussions.

5 Roadmap

Now that the project is finished, the roadmap concerns the exploitation of project results by the Superfluidity partners. The exploitation actions and plans have been described in deliverable D8.8.

6 References

[HOLLY] Global Online TV, Video Revenue Forecast to Hit $51.1B by 2020 http://www.hollywoodreporter.com/news/global-ott-revenue-forecast-netflix-802417

[JAI13] R. Jain and S. Paul: "Network Virtualisation and Software Defined Networking for Cloud Computing: A Survey", IEEE Communications Magazine, Nov 2013, pp. 24-31

[JAN15] S. Jankowski, B. Feldman, Duval, D. Takyama, T. Hari, D. Lu, I. Matsuhashi, M. Shin, D. Clark, M. Delaney and In Y. Chung, 5G: How 100x faster wireless can the future, Goldman Sachs Equity Research, April, 2016

[MEC] "Mobile Edge Computing – Introductory Technical White Paper". https://portal.etsi.org/Portals/0/TBpages/MEC/Docs/Mobile-edge_Computing_-_Introductory_Technical_White_Paper_V1%2018-09-14.pdf

[MICRODC] Benjamin Frank, Ingmar Poese, Yin Lin, Georgios Smaragdakis, Anja Feldmann, Bruce Maggs, Jannis Rake, Steve Uhlig, and Rick Weber. Pushing cdn-isp collaboration to the limit. SIGCOMM Comput. Commun. Rev., 43(3):34–44, July 2013.

[SMART] Smart Cells Revolutionize Service Delivery, Intel white paper. http://www.intel.co.uk/content/dam/www/public/us/en/documents/white-papers/smart-cells-revolutionize-service-delivery.pdf

[TERA] Deutsche Telekom tests TeraStream, the network of the future, in Croatia. http://www.telekom.com/media/company/168008

[WER13] Werthmann, Grob-Lipski, Proebster: Multiplexing Gains Achieved in Pools of Baseband Computation Units in 4G Cellular Networks. PIMRC 2013, London