SUPERFLUIDITY

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

Research and Innovation Action GA 671566

DELIVERABLE D8.7:
FINAL REPORT ON STANDARDISATION AND OPEN SOURCE CONTRIBUTIONS

Deliverable Type:	Report
Dissemination Level:	Public
Contractual Date of Delivery to the EU:	31/03/2018
Actual Date of Delivery to the EU:	18/04/2018
Workpackage Contributing to the Deliverable:	WP8

Editor(s):	Luis Tomás, Daniel Mellado (RED HAT)
Author(s):	Luis Tomás, Daniel Mellado (RED HAT), Maria Bianco, Nicola Blefari Melazzi (CNIT), Michael J. Mcgrath (INTEL), Francisco Fontes, Carlos Parada (ALB), Bessem Sayadi (NOKIA FR), Erez Biton (NOKIA IL), Philip Eardley (BT), Rufael Mekuria (USTR), Cyril Soldani (ULG), Christos Tselios, George Tsolis (CITRIX), Felipe Huici (NEC)
Internal Reviewer(s)	George Tsolis (CITRIX)

Abstract: This deliverable provides the final report on the standardisation and open source contributions, and the impact that these have been perceived to have.

Keyword List: Standardisation, Open Source.

VERSION CONTROL TABLE

VERSION N.	PURPOSE/CHANGES	AUTHOR	DATE
0.1	First release	Luis Tomás	3/03/2018
0.2	Updated contributions from Intel, NEC, USP and ALB	Luis Tomás	2/04/2018
0.3	Updated Telefonica contributions	Luis Tomás	3/04/2018
0.4	Updated contributions from UPB and OnApp	Luis Tomás	3/04/2018
0.5	Updated contributions from BT	Luis Tomás	3/04/2018
0.6	Updated contributions from CITRIX and internal review	George Tsolis	16/04/2018

INDEX

Glossary

SUPERFLUIDITY DICTIONNARY	
TERM	DEFINITION
Communication	Targeted information to multiple audiences (including the media and the public) aimed at non specialists, including stakeholders whose interest is in potential application of the results
Results	Any tangible or intangible output of the Action, such as data, knowledge and information whatever their form or nature, whether or not they can be protected, which are generated in the Action as well as any rights attached to them, including Intellectual Property Rights.
OpenStack	Free and open-source software platform for cloud computing, mostly deployed as an infrastructure-as-a-service (IaaS). The software platform consists of interrelated components that control hardware pools of processing, storage, and networking resources throughout a datacentre.
Neutron	Neutron is an OpenStack project to provide "networking as a service" between interface devices (e.g., vNICs) managed by other OpenStack services (e.g., Nova).
Open Source Mano (OSM)	ETSI-hosted project to develop an Open Source NFV Management and Orchestration software stack aligned with ETSI NFV.
OpenDaylight (ODL)	An open source platform for programmable, software-defined networks.

Table 1 – SUPERFLUIDITY Dictionary

Executive Summary

This deliverable provides the final report on the standardisation and open source contributions performed and its impact into their respective communities.

1 Contributions to Standardisation

1.1 ALB

1.1.1 Contributions

1.1.1.1 ETSI MEC

The following table shows where ALB has contributed to the ETSI MEC activities. The purpose of these contributions is to influence the standards and be ready to develop a MEC Proof-of-Concept (PoC).

SUMMARY	CONTRIBUTION LINKS	IMPACT ON SUPERFLUIDITY PROJECT
Reference: MEC Architecture [GS MEC 003]	**MEC ARCH**	Defines the MEC overall architecture, including the edge level and system level. This is a key document that will guide the development work on the MEC area.
Reference: End-to-End Mobility [GS MEC 010-2]	**MEC Mobility**	Defines how the end-to-end user mobility is handled along the MEC system. This is an advanced MEC feature that will be targeted by the SUPERFLUIDITY project.
Reference: LifeCycle Management (LCM) [GS MEC 018]	**MEC LCM**	Defines how the lifecycle of the MEC Applications is managed. This is an advanced MEC feature that will be targeted by the SUPERFLUIDITY project.

1.1.2 Future Plans

1.1.2.1 ETSI MEC

ALB is actively contributing to ETSI MEC and will continue to be involved in the working items mentioned above and eventually on others that will be raised in the future and have particular relevant interest.

1.2 NOKIAIL

1.2.1 Contributions

NOKIA IL is active in two main SDOs, ETSI NFV and OASIS TOSCA. In the ETSI NFV ISG, the architecture (IFA) group deals with the architecture of a MANO system. To this end, it defines the interfaces of the different components, such as the NFVO and the VNFM, as well as the descriptor and VNF packaging format. We follow the ETSI activities very closely and regularly attend the IFA meetings. Moreover, we provide feedback to ETSI on implementation aspects and contribute to the discussions on the descriptor formats. These descriptors play an important part in SUPERFLUIDITY vision to allow QoS based and dynamic deployment of RFBs.

The recently created SOL WG has gathered a great attention in NFV community. Since the only current work item and many of the envisaged ones deal with service and VNF descriptors, there are important opportunities to contribute. Specifically, the TOSCA work of SOL is of great interest to SUPERFLUIDITY, as it is a leading candidate to be selected as the deployment language of choice.

In accordance to the SOL work, NOKIA IL keeps its involvement in TOSCA to push new descriptors to support the deployment and life cycle management functionalities envisioned by SUPERFLUIDITY.

1.3 Telefónica, I+D

1.3.1 Contributions

Telefónica, I+D (TID) has been promoting the SUPERFLUIDITY project in different Standardisation bodies. In this line, we presented the project objectives at the Internet Research Task Force (IRTF) during the IETF’94 in Yokohama. The project was presented both to the Software-Defined Networks Research Group (SDN-RG) and to the Network Function Virtualisation Research Group (NFV-RG).

TID has contributed as well the activities around the integration of Virtual Network Function Descriptors (VNFD) into the Network Modelling (NEMO) language to provide a human-readable, intent-based specification language for the Recursive Function Blocks (RFBs). The scope of this work extends RFBs beyond VNFs to ETSI-NFV Network Services and allows RFBs to be used in VNF

Forwarding Graphs. The contribution is available at https://datatracker.ietf.org/doc/draft-aranda-nfvrg-recursive-vnf/, and was first introduced at the IETF'97 (Berlin, July 2016). The latest version has been discussed at the IETF'101 (London, March 2018), taking advantage of the recent interest on NFV applications to Path-Aware Networking.

1.3.2 Future Plans

TID intends to continue this path on intent-based NFV specifications, and their application to the evolution of the Internet protocol stack, within the IETF, and possibly within the newly created ETSI ISG on Zero-Touch Service Management (ZSM)

1.4 Unified Streaming

1.4.1 Contributions

Unified Streaming occupies a unique position in the media technology industry space, in which it despite its small size was able to initiate and lead several standardization projects that have had a large impact on the industry. Up to July 2017 Unified Streaming chaired the MPEG PCC AhG that published a call for proposals in early 2017 (January and a second version in April). The call was developed in close collaboration with industry leaders like Sony, Technicolor, Huawei and others. The call results in responses from all leading mobile device manufacturers in October including Apple, Nokia, Huawei, Sony and Samsung. This has started the work on a new video codec for immersive and augmented reality video for the media industry. Unified Streaming also contributed to this call but, unfortunately, it was not a winning proposal.

In July 2017 Unified Streaming joined forces with Samsung Electronics to chair the AhG on Network Based Media processing which aims at developing standards intermediate data exchange formats for media processing distributed over the network such as at the edge. In this work, it is currently working on the call for proposals, use cases and requirements. The edge based processing LTM prototype developed in Superfluidity and the prototype for Remix have shown good results so far and the potential to show improved performance of video delivery content change. In addition Unified Streaming joined the DVB CM-I activity that will define standards for internet television services with similar quality as traditional broadcast services such as DVB-I. To summarize, within the superfluidity project timeframe Unified Streaming managed to start one large standardization activity on 3D point cloud coding. Further the ideas of Superfluidity have been accepted to MPEG and MPEG is trying to develop some support for the media vertical in edge converged cloud native networks such as superfluidity, where already companies like Nokia, ZTE, Samsun Dolby and many others have contributed.

MPEG JTC1/SC29/WG11 Geneva May/June2016 (http://mpeg.chiariglione.org/):

- m38753 MP3DG-PCC Software Platform for Point Cloud Compression (input)
- m38754 Proposal for MPEG-4 PCC features roadmap (input)
- m38801 Plane Projection Approximation for Voxel Colour Attributes Compression (input)
- m38816 Point Cloud Geometry Compression with Plane Projection Approximation and Learning Based Compression (input)
- W16333 Draft-dataset-point-cloud-coding (output)
- W16334 Draft Call for Proposals for Point Cloud Compression (output)
- W16335 Evaluation Criteria for Point Cloud Compression (output)
- W16336 Use Cases for Point Cloud Compression (output)
- W16337 Requirements for Point Cloud Compression (output)

MPEG JTC1/SC29/WG11 Chengdu October 2016 (http://mpeg.chiariglione.org/):

- m38962: Additional use case and requirement for network distributed video coding: Just in Time Transcoding as a Service (JiTTaaS)
- m38136: Point Cloud Codec for Tele-immersive Video
- m40100: Additional use case and requirement for network distributed video coding: JiTTaaS, relation to NDVC

MPEG JTC1/SC29/WG11 Geneva 2017 (http://mpeg.chiariglione.org/):

- m40084: Summary of Anchors Generated for Dynamic Point Cloud Data

MPEG JTC1/SC29/WG11 Hobart 2017 (http://mpeg.chiariglione.org/):

- m40279 Superfluidity: a 5GPPP future network architecture

MPEG SVN Software Tools for PCC: Anchors, Metrics and Evaluation

MPEG JTC1/SC29/WG11 Macau 2017 (http://mpeg.chiariglione.org/):

- m40279 Superfluidity: a 5GPPP future network architecture
- m40264 MPEG SVN Software Tools for PCC: Anchors, Metrics and Evaluation
- m40613 Notes and results of PCC Anchors for dynamic objects from 8i and Technicolor

MPEG JTC1/SC29/WG11 Gwangju 2017 (http://mpeg.chiariglione.org/):

Summary of NBMP contributions MPEG 120 and Seoul AhG meeting

Chair of MPEG PCC AhG (with Technicolor)

Co-Chair of MPEG NBMP (with Samsung and/or SK Telecom)

DASH-IF:

Following of the MPEG DASH SAND Taskforce

Unofficial:

Chairing of specification on media ingest protocol with BBC, Amazon, Microsoft, Akamai, MediaExcel, Harmonic, centuryLink and ComCast. https://github.com/unifiedstreaming/fmp4-ingest

1.5 BT

1.5.1 Contributions

BT is one of the companies that created the original work at ETSI and defined the concept of network virtualisation. BT believes that standardisation is key in the telecoms field, and certainly in the NFV space, in order to ensure interoperability and allow us to source components from different / multiple vendors.

BT has continued to play a leading role in NFV standardisation at ETSI Industry Steering Group. Andy Reid is Vice Chair of Open Source MANO (OSM), and Chair of the End User Advisory Group. We played a key role in the ISG's initiative to bring together the Information Models of many SDOs and industry groups. BT led the discussions for a new work item on 'end to end process descriptions' and was Rapporteur for the Work Item.

This work has progressed as a White Paper about "Experience with NFV architecture, interfaces and information models", which OSM will publish soon. BT initiated and is leading the writing of the Whitepaper. It will capture some of the points that have been learnt from implementing the NFV architecture, interfaces and information models. We hope that these will be considered by the ETSI NFV ISG as it develops its next iteration of the NFV standards. We believe that the OSM and ETSI NFV should target the following (linked) objectives in its next round of work:

- the network service should be presented as a single entity, even where it is constructed of sub network services,
- another way of putting this is that network services are recursive, meaning they can be built from other network services in hierarchical fashion,

- the northbound API needs to be parsimonious, meaning that it includes only what is necessary and all that is sufficient,
- this API should be client-server style.

2 Contributions to Open Source Projects

2.1 RED HAT

2.1.1 OpenStack Neutron

We have worked on several blueprint, design documents and discussions in order to add support for QoS as well as for the OVS-Firewall. The next covers the main work items performed within the Superfluidity project.

Blueprints and Design documents

- Role-based Access Control for QoS policies: https://bugs.launchpad.net/neutron/+bug/1512587
- devref: update quality_of_service: https://review.openstack.org/#/c/211259/
- Neutron QoS API extension: https://blueprints.launchpad.net/neutron/+spec/quantum-qos-api

Code Contributions

- Forbid attaching rules if policy isn't accessible: https://review.openstack.org/#/c/208543/
- QoS core extension: fixed dict extension when QoS policy is unset: https://review.openstack.org/#/c/210625/
- Gracefully handle duplicate rule creation: https://review.openstack.org/#/c/209056/
- Clean up QoS rules first, then QoS policies: https://review.openstack.org/#/c/210354/
- Don't claim Linux Bridge ml2 driver supports bandwidth limit QoS rules: https://review.openstack.org/#/c/210358/
- Introduce base interface for core resource extensions: https://review.openstack.org/#/c/209987/
- Add rpc agent api and callbacks to resources_rpc: https://review.openstack.org/#/c/208943/
- Propagate notifications to agent consumers callbacks: https://review.openstack.org/#/c/209620/
- L2 agent extension manager: read extensions list from config file: https://review.openstack.org/#/c/208223/
- Use single transaction to update qos policy associatation: https://review.openstack.org/#/c/208251/

- Guarantee there is only one bandwidth limit rule per policy: https://review.openstack.org/#/c/207170/
- QoS fixes: https://review.openstack.org/#/c/364123/, https://review.openstack.org/#/c/347028/, https://review.openstack.org/#/c/207043/, https://review.openstack.org/#/c/347017/
- Extensions to OpenStack SDK to enable QoS on Neutron: https://review.openstack.org/#/c/357230/
- QoS improved validation mechanisms: https://review.openstack.org/#/c/323474/

Testing/CI

- OVS agent functional test for policy rule delete: https://review.openstack.org/#/c/211537/
- Functional test for QoS policy bandwidth rule update: https://review.openstack.org/#/c/211173/
- OVS agent QoS extension functional test for bandwidth limit rules: https://review.openstack.org/#/c/210012/
- Add API tests for non-accessible policies: https://review.openstack.org/#/c/208466/
- Full stack tests for QoS and VLAN-Aware VMs https://review.openstack.org/#/c/372553/, https://review.openstack.org/#/c/369001/

2.1.2 OpenStack Kuryr

We have worked on several blueprint, design documents and discussions in order to add the mentioned capabilities at D6.1 into Kuryr upstream. The next covers the main work items performed within the Superfluidity project.

1.1 Nested Containers

Blueprints and Design Documents

- Nested-Containers: trunk subports management: https://review.openstack.org/#/c/402462/
- Update ports manager spec with reboot information: https://review.openstack.org/#/c/475389/

Code Contributions

- Nested-Containers: vlan driver: https://review.openstack.org/#/c/361993/

- Add support for nested pods with Vlan trunk port: https://review.openstack.org/#/c/410578/
- Add randomness to the returned vlan_ids to avoid vlan_id collisions: https://review.openstack.org/#/c/422641/
- Ensure subports are deleted after container deletion: https://review.openstack.org/#/c/424067/

Testing/CI

- Make segmentation driver testable: https://review.openstack.org/#/c/427190/

1.2 Ports Pool

Blueprints and Design Documents

- Kuryr Kubernetes Port Pool Manager design reference document https://review.openstack.org/#/c/427681/
- Adding ports pools to speed up containers booting and deletion: https://blueprints.launchpad.net/kuryr-kubernetes/+spec/ports-pool
- Improve control plane performance of the Port creation flows: https://blueprints.launchpad.net/kuryr-kubernetes/+spec/port-creation-control-plane-perf
- Improve control plane performance of the Port deletion flows: https://blueprints.launchpad.net/kuryr-kubernetes/+spec/port-deletion-control-plane-perf
- Updates ports pool doc information: https://review.openstack.org/#/c/529617/

Code Contributions

- Generic vif pool driver extension to precreate reusable ports: https://review.openstack.org/#/c/436877/
- Nested vif driver extension to enable ports reuse: https://review.openstack.org/#/c/436893/
- Nested vlan vif pool driver extension to precreate reusable subports: https://review.openstack.org/#/c/436894/
- Add ports pool configuration information at README: https://review.openstack.org/#/c/474589/
- Adding support for vif pool driver: https://review.openstack.org/#/c/436875/
- Delete or recover precreated ports upon controller restart: https://review.openstack.org/#/c/474576/

- Update documentation about nested deployment: https://review.openstack.org/#/c/487906/
- Add DevStack base installation section: https://review.openstack.org/#/c/497151/
- Ensure pool_key is properly created/retrieved: https://review.openstack.org/#/c/498353/
- Add methods to populate/free subport pools: https://review.openstack.org/#/c/498523/
- Avoid port update neutron call during pods boot up: https://review.openstack.org/#/c/504915/
- Recover precreated ports at NeutronVIFPool driver: https://review.openstack.org/#/c/507932/
- Optimize return_to_pool actions at Pool drivers: https://review.openstack.org/#/c/505605/
- Add devstack ports pool configuration options: https://review.openstack.org/#/c/510532/
- Speed up neutron request_vifs function: https://review.openstack.org/#/c/509989/
- Add Pool Manager to handle subports: https://review.openstack.org/#/c/498698/
- Add list and show pool commands to Pool Manager: https://review.openstack.org/#/c/504410/
- Avoid neutron calls at recovering precreated ports: https://review.openstack.org/#/c/510157/
- Add oslo_cache to default_subnet driver: https://review.openstack.org/#/c/519704/
- Add multi pools support: https://review.openstack.org/#/c/528345/
- Ensure pools recovery creates the right pool key: https://review.openstack.org/#/c/550728/
- Add network id to pools keys: https://review.openstack.org/#/c/548673/

1.3 LBaaS integration (LBaaSV2 and Octavia)

Blueprints and Design Documents

- Containerized Amphora Design Document: https://etherpad.openstack.org/p/ptq-queens-octavia-kuryr

Code Contributions

- Ensure DIB_REPOREF_amphora_agent is set: https://review.openstack.org/#/c/505301/
- Add support for service type=LoadBalancer: https://review.openstack.org/#/c/498253/
- Add Octavia L2 member mode support: https://review.openstack.org/#/c/499103/
- octavia: Make Octavia ready devstack: https://review.openstack.org/#/c/489157/
- Make ext subnet config optional: https://review.openstack.org/#/c/545270/
- Services: Set missing SGs for haproxy provider: https://review.openstack.org/#/c/545363/
- Services: Set SGs for N-S with haproxy provider: https://review.openstack.org/#/c/546777/

1.4 SDN Integration (networking-odl and networking-ovn

Blueprints and Design Documents

- Add OpenDaylight support to Kuryr-Kubernetes: https://blueprints.launchpad.net/kuryr-kubernetes/+spec/kuryr-k8s-odl-integration
- Add support and documentation for OVN: https://review.openstack.org/#/c/543556/

Code Contributions

- OpenDaylight support: Installation & Configuration: https://review.openstack.org/#/c/487885/
- Make precreated ports pool recovery not ML2/OVS specific: https://review.openstack.org/#/c/498399/
- (ODL) Remove JVM memory limitations by default: https://review.openstack.org/#/c/518540/
- (ODL) Update subport status for trunk ports: https://review.openstack.org/#/c/489517/
- (OVN) subports: add binding support to them: https://review.openstack.org/#/c/488354/

1.5 CNI Daemon

Blueprints and Design Documents

- CNI Daemon documentation: https://review.openstack.org/#/c/509380/

- Split CNI into its executable part and a long running daemon: https://blueprints.launchpad.net/kuryr-kubernetes/+spec/cni-split-exec-daemon

Code Contributions

- CNI split - introducing CNI daemon: https://review.openstack.org/#/c/515186/
- Support kuryr-daemon when running containerized: https://review.openstack.org/#/c/518024/
- Make daemon wait for VIF to become active: https://review.openstack.org/#/c/529092/
- Delete pods from registry in CNI daemon: https://review.openstack.org/#/c/545944/
- Move cni plugins to a common folder: https://review.openstack.org/#/c/548228/
- Make CNI Registry Plugin namespace aware: https://review.openstack.org/#/c/548313/

1.6 Containerized

Blueprints and Design Documents

- Make Kuryr CNI and controller installable via kubeadm: https://blueprints.launchpad.net/kuryr-kubernetes/+spec/kubeadminstallable

Code Contributions

- Add support to install Kuryr as a network addon: https://review.openstack.org/#/c/466675/
- CNI container: parametrize and clean up: https://review.openstack.org/#/c/490377/
- Add readiness probe to kuryr-controller pod: https://review.openstack.org/#/c/518502/
- Ensure readiness probe is not always enabled: https://review.openstack.org/#/c/523443/
- Create lockfiles directory in CNI Dockerfile: https://review.openstack.org/#/c/551261/
- cni health: track all cgroup memory usage: https://review.openstack.org/#/c/549330/

1.7 CI+Packaging

Blueprints and Design Documents

- Migrate gates to ZuulV3.

Code Contributions

- https://review.openstack.org/#q,topic:zuulv3-native-gates,n,z
- Creating gate for testing ports pool feature
- Kuryr-kubernetes
  - Do not treat warnings as errors in build_sphinx
  - Add cni_ds_init to kuryr-kubernetes distgit
  - Add python-oslo-cache as dependency and BR
  - Add kuryr CNI Daemon support
  - Moved package description to global variable and replace with it.
  - convert individual requires to macro
  - Update to 0.2.0 pike-rdo
  - Requirements sync for pike
  - Use explicit 'man' builder
  - switch from oslosphinx to openstackdocstheme
  - Add python-oslo-reports requirement
  - Remove %clean section to remove buildroot
  - Fix config file generation
  - Initial import of the spec file and sources rpm-master
- Kuryr-tempest-plugin
  - Add initial spec for kuryr tempest plugin
  - Requirements sync for pike
  - Update to 0.1.0 pike-rdo
  - Requirements sync for Queens release
- Kuryr-libnetwork
  - Initial import of the spec file
  - apply review improvements

- switch from oslosphinx to openstackdocstheme
- Fix unit tests, fix python3 requirements, fix doc building
- Replace rm -rf requirements.txt step with proper rpm macro
- Remove bogus python3 requirements
- Move git and openstack-macros BR to top level

1.8 Testing (kuryr-tempest-plugin)

Blueprints and Design Documents

- Functional testing enhancement

Code Contributions

- https://review.openstack.org/#q,topic:pod_icmp,n,z
- Testing pod to pod connectivity
- Testing port pool feature
- Check connectivity from Pod to VM
- Add scenario test manager
- Add a devstack plugin for Kuryr Tempest Plugin
- Merge create and list pods tests
- Migrate kuryr-tempest-plugin to zuulv3 syntax

2.1.3 OpenShift-Ansible

We have finally worked on several modifications at different project to enable Kuryr support at the OpenShift-Ansible installation tool, to enable an easy way of consuming the previously implemented features. The next covers the main work items performed within the Superfluidity project.

1.1 OpenStack SDK

- Add reno for tag support on heat stacks
- Add tag support to create_stack

1.2 Shade

- Add tag support to create_stack

- Add supported method for checking the network exts: https://review.openstack.org/#/c/529360/

1.3 Ansible

- Add support for heat stack metadata

1.4 Heat

- Fix bypass list length check when all values are None: https://review.openstack.org/#/c/530942/

1.5 OpenShift-Ansible

- openstack: make server ports be trunk ports: https://github.com/openshift/openshift-ansible/commit/1cf6275b983a108a02b6ef178fe35e610162b963
- Initial Kuryr support: https://github.com/openshift/openshift-ansible/commit/e6ea6839a8f657c1266c25ba4aba43c837329fa3
- Configure Kuryr CNI daemon: https://github.com/openshift/openshift-ansible/commit/d515d4542f648194effb5be242d95f2d4834d7de
- Kuryr var generation in OSt dynamic inventory: https://github.com/openshift/openshift-ansible/commit/ef4a49ebe5470281b81ef48ef05839fbb523bb13
- Initial Kuryr Ports Pool Support: https://github.com/openshift/openshift-ansible/commit/d6fa97f7c333b90034d57e780bee3fdbc57acef7
- Add support to pre-create subports at each trunk: https://github.com/openshift/openshift-ansible/commit/a6ad755628dcb70f738cf34db9253b99f2bec7e1
- Enabling multi vif pool drivers: https://github.com/openshift/openshift-ansible/commit/93dddb8809ea30bcd0be1ebe5f9e9159015e60ea
- Add Kuryr documentation: https://github.com/openshift/openshift-ansible/commit/d29f4e42e11af8a81864398d4a0393b2a0288584
- Add readiness probe to kuryr controller pod: https://github.com/openshift/openshift-ansible/commit/af74b50d3a6d6ea420c0892ef76eca09ce60fcef

2.2 NOKIABLF

2.2.1 Open Air Interface

As part of the Cloud RAN research, Nokia BLF is using the OpenAirInterface (www.openairinterface.org) software which is an open source platform implementing a standard-like LTE protocols of the radio interface between a MN (Mobile Node), also referred as UE (User Equipment), and a Base Station (eNodeB). The MN and eNodeB code is implemented on PC platforms running Linux. All the processing of the LTE physical layer is done in software on the PC. New 5G features start to be supported like a new waveform Filtered OFDM. Thanks to this software capability, Nokia BLF selected the choice to use OAI for Cloud RAN research facilitating the virtualisation and the cloudification work.

Nokia BLF is working on splitting the OAI into a set of functions following a split design (PHY-PHY, PHY-MAC, RLC-PDCP). This split is the cornerstone cloud RAN facilitating the shift of functions from the cloud to the remote antenna and vice-versa according to the service requirements.

Aware that the 5G ecosystem is moving very fast, Nokia BLF joined recently the OAI alliance which aims to provide a similar ecosystem for the core (EPC) and access-network (EUTRAN) of 3GPP cellular systems with the possibility of interoperating with closed-source equipment in either portion of the network. In the context of the evolutionary path towards 5G, there is clearly the need for open-source tools to ensure a common R&D and prototyping framework for rapid proof-of-concept designs.

2.2.2 OAI/Code contributions:

During the last period, Nokia BLF worked on bug fixing and improving the functions library.

- Configuration procedure enhancements: bug fixing accepted in Nov 2017. The current configuration of Open Air Interface (OAI) is using libconfig and is managed by each OAI component. Each OAI module needing configuration parameters is using libconfig APIs. This part is re-written by developing a configuration module, which will perform the libconfig calls and offer higher level APIs to the OAI modules.
- Improvement in the user interface (telnet server): code contribution, accepted in Oct 2017. It includes a light debugging and monitoring tool in OAI.

- OAI functions in shared library: accepted in Nov 2017. The contribution is related to OAI modularity enhancement. It allows an easy replacement of the specified functions by Nokia proprietary versions. The first targeted functions are the turbo decoder/encoder, the scheduler, the FFT/IFFT.
- Multi-UE support: Bug fixing submitted in Dec 2017
- L1 threads managements: Bug fixing accepted in Jan 2018.
- Nokia BLF participate in the evaluation and the test campaign related to NB-IoT feature of OAI. The campaign is still on-going.

2.3 NOKIAIL

2.3.1 Open Source Contributions

NOKIA IL is working in the open source community to advance NFV in general and the concept of SUPERFLUIDITY in particular. Specifically, NOKIA IL contributes mostly to OpenStack and to OPNFV.

2.3.1.1 Open Stack Contributions

NOKIA IL contributes to three main projects within OpenStack, namely, (i) Mistral, (ii) Heat, and (iii) Vitrage.

Mistral is a key component in SUPERFLUIDITY Generic VNFM, enabling efficient life cycle management of the various VNFs and MEC applications. Thus, NOKIA IL is highly involved in this project, and indeed recently Mistral PTL, Renat Akhmerov became a CloudBand (NOKIA IL) employee.

Within Mistral, we designed and developed many features that are important for NFV use cases such as supporting large datasets and expiration policies. We found and fixed issues that occurred when running Mistral in full HA mode – a must in a Telco grade application and the default topology we use in CloudBand node. Additionally, CloudBand created the Mistral puppet module installation and contributed it as a whole back to the community. RedHat are now using this project in order to install Mistral as part of their OpenStack distribution.

HOT, the Heat template, is considered as a leading deployment description language, when it comes to virtualised resource managed by OpenStack. Accordingly, we introduced NFV use cases to the core team in Heat as early as the Kilo release. At that point, OpenStack did not consider NFV use cases as valid and our discussions helped shaping key features in Heat (e.g., we prevented ResourceGroup index variable removal that is important for VNFs in order to support personalisation like slotId). More recently, we found, reported and fixed many issues that concern VNF use cases starting from backward compatibility issues, like Heat template fail from Kilo release on validation, and performance to performance issues on massive loads, like missing index on stack.owner_id. Furthermore, we designed and contributed to resource types that are important for the Mistral based VNFM like the OS::Mistral::Workflow.

Last but not least, NOKIA IL initiated the Vitrage project, which is now under the OpenStack big tent, to allow collection on infrastructure data correlate the data and perform Root Cause Analysis. Recent work of Vitrage focused on: (i) connecting more sources to Vitrage, and (ii) code refactoring to support HA. For example, NOKIA IL added CollectD of Intel as another input source for Vitrage. Additionally, the refactoring work was mainly around the dB and the multi jobs handling.

2.3.1.2 OPNFV

In addition to OpenStack, NOKIA IL further works with OPNFV to support its work and disseminate NOKIA IL and SUPERFLUIDITY concepts. Here, for now we mostly contribute to Doctor, where we have Vitrage as a reference implementation for this project.

2.3.2 ONAP

While the team in Israel was not directly involved in ONAP, NOKIA IL was in close contact with its delegate to the technical steering committee (Ranny Haiby). Specifically, we pushed with Ranny a better alignment of ONAP with ETSI NFV and particularly, with the support for generic VNFM based on workflow engine. This concept is aligned with Superfluidity architecture and its use of Mistral as a workflow engine for LCM. Additionally, following the concept of RFB and the push for support of both VMs and containers by Superfluidity (e.g., Kuryr), Ranny was advocating in ONAP for adopting Cloud Native concepts and supporting containers.

2.4 INTEL

2.4.1 Open Source Contributions

Intel open sourced a new version of its infrastructure landscaper, which was used in Tasks 4.1, 4.2 and Task 7.2.

(https://github.com/IntelLabsEurope/landscaper). The landscaper provides the ability to generate graph-based view of the relationship between physical, virtual and service resources in a cloud based infrastructure environment. The landscaper also support dynamic updating of the landscape view as new entities such as VM's are added or removed from the landscape.

Intel has also open sourced a contribution to snap the telemetry framework in the form of a new plugin which collects Linux scheduler statistics from /proc/schedstat (https://github.com/intelsdi-x/snap-plugin-collector-schedstat).

2.5 CNIT

2.5.1 Open Source Contributions

CNIT has released an open source project called RDCL 3D, which is the result of the Superfluidity activity. RDCL 3D is a web framework for the design of NFV services and components. The framework allows editing, validating, visualizing the descriptors of services and components both textually and graphically.

The platform is designed with a modular approach, allowing developers to "plug in" the support for new models (project types). Currently supported project types are:

- ETSI Release 2 NS and VNF descriptors
- TOSCA Simple Profile for NFV
- Click modular router configurations
- Superfluidity-ETSI (ETSI R2 + Click)

The RDCL 3D source code is available on GitHub (https://github.com/superfluidity/RDCL3D) and it has been released under the Apache 2.0 License.

2.6 Unified Streaming

2.6.1 Future Plans for Open Source Contributions

Maintenance and development MPEG PCC Software platform (https://github.com/RufaelDev/pcc-mp3dg/).

2.7 ULG

2.7.1 Open Source Contributions

Improvements to Click[1], netmap[2] and FastClick[3] (our own extension of Click).

1: https://github.com/kohler/click

2: https://github.com/luigirizzo/netmap

3: http://fastclick.run.montefiore.ulg.ac.be/

2.8 OnApp

2.8.1 Open Source Contributions

OnApp had contributed the SIM (Superfluidity Information Model) model that takes the concepts from the three other models used in Superfuidity (RDCL3D, CRAN deployment, Landscaper suite) as Open Source Contributions (https://github.com/superfluidity/sim).

2.8.2 Future Plans for Open Source Contributions

OnApp had been developing the Microvisor (a scalable hypervisor architecture for microserver) for a while. OnApp has interaction with the Xen project and is planning for open-source contributions of their Microvisor to the Xen project. OnApp is also planning to contribute some OpenStack components as parts of the Microvisor to the OpenStack community.

2.9 UPB

2.9.1 Open Source Contributions

UPB has developed Symnet as open-source in SUPERFLUIDITY. The source-code of Symnet, including the translators from FIB snapshots and Click modular router configurations are open-source at https://github.com/nets-cs-pub-ro/Symnet/. UPB has also developed Beamer, a high-performance load balancer that has been published in NSDI 18; Beamer is available as open-source at https://github.com/Beamer-LB/beamer-p4

2.10 CITRIX

2.10.1 Open Source Contributions

CITRIX has implemented two Ansible modules for NetScaler, which were contributed as open source (https://github.com/citrix/netscaler-ansible-modules):

- netscaler_server: automates and manages the configuration of NetScaler servers (http://docs.ansible.com/ansible/latest/netscaler_server_module.html)
- netscaler_service: automates creation and manages the configuration of NetScaler services (http://docs.ansible.com/ansible/latest/netscaler_service_module.html)

2.11 NEC

2.11.1 Open Source Contributions

NEC has released the bulk of the code produced in the Superfluidity project as open source, always under a commercially friendly 3-clause BSD license. In particular, the following software is now open source:

- **LightVM:** The bulk of the work described in the SOSP 2017 paper has been released as open source. The only missing part is Tinyx, which we hadn't had the time to clean up properly for release. Having said that, we have now found a partner university that is doing that work, with views towards releasing even this part as open source. More information about LightVM along with the source code can be found at http://sysml.neclab.eu/projects/lightvm/.

- **HyperNF:** All code described in the SoCC 2017 paper is now open source and can be found at http://sysml.neclab.eu/projects/hypernf/.

- **Minicache:** The code for this high-performance CDN cache unikernel, along with its accompanying VEE 2017 paper, can be found at http://sysml.neclab.eu/projects/minicache.

- **Unikraft:** Last but not least, the Unikraft tool for automated building of unikernels is also open source as a Xen Project incubator under the auspices of the Linux Foundation. For more information, as well as access to the source code, please refer to https://www.xenproject.org/developers/teams/unikraft.html.

3 Conclusion

As reported by this deliverable, the SUPERFLUIDITY project has been quite successful regarding standardization and open source contributions, with a remarkable impact in some communities, such as Kuryr OpenStack project.

4 References

[1] OpenStack code review system https://review.openstack.org/

[2] Mobile Edge Computing (MEC) Framework and Reference Architecture https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=46046

[3] Open Source MANO https://osm.etsi.org/

[4] OpenDaylight https://www.opendaylight.org/