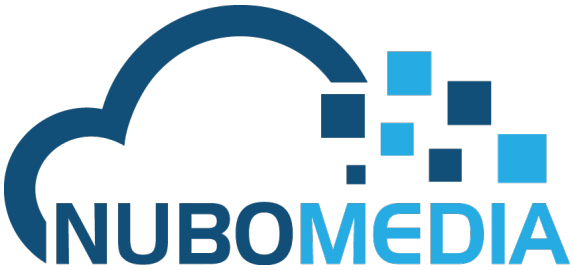
D2.2.2	
Version	2.2
Author	URJC
Dissemination	CO
Date	31/01/2016
Status	Final



D2.2: State-of-the-art revision document v2

Project acronym:	NUBOMEDIA
Project title:	NUBOMEDIA: an elastic Platform as a Service (PaaS) cloud for interactive social multimedia
Project duration:	2014-02-01 to 2017-01-30
Project type:	STREP
Project reference:	610576
Project web page:	http://www.nubomedia.eu
Work package	WP2
WP leader	Victor Hidalgo
Deliverable nature:	Report
Lead editor:	Luis Lopez
Planned delivery date	01/2016
Actual delivery date	31/01/2016
Keywords	State-of-the-art revision

The research leading to these results has been funded by the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement nº 610576

FP7 ICT-2013.1.6. Connected and Social Media

DISCLAIMER

Contributors:

URJC
LIVEU
VTOOLS
FRAUNHOFER
NAEVATEC
VTT
USV
ZED
TUB
TI

Internal Reviewer(s):

Constantin Filote (USV)
Noam Amram (LIVEU)

Version History

Version	Date	Authors	Comments
0.1	01-04-2014	Luis Lopez	Initial Version
	28-05-2014	Constantin Filote	First Reviewed
0.2	24-09-2014	Luis Lopez	Additional topics added.
0.3	15-11-2014	Luis Lopez	Additional topics added
0.4	26-12-2014	Luis Lopez	Additional topics added
1.0	22-01-2015	Luis Lopez	Added final contributions from partners
1.1	21-04-2015	Luis Lopez	Removed all previous information and restarted following review 1 outcome.
2.0	21-09-2015	Luis Lopez	Crated new SotA structure
2.1	20-01-2016	Luis Lopez	Integrated contributions from partners.
2.2	23-01-2016	Luis Lopez	Integrated FRAUNHOFER contribution

Table of contents

1 Executive summary 9
2 Introduction 9
3 Cloud technologies for advanced media communications 9
3.1 Cloud infrastructures for real-time media 9
3.1.1 Description of current SoTA 9
3.1.2 NUBOMEDIA approach beyond SotA 13
3.1.3 NUBOMEDIA outcomes 13
3.1.4 References 13
3.2 Orchestration and Management of Real-Time Network Functions with guaranteed QoS14
3.2.1 Description of current SoTA 15
3.2.2 NUBOMEDIA approach beyond SotA 28
3.2.3 NUBOMEDIA outcomes 32
3.2.4 References 33
3.3 PaaS for Real-Time Multimedia Applications 33
3.3.1 Description of current SoTA 39
3.3.2 NUBOMEDIA approach beyond SotA 47
3.3.3 NUBOMEDIA outcomes 49
3.3.4 References 49
3.4 Media monitoring in cloud infrastructures 50
3.4.1 Description of current SoTA 51
3.4.2 NUBOMEDIA approach beyond SotA 55
3.4.3 NUBOMEDIA outcomes 56
3.4.4 References 56
3.5 Deploying and installing media cloud infrastructures 56
3.5.1 Description of current SoTA 57
3.5.2 NUBOMEDIA approach beyond SoTA 60
3.5.3 NUBOMEDIA outcomes 60
3.5.4 References 60
4 RTC media server technologies 61
4.1 RTC media servers 61
4.1.1 RTC media servers: an overview 61
4.1.2 Description of current scientific and engineering SoTA 65
4.1.3 NUBOMEDIA approach beyond SotA 71
4.1.4 NUBOMEDIA outcomes 72
4.1.5 References 74
4.2 Real-time Video Content Analysis on the cloud 77
4.2.1 Description of current SoTA 79
4.2.2 NUBOMEDIA approach beyond SotA 86
4.2.3 NUBOMEDIA outcomes 87
4.2.4 References 88
4.3 Augmented Reality capabilities on real-time media servers 89
4.3.1 Description of current SoTA 89
4.3.2 NUBOMEDIA approach beyond SotA 93
4.3.3 NUBOMEDIA outcomes 95
4.3.4 References 95
4.4 Interoperability on real-time media infrastructures servers 95
4.4.2 NUBOMEDIA approach beyond SotA 101
4.4.3 NUBOMEDIA outcomes 102
4.4.4 References 102
4.5 Cloud APIs for accessing Media Servers 102

4.5.1 Related work *104*
4.5.2 NUBOMEDIA approach beyond SotA *106*
4.5.3 NUBOMEDIA outcomes *109*
4.5.4 References *111*
4.6 Real-time media APIs in smartphone platforms 113
4.6.1 Description of current SoTA *113*
4.6.2 NUBOMEDIA approach beyond SotA *116*
4.6.3 NUBOMEDIA outcomes *117*
4.6.4 References *117*
4.7 Cloud Videoconferencing APIs 117
4.7.1 Description of current SotA *118*
4.7.2 NUBOMEDIA approach beyond SotA *120*
4.7.3 NUBOMEDIA outcomes *121*
4.7.4 References *121*
4.8 Enhancing real-time media developer efficiency 122
4.8.1 Description of current SoTA *122*
4.8.2 NUBOMEDIA approach beyond SotA *124*
4.8.3 NUBOMEDIA outcomes *124*
4.8.4 References *124*
5 Real-time media in vertical segments 125
5.1 Real-time media for video surveillance and security 125
5.1.1 Description of current SoTA *129*
5.1.2 NUBOMEDIA approach beyond SotA *135*
5.1.3 NUBOMEDIA outcomes *136*
5.1.4 References *136*
5.2 Real-time media for news reporting 137
5.2.1 Description of current SoTA *137*
5.2.2 NUBOMEDIA approach beyond SotA *139*
5.2.3 NUBOMEDIA outcomes *141*
5.2.4 References *141*
5.3 Real-time media on e-health environments 141
5.3.1 Description of current SoTA *141*
5.3.2 NUBOMEDIA approach beyond SotA *147*
5.3.3 NUBOMEDIA outcomes *147*
5.3.4 References *147*
5.4 Real-time media on games 149
5.4.1 Description of current SoTA *150*
5.4.2 NUBOMEDIA approach beyond SotA *154*
5.4.3 NUBOMEDIA outcomes *155*
5.4.4 References *155*
5.5 Real-time media for social TV 156
5.5.1 NUBOMEDIA approach beyond SotA *157*
5.5.2 NUBOMEDIA outcomes *157*
5.5.3 References *157*

List of Figures

Figure 1 Feature comparison of IaaS frameworks [LASZEWSKI2012] - indicates a positive evaluation (the more checkmarks the better). 10
Figure 2 Feature comparisons between OpenStack and other FOSS for IaaS 11
Figure 3 A comparison chart between Xen, KVM, VirtualBox, and VMWare ESX [YOUNGE2011]. 12
Figure 4 IBM's results for Docker vs. KVM results running Linpack on two sockets with 16 cores. Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained overall runs. 12
Figure 5. ETSI NFV Architecture 15
Figure 6. OpenMANO Architecture 16
Figure 7. High level overview of the Tacker focus on the NFV Architecture 17
Figure 8. Detailed architectural overview of the Tacker project 18
Figure 9. Achitectural overview of Cloudify 20
Figure 10. Mapping of Juju to NFV Architecture 22
Figure 11. Service Manager (SM) internal architecture 23
Figure 12. Service Orchestrator (SO) internal architecture 24
Figure 13. Cloud Controller (CC) internal architecture 24
Figure 14. High-level view of overall T-NOVA System Architecture 26
Figure 15. T-NOVA Orchestrator platform, modules and itnerfaces 26
Figure 16. 5G Exchange architectural approach 28
Figure 17. High-level Architecture of OpenBaton 31
Figure 18. High-level Architecture of OpenBaton and the Generic VNFM inlcuding the EMS 32
Figure 19 NIST PaaS Reference Architecture 34
Figure 20 Gartner Refernece Archiecture for PaaS 35
Figure 21 Forrester Research PaaS Reference Architecture 36
Figure 22 Cloud Foundry Components Overview 39
Figure 23 OpenShift Origin Architecture Overview 41
Figure 24Google App Engine High Level Overview 43
Figure 25 Solutions Review report on 2016 Comparison Matrix Report 45
Figure 26 InfluxDB design 52
Figure 27. InfluxDB Web Interface for management 52
Figure 28. Graphite architecture 53
Figure 29. How Graphite components interact 54
Figure 30. Prometheus architecture 55
Figure 32. RTC applications, in general, and WebRTC applications, in particular, may use two different models. As shown at the top, the peer-to-peer model is based on direct communication among clients. This model provides minimum complexity and latency, but it also has important limitations. At the bottom, the infrastructure-mediated model, where a media server is mediating among the communicating clients. This model has higher latency and complexity, but it makes possible to enrich RTC services with additional capabilities such as transcoding (i.e. interoperability), efficient group communications (i.e. MCU or SFU models), recoding and media processing. 62
Figure 33: Media capabilities provided by state-of-the-art media server include: transcoding (top), group communications (middle) and archiving (bottom) 64
Figure 34: The most popular Computer Vision libraries 80
Figure 35: Microsoft Project Oxford 82
Figure 36: CloudCV Architecture 84
Figure 37: CloudCV backend 85
Figure 38 IMS Layered Architecture 97
Figure 39 WebRTC integration with IMS user agent and data repository 98
Figure 40 webRTC linking to IMS via NNI 98
Figure 42 Comparing SIP proxy with SIP-webRTC Gateway 101
Figure 42 Review of VCF tools [GASPARINI2013] 119
Figure 43: Video Surveillance industry segments 126
Figure 44: Analog Video Surveillance System Architecture 127
Figure 45: IP Video Surveillance System Architecture 128
Figure 46: Analog vs IP cameras 128
Figure 47. Video Surveillance as a Service (VSaaS) 129
Figure 48: Next (left side) and Axis cameras (right side) 130
Figure 49: iSpy FOSS solution 133
Figure 50: Framework for a Cloud-Based Multimedia Surveillance system 135

1 Executive summary

This document contains a revision on SotA (State-of-the-Art) on all technological areas of interest of the project providing, for each of them, the following information:

- The current technological status including shortcomings and limitations.
- How the project is planning to advance them.
- The expected results including how these advance in the scientific and technical domains.

2 Introduction

NUBOMEDIA a cloud platform specifically designed for hosting real-time interactive multimedia services. For this, in this project, we have created a number of technological enablers making possible for developers to use different PaaS APIs for creating media applications leveraging advanced capabilities which include WebRTC and RTP communications, media transcoding, media mixing, media routing and advanced media processing with features such as Video Content Analysis and Augmented Reality.

As it can be inferred from this description, NUBOMEDIA is a quite ambitious project that aims at evolving current SotA (State-of-the-Art) in many technological domains. For the sake of simplicity, in this document we concentrate on the ones which have more relevance for the project. Due to this, the document is organized in the following sections:

- Cloud technologies for advanced media communications: this section is devoted to analyzing SotA status on the area of cloud computing infrastructures and orchestration: the ones having more relevance for NUBOMEDIA.
- Media server technologies: in this section we review current efforts for creating Real-Time Communications media servers and their development APIs.
- Real-time media in vertical segments: This section comprises all the specific vertical domains where NUBOMEDIA partners are involved. For every of them we specify current trends in the market analyzing the features and limitations of the most popular solutions.

For every of these sections, we specify the NUBOMEDIA approach for overcoming current SotA as well as the expected outcomes generated by the project.

3 Cloud technologies for advanced media communications

3.1 Cloud infrastructures for real-time media

3.1.1 Description of current SoTA

Regarding IaaS cloud infrastructures, in current state-of-the-art we can find different solutions.

Commercial IaaS solutions for - virtual infrastructure

- VMware vSphere - https://www.vmware.com/products/vsphere
  - Is a commercial solution from VMware for cloud computing and is targeted mostly to enterprise companies.
- Microsoft System Center - http://www.microsoft.com/en-us/server-cloud/products/system-center-2012-r2/
- Oracle Cloud IaaS - https://www.oracle.com/cloud/iaas.html

These solutions have a number of limitations:

- They are expensive in economic terms.
- Limited capability in integrating with third party solutions
- Closed source, meaning you can not modify the platform to fulfill any special needs

FOSS (free open source solutions):

- OpenNebula
- Apache CloudStack
- OpenStack
- Eucalyptus 2.0
- Nimbus

In this case, the limitations include:

- Lack of sutable paid / professional support for some of them.

For NUBOMEDIA objectives, FOSS solutions are more relevant. The following tables show a comparison among them.

	OpenStack	Eucalyptus 2.0	Nimbus	OpenNebula
Interfaces	EC2 and S3, Rest Interface. Working on OCCI ✓✓	EC2 and S3, Rest Interface. Working on OCCI ✓✓	EC2 and S3, Rest Interface ✓	Native XML/RPC, EC2 and S3, OCCI, Rest Interface ✓✓✓
Hypervisor	KVM, XEN, VMware Vsphere, LXC, UML and MS HyperV ✓✓✓	KVM and XEN. VMWare in the enterprise edition. ✓✓	KVM and XEN ✓	KVM, XEN and VMWare ✓✓
Networking	- Two modes: (a) Flat networking (b) VLAN networking -Creates Bridges automatically -Uses IP forwarding for public IP -VMs only have private IPs ✓✓✓	- Four modes: (a) managed; (b) managed-novLAN; (c) system; and (d) static - In (a) & (b) bridges are created automatically - IP forwarding for public IP -VMs only have private IPs ✓✓✓	- IP assigned using a DHCP server that can be configured in two ways. - Bridges must exists in the compute nodes ✓✓	- Networks can be defined to support Ebtable, Open vSwitch and 802.1Q tagging -Bridges must exists in the compute nodes -IP are setup inside VM ✓✓✓
Software deployment	- Software is composed by component that can be placed in different machines. - Compute nodes need to install OpenStack software ✓	- Software is composed by component that can be placed in different machines. - Compute nodes need to install OpenStack software ✓	Software is installed in frontend and compute nodes ✓✓	Software is installed in frontend ✓✓✓
DevOps deployment	Chef, Crowbar, Puppet ✓✓✓	Chef*, Puppet* ✓ (*according to vendor)	no	Chef, Puppet ✓✓
Storage (Image Transference)	- Swift (http/s) - Unix filesystem (ssh) ✓	Walrus (http/s) ✓	Cumulus (http/https) ✓	Unix Filesystem (ssh, shared filesystem or LVM with CoW) ✓
Authentication	X509 credentials, LDAP ✓✓✓	X509 credentials ✓	X509 credentials, Grids ✓✓	X509 credential, ssh rsa keypair, password, LDAP ✓✓✓
Avg. Release Frequency	<4month	>4 month ✓	<4 month	>6 month ✓
License	OpenSource - Apache ✓	OpenSource ≠ Commercial	OpenSource Apache ✓	OpenSource Apache ✓

Figure 1 Feature comparison of IaaS frameworks [LASZEWSKI2012] - indicates a positive evaluation (the more checkmarks the better).

	Eucalyptus	OPenNebula	Abicloud	Nimbus
Cloud type	Public	Private	Public/private	Private
Scalability	Scalable	Dynamical, scalable	Scalable	Dynamical, scalable
Cloud form	IaaS	IaaS	IaaS	IaaS
Compatibility	Support EC2, S3	Open, multi- platform	Not supported EC2	Support EC2
Deployment	Dynamical deployment	Dynamical deployment	Pack and redeploy	Dynamical deployment
Deployment manner	Commandline	Commandline	Web interface drag	Commandline
Transplant ability	Common	Common	Easy	Common
Hypervisors support	VMware, Xen, KVM	Xen, VMware, KVM	Virtualbox, Xen,VMware,KVM	Xen, KVM
Web interface	Web service	Libvirt,EC2, OCCI,API	Libvirt	EC2 WSDL WSRF
Structure	Module	Module	Open platform, encapsulate core	Lightweight components
Reliability	--	Rollback host and VM	--	---
OS support	Linux	Linux	Linux	Linux
Development language	java	java	Ruby, C++, python	Java, python
Security	Public/Private key authentication	Authen.password, RSA, SSH, LDAP,	Code Access System(CAS)	PKI
VM build	unavailable	unavailable	Drag & drop	unavailable
OS Licence	BSD Licence	Apache version 2 licence	Common Public Attribution Licence	Apache version 2 licence

Figure 2 Feature comparisons between OpenStack and other FOSS for IaaS

The NUBOMEDIA project is particularly interested in OpenStack, which is used as IaaS in the project. The following virtualization solutions are available for OpenStack: Xen, KVM and VMware ESXi.

	Xen	KVM	VirtualBox	VMWare
Para-virtualization	Yes	No	No	No
Full virtualization	Yes	Yes	Yes	Yes
Host CPU	x86, x86-64, IA-64	x86, x86-64, IA-64, PPC	x86, x86-64	x86, x86-64
Guest CPU	x86, x86-64, IA-64	x86, x86-64, IA-64, PPC	x86, x86-64	x86, x86-64
Host OS	Linux, UNIX	Linux	Windows, Linux, UNIX	Proprietary UNIX
Guest OS	Linux, Windows, UNIX	Linux, Windows, UNIX	Linux, Windows, UNIX	Linux, Windows, UNIX
VT-x / AMD-v	Optional	Required	Optional	Optional
Cores supported	128	16	32	8
Memory supported	4TB	4TB	16GB	64GB
3D Acceleration	Xen-GL	VMGL	Open-GL	Open-GL, DirectX

Live Migration	Yes	Yes	Yes	Yes
License	GPL	GPL	GPL/Proprietary	Proprietary

Figure 3 A comparison chart between Xen, KVM, VirtualBox, and VMWare ESX [YOUNGE2011].

One of the features that NUBOMEDIA requires is to have the instance boot time as small as possible in order to allow deployment of new media servers on demand whenever the number of users accessing applications increases rapidly. For all the previously enumerated solutions the boot time is somewhere between 2 and 4 minutes depending on the distribution type. However, recently a new technology has emerged and it is gaining more and more adopters: container. In particular, we are interested in Docker containers. Many research works demonstrate that Docker performs better than hypervisors such as KVM [IBM Research paper].

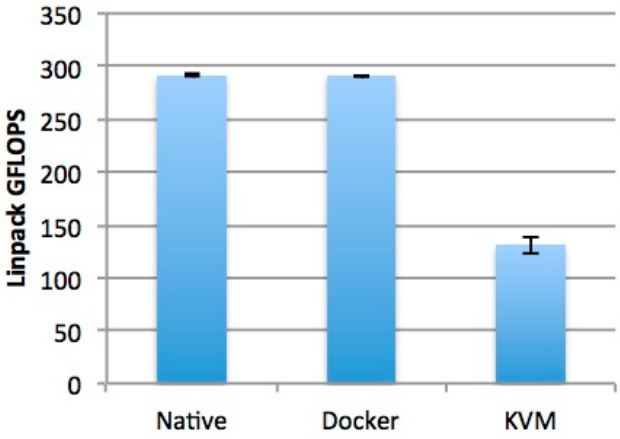


Figure 4 IBM's results for Docker vs. KVM results running Linpack on two sockets with 16 cores. Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained overall runs.

The Docker platform has announced the coming of new offerings: Docker Swarm [DOCKER/SWARM] and Docker Compose [DOCKER/COMPOSE]. When they are combined together, they may provide a solution to the service discovery problem. Docker Compose already provides a very limited service discovery mechanism [STUBBS], but it currently only works on a single host and it does not update as containers are stopped and restarted. It is unclear exactly what Docker Swarm will enable as details of that feature have yet to be announced.

Another platform that is using Docker technology is Agrave. Agave is a Science-as-a-Service platform, an ecosystem of open-source hosted developer APIs and tools for building science gateways [AGAVEAPI], [DOOLEY]. Agave is the backbone of NSF funded projects: NSF Plant Cyber-Infrastructure Program (DBI-0735191), NSF Plant Genome Research Program (IOS-1237931 and IOS-1237931), NSF Division of Biological Infrastructure (DBI-1262414), NSF Division of Advanced Cyber-Infrastructure (1127210), and the National Institute of Allergy and Infectious Diseases (1R01A1097403) such as the iPlant collaborative with over 15.000 users submitting thousands of jobs and moving over two petabytes of data every month. Production components of Agave run as Docker containers packaged up with Serfnode for service discovery. A Serfnode is a Docker image containing a serf agent and a supervisor instance [STUBBS]. Additionally, iPlant has open sourced the Agave deployer program, itself a Docker container, that interested parties can use to stand up their own Agave instance—nearly 40 containers in total - across multiple virtual machines with a single command. Users can bring their own apps as source code, binary code, VM images, or Docker images and mix and match them with apps from the catalog to create their own boutique pipelines they can save, reuse, and share. Because Agave manages

the end-to-end lifecycle of an application's execution, users are able to get out of the sysadmin business and think of their science as a service. The only configurations needed for the Agave deployer are the IP addresses.

3.1.2 NUBOMEDIA approach beyond SotA

Given the SotA status described above, the NUBOMEDIA progresses beyond SotA are more pragmatic than fundamental. Our main contribution was to enable Docker to work seamlessly inside OpenStack. In order to achieve this there are many alternatives, but only one making possible for virtual instances to run on Docker as a hypervisor: nova-docker.

Nova-docker is an alpha version hypervisor driver for OpenStack Nova Compute. It was first introduced within the Havana release, being ported up to the latest stable release. We have evolved nova-docker adding a patch to automatically support Docker images provisioning on all compute nodes running Docker as a hypervisor on OpenStack.

3.1.3 NUBOMEDIA outcomes

We enabled the possibility to have docker instances on top of the OpenStack compute nodes by implementing the nova-docker driver. A lot of missing documentation on the OpenStack page and the wiki page for nova-docker [nova-docker wiki] made the deployment not straightforward, but we documented all the needed steps enabling an easy and fast deployment of the Docker hypervisors by other IaaS. The instructions on how to properly deploy and configure nova-docker can be found on the WP6 deliverable.

We found a missing features for the nova-docker driver. The provisioning of container images that are added by users in the Glance image repository. Without this feature we couldn't have used KMS Docker containers on NUBOMEDIA because the IaaS was not able to start them on the compute nodes. In order to solve this problem we created a patch for the nova-docker which is written in python and uses the OpenStack APIs. The patch is available on a public github repository [nova-docker patch].

These outcomes shall be disseminated in the following way. First, we plan to make a pull request to the [nova-docker repository] and contribute it to the community to enable this capability directly in the hypervisor. Second, using the knowledge created on IaaS part, we are planning to publish a paper at the International Conference on Development and Application Systems 2016 [DAS conference 2016].

3.1.4 References

Websites

[nova-docker patch] https://github.com/usv-public/nubomedia-nova-docker
[nova-docker wiki] https://wiki.openstack.org/wiki/Docker ;
[nova-docker repository] https://github.com/openstack/nova-docker ;
[DAS conference 2016] http://www.dasconference.ro/ ;
[DOCKER/SWARM] https://github.com/docker/swarm/ ;
[DOCKER/COMPOSE] https://github.com/docker/compose/ ;
[AGAVEAPI] http://agaveapi.co/ ;
[CALINCIUC2013] Calinciuc, Alin, "OpenStack, the right solution for private cloud", 2013, http://academia.edu/10599168/OpenStack_the_right_solution_for_private_cloud ;

[CALINCIUC2014] Calinciuc, Alin; Spoiala, Cristian, "Docker - container virtualization and its impact in NUBOMEDIA", 2014, http://academia.edu/10599188/Docker_-_container_virtualization_and_its_impact_in_NUBOMEDIA .

Papers and books

[IBM Research paper] http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf ;
[STUBBS2015] Stubbs, Joe; Moreira, Walter; Dooley, Rion, "Distributed Systems of Microservices Using Docker and Serfnode", 7th International Workshop on Science Gateways, (2015): 34-39.
[DOOLEY2012] Dooley, R.; Vaughn, M.; Stanzione, D.; Terry, S., and Skidmore, E. "Software-as-a-Service: The iPlant foundation API" 5th IEEE Workshop on Many-Task Computing on Grids and Supercomputers, (2012), http://datasys.cs.iit.edu/events/MTAGS12/p07.pdf .
[ANDERSON2015] Anderson, Charles, "Docker", IEEE Software, MAY/JUNE, (2015): 102-105.
[LASZEWSKI2012] Laszewski, G. von; Diaz, J.; Wang, F. and Fox, G. C.,"Comparison of MultipleCloud Frameworks," IEEE Cloud 2012, Honolulu, HI, June 2012, (2012).
[YOUNGE2011] Younge, Andrew J.; Henschel, Robert; Brown, James T.; Laszewski, Gregor von; Qiu, Judy; Fox, Geoffrey C., "Analysis of Virtualization Technologies for High Performance Computing Environments", IEEE 4th International Conference on Cloud Computing CLOUD '11, (2011): 9-16.

3.2 Orchestration and Management of Real-Time Network Functions with guaranteed QoS

Network Function Virtualization (NFV) [ETSI_WP] considers the implementation of NFs as entities only in software that run over the Network Function Virtualization Infrastructure. Considering that Media Servers are intrinsically Network Functions, NUBOMEDIA followed the ETSI NFV specification for the Management and Orchestration (MANO) [MANO] of the media service components.

In Figure 5, it is shown the ETSI NFV architecture as described in the MANO specification. Our main focus in on the management and orchestration component highlighted by the red box. The Network Function Virtualization Orchestrator (NFVO) is the component managing the lifecycle of a Network Service (NS) composed by multiple MS and a Cloud Repository, while one or more Virtual Network Function Managers (VNFM) are in charge of the lifecycle of the VNF (refer to D2.4.2 and D3.2 for more details about the NUBOMEDIA Architecture and software implementation).

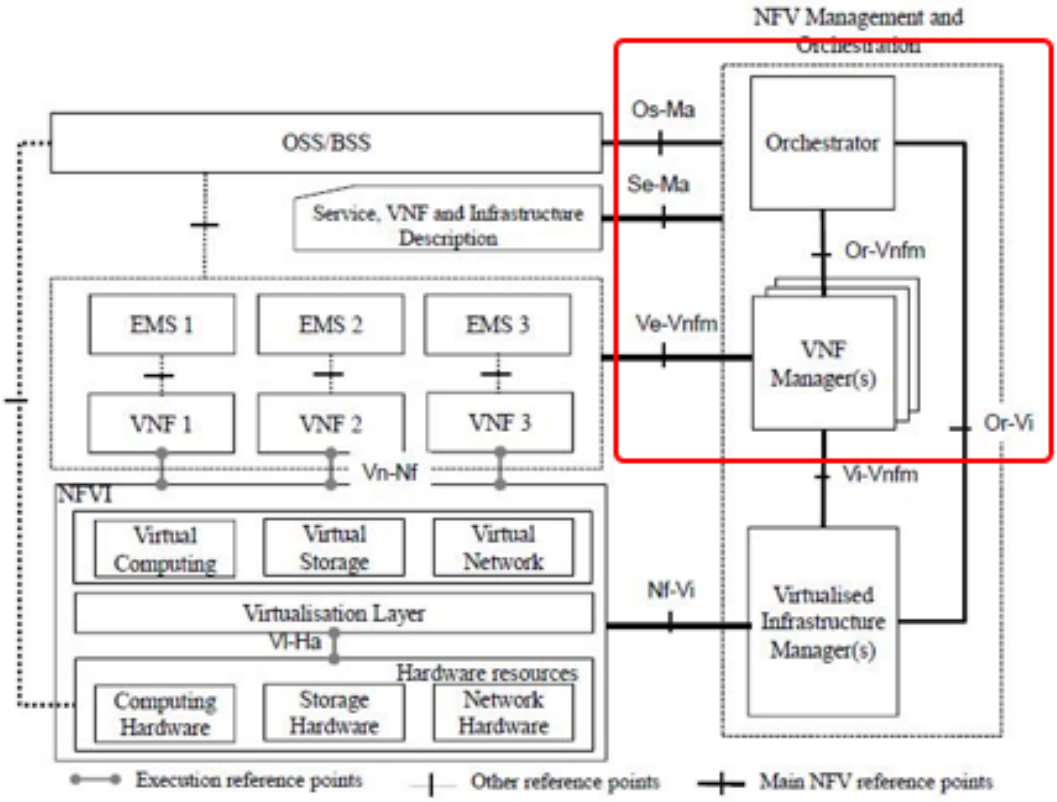


Figure 5. ETSI NFV Architecture

In order to properly identify the software components which can be used for implementing this functional elements, it is important to firstly identify the main functional requirements this tool should support.
This component must support:

- On-demand provisioning of Network Services as combination of multiple Network Functions.
- Composition of multiple Network Functions in a Network Service providing different capabilities.
- Broad Network Access so that those Network Services could be accessed from different locations.
- Monitoring of all the resources used by the different Network Functions.
- Mechanisms for providing network slicing, specifically providing different QoS requirements to multiple Network Services deployed on the NFV Infrastructure
- Autoscaling of VNFs.
- On demand scaling of VNFs.
- OpenStack integration.

3.2.1 Description of current SoTA

In this section are presented the current on-going activities on management and orchestration in cloud environment.

3.2.1.1 OpenMANO

OpenMANO [OPEN_MANO] represents an open source implementation of the ETSI NFV architecture. Figure 6 shows the mapping between the OpenMANO components and the ETSI NFV Architecture.

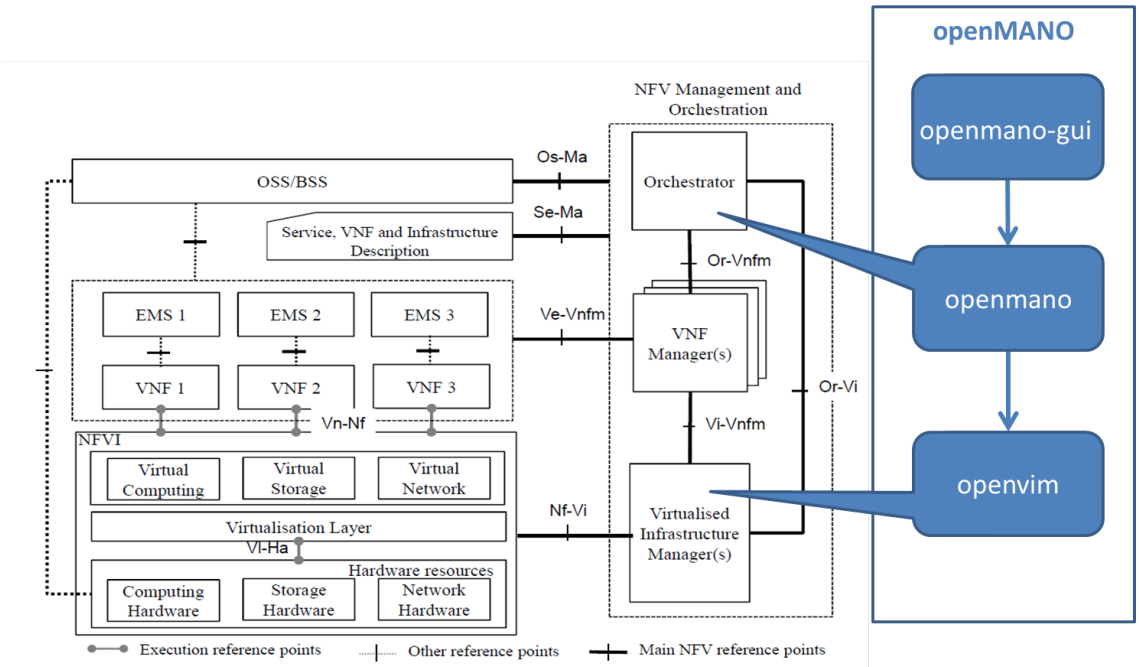


Figure 6. OpenMANO Architecture

OpenMANO follows an NFVO-centric approach what means that the NFVO is mainly responsible for interacting with the VIM by creating, deleting and managing instances, images, flavors and networks/links whereas the VNF lifecycle management is split between NFVO (VNF instantiation and termination) and VNFM. Apart from that, the VNFM is also in charge of VNF performance monitoring, fault management and event management.

Basically, OpenMANO consists of three main components:

- openmano: This component includes the reference implementation of an NFVO. It communicates with the VIM through its REST API and is in charge of managing NFV services by being responsible for the creation and deletion of VNF templates, VNF instances, network service templates and network service instances by interacting with the openvim. The openmano provides a northbound API for managing these service entities respectively.
- openmano-gui: The GUI, accessible via a web dashboard, interacts with openmano through its API to administrate termplates, services and instances.
- openvim: The openvim offers enhanced cloud services to openmano through a REST API including the creation, deletion and management of images, flavors, instances and networks in the NFVInfrastructure. The openvim interacts with an openflow controller for providing those network capabilities.

Since OpenMANO is a relatively new project, started in 2015, it is still under development. The first stable version (v0.3) for normal use was released in July 2015 supporting basic functionalities like deploying and terminating simple Network Services - called Network Scenarios in their terminology. This version was restricted to the usage of hypervisors and compute nodes controlled via libvirt whereas the Openflow switches were controlled by proactive rules (using floodlight). This brings complexity to the configuration and limitations as well:

- Hosts must be managed
- No identity management
- No concept of subnets
- Image uploading must be done manually by the end user
- Hardware open flow switch required for building a IaaS

The latest version (v0.4) was released at the end of 2015 supporting an additional Openflow controller, namely opendaylight. Additionally, it was integrated a Multi-VIM support for supporting Openstack as well. A first VNFM (GenericVNFM) implementation is already mentioned but still under development at this point in time. Therefore, it cannot support complex VNF management (complex lifecycle management and autoscaling), performance monitoring, fault management and event management.

3.2.1.2 Tacker

Tacker, launched in 2014, is an open source implemented OpenStack project building a Network Function Virtualization Orchestrator (NFVO) including an in-built general purpose Virtual Network Function Manager (VNFM) allowing the deployment and management of Virtual Network Functions (VNFs). Tacker is directly integrated into OpenStack and hence, it provides only this type of cloud environment. As mentioned before and shown in Figure 3, Tacker focuses on the reference implementation of an NFVO and a VNFM aligned to ETSI's NFV MANO specifications. This project is highly active with almost daily implementation activities and furthermore, it is documented using Blueprints with weekly-scheduled meetings summarized and published frequently on their website.

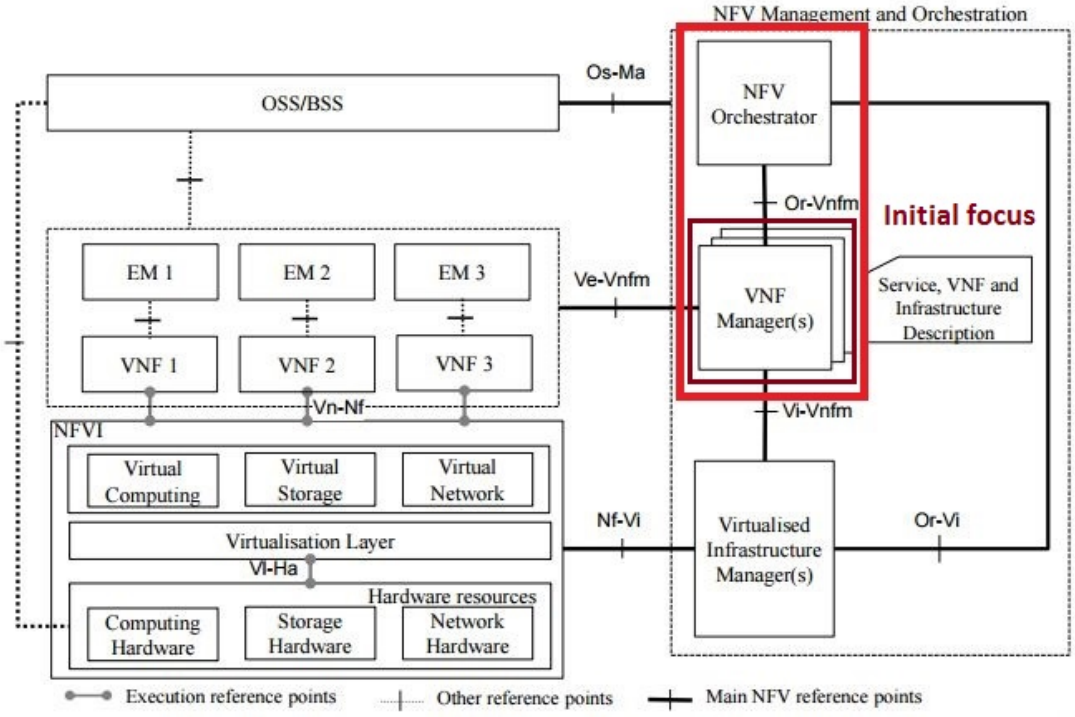


Figure 7. High level overview of the Tacker focus on the NFV Architecture

Figure 8 depicts a more detailed overview of Tacker's architecture, its components and relations between them. First of all, Tacker combines the NFVO and VNFM in a single component whereas, internally, the functionalities are clearly distinct as described below.

The NFVO is responsible for the high-level management of VNFs. Therefore, it makes use of a template-based end-to-end Network Service deployments able to compose several VNFs to a more complex NS. Furthermore, the NFVO is in charge of managing resources in the VIM – Vim Resource Checks and Resource Allocation. It allows to orchestrate VNFs across multiple VIMs using VNF placement policies to ensure efficient placement of VNFs. VNFs itself are connected through Service Function

Chaining (SFC) that are described in the VNF Forwarding Graph Descriptor (VNFFGD).

The VNFM, in turn, is in charge of managing groups of components that belongs to the same VNF instance controlling the basic lifecycle of a VNF (define, start, stop, undefined) and therefore, it facilitates the initial configuration. Additionally, it is also responsible for performance and health monitoring of already deployed VNFs. The VNFM uses these performance and health information for processing auto-healing and autoscaling where monitored parameters and corresponding actions are defined in policies. The Tacker VNF Catalogue, also maintained by the VNFM is basically the Repository of VNF Descriptors stored in the Tacker DB. Therefore, it provides an API to on-board, update and delete VNFDs. VNFDs can be defined additionally in TOSCA templates for describing VNF attributes like Glance image IDs, Nova properties (Placement, CPU Pinning, Numa policies, etc), Performance Monitoring policies, Auto-Healing policies and Auto-Scaling policies.

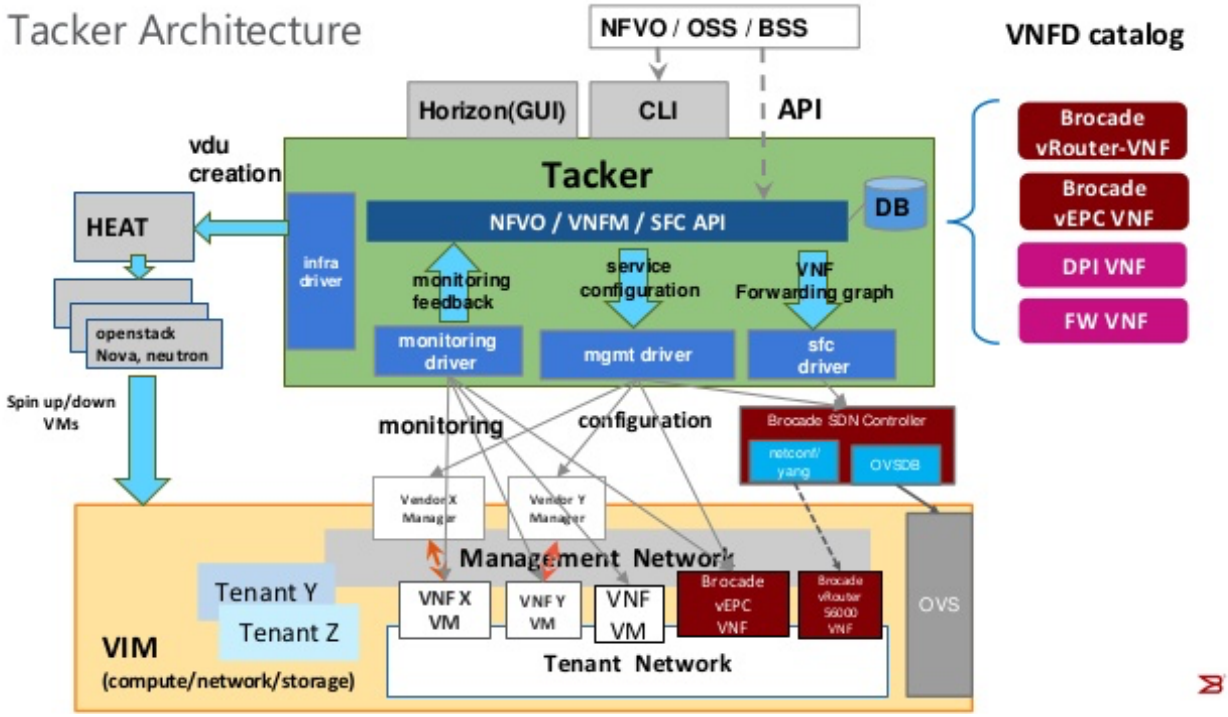


Figure 8. Detailed architectural overview of the Tacker project

Tacker provides several frameworks to facilitate configuration and management. An extendable Management Driver Framework is supported for VNF Auto Configuration that simplifies the VNF configuration based on Service selection. Injecting initial configurations can either be done through config-drive (special configuration drive that attaches to the instance when it boots) or custom management driver (connecting via ssh / REST API and apply configuration) also in active state. Another framework is the extendable Vendor and Service specific Health Monitoring Driver framework enabling Tacker to do health checks and healing (e.g. auto-restart on failure) once the VNF becomes ready. A similar framework is the extendable Vendor and Service specific Performance Monitoring Driver framework used for VNF autoscaling. Continuous performance monitoring according to KPIs described in the VNFD are used to processing VNF based autoscaling based on policies as well.

Since Tacker is a relatively new project where the implementation is at a very early stage, not all of the features mentioned are already supported. The following list shows next steps and the roadmap announced by the developers of Tacker:

- TOSCA NFV Profile support (using heat-translator)
- MANO API enhancements
- Enhanced Health Monitoring (framework, http-alive, etc)
- Auto Scaling support
- Support for NSD and VNFFG
- VNFFG to SFC mapping

Tacker uses Heat as the resource orchestration engine. The OpenStack Heat Orchestrator is completely integrated with OpenStack and is an utility able to manage multiple composite cloud applications using templates, over both an OpenStack-native ReST API and a CloudFormation-compatible Query API.

A Heat Orchestration Template (HOT) is sent to Heat in order to deploy a specific topology. The assignment of the Heat template is to define a topology infrastructure for a Cloud application, using a readable and writable way of representation. Heat templates typically take the shape of plain Yaml documents. Inside the HOT are also defined all the policies needed to enable the auto scaling. In particular, can be set policies regarding CPU and memory of each VM (Media Server in our specific case). There are also relationships between resources thus to infer a particular launch order on OpenStak. Those ones are likewise represented in the templates and Heat will follow the correct launching order.
The HOT template provides different fields:

- heat_template_version
  - This key indicates that the YAML document is a HOT template of the specified version.
- description
  - This optional key allows for giving a description of the template, or theworkload that can be deployed using the template.
- parameter_groups
  - This section allows for specifying how the input parameters should be grouped and the order to provide the parameters in. This section is optional and can be omitted when necessary.
- Parameters
  - This section allows for specifying input parameters that have to be provided when instantiating the template. The section is optional and can be omitted when no input is required.
- Resources
  - This section contains the declaration of the single resources of the template. This section with at least one resource should be defined in any HOT template, or the template would not really do anything when being instantiated.

In the resources field it is possible to define an auto scaling group that can scale arbitrary resources. The auto scaling system of Heat follows the concepts used by AWS

3.2.1.3 Cloudify

Cloudify [CLOUDIFY] is an enterprise-class open source Platform as a Service (PaaS) stack providing the full end-to-end lifecycle of NFV orchestration through a simple TOSCA-based YAML blueprint following a topology-driven and application-centric approach. As claimed by themselves: "*Cloudify is the only pure-play orchestration framework uniquely positioned to fit into heterogeneous enterprise and Telco*

environments". Therefore, many big Telcos have chosen Cloudify in the sense of being open source and modular in nature. Thanks to the "pluggability" it allows to make use of a various set of toolsets and environments including many plugins to interface, for instance, with peripheral network databases. Furthermore, it supports multiple clouds, data centers and availability zones.
The architecture of Cloudify, as depicted in Figure 9, consists of three main components:

- Manager (Orchestrator): The manager is a stateful orchestrator that deploys and manages applications (described in blueprints). The manager is in charge of running automation processes described in workflow scripts and forwards execution commands to the agents.
- Agents: Agents are responsible for executing commands based on manager's requests. In general, there is one agent per application deployment and optionally, an agent on each application VM. The manager side agents manage IaaS related tasks, like create a VM or network, assigning floating IPs, and allowing to execute tasks remotely via REST for example. As mentioned before, applications side agents are optionally and located on the application VM to install plugins and execute tasks locally.
- CLI client: The CLI client provides two main functions. It allows Manager Bootstraping for installing the manager with the preferred tool. Furthermore, it allows Managing Applications in order to simplify the deployment and management of applications including log and event browsing.

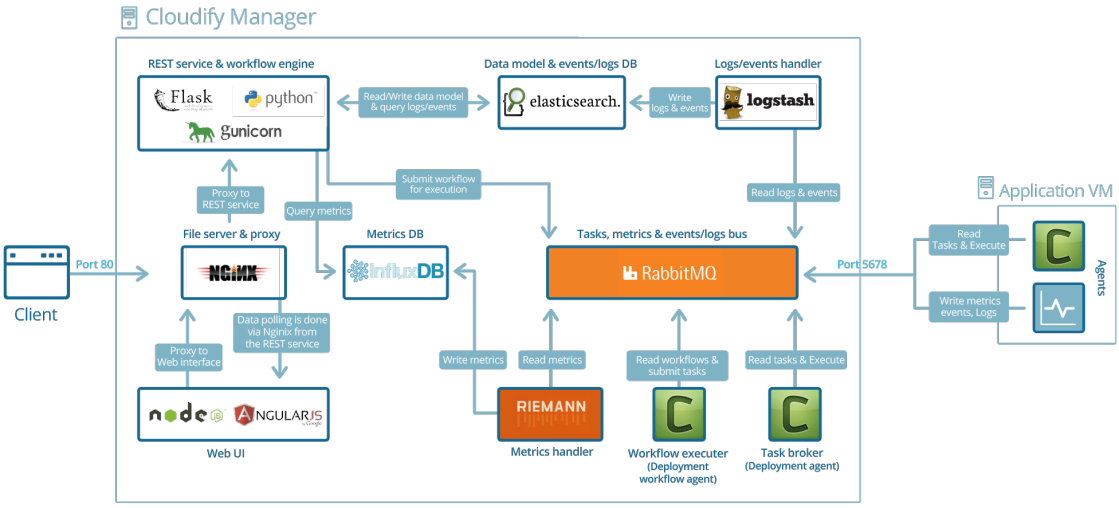


Figure 9. Achitectural overview of Cloudify

Cloudify lays between the application and the chosen CPI. Thanks to it, an application is able to focus on doing what it does best, it will be the task of Cloudify to manage the resources it needs and to make sure that they are available independently of which cloud and stack will be employed. Thanks to its design, Cloudify is able to offer some features, for instance, without any changes to the code, it is possible to move your application to the cloud, without any concerns about the application stack, database store or any other middleware components it uses. Therefore, no code has to be changed if you want to transfer your application to the Cloud. Moreover, Cloudify supports public clouds (Amazon EC2, Windows Azure, Rackspace, etc.) and private clouds (OpenStack, CloudStack, VMWare vCloud, Citrix XenServer, etc.)

The application is completely isolated from the underlying CPI to support enterprises that want to deploy the same application in multiple environments (for cloud bursting).

All the lifecycle of the application is managed by Cloudify's mechanism. Cloudify makes use of recipes in order to define an application, its services and their interdependencies, how to monitor, self-heal, scale them and their resources. So the process to deploy and manage an application results from:

1. Preparing the deployment
    a. Set up the cloud and describe your machines in the cloud driver
    b. Prepare the binaries required for your services
    c. Describe the application lifecycle and its services in recipes
2. Deploying the services and application
    a. Provisions machines in the cloud using cloud drivers
    b. Downloads, installs, and configures services
    c. Installs your application
    d. Configures the monitoring and scaling features
3. Monitoring and managing the deployment using the Cloudify web management console or the Cloudify shell

The following list of benefits can be seen as a summarization of already mentioned features with additional aspects:

- TOSCA-based, pure-play standard-driven orchestration
- Topology-driven and application-centric management and monitoring of the entire NFV lifecycle
- Integration with any tool chain
- Support for any application stack, with native OpenStack, bare metal, and virtual appliance support (enabling portability to any cloud and hybrid cloud models)
- Support for containerized and non-containerized workloads
- Designed for federated deployment
- Support legacy network functions
- Built in auto-healing and auto-scaling policies for deployed applications
- Embeddable (OEM)

3.2.1.4 Juju

Juju [JUJU] is an open source project based on a universal service modelling system with a service oriented architecture and service oriented deployments. It allows to deploy, configure, manage, maintain and scale cloud services easily and efficiently on public clouds, physical servers, OpenStack and containers. The basic approach of Juju is to make use of so-called Charms that contain all the instructions necessary for deploying and configuring cloud-based services. These Charms can be composed to a more complex service – also known as Service Composition. Service Composition is one of the most important features of Juju. In particular, it allows the combination of multiple services into a single functional system by providing two mechanisms for such composition:

- Integration of services across network through standard interfaces
- Co-locate services on a common machine

Originally, Juju was not intended to follow the ETSI NFV MANO specifications but the impact of NFV solutions in the growing market convinced to try at least a mapping of Juju to the ETSI NFV architecture ideas. Actually, as claimed by the developers of Juju, the service-orientation makes Juju well suited to the role of the VNFM. From this point of view, Juju enables higher-level orchestrators to make decisions and communicate them clearly and simply to the underlying model provided by Juju.

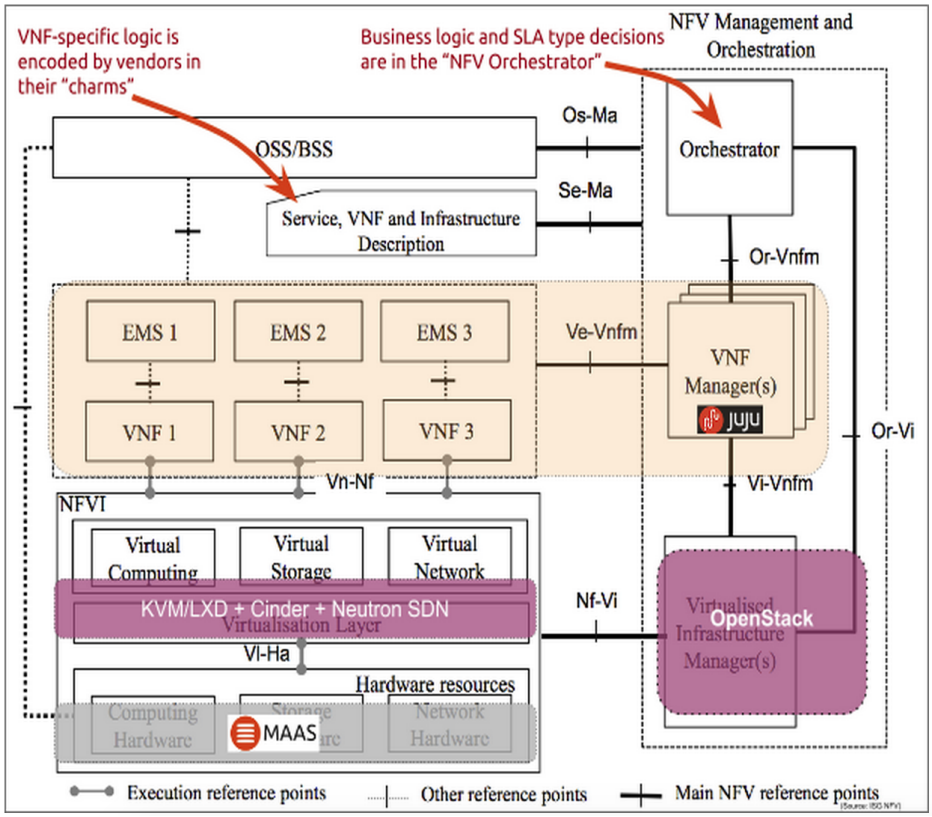


Figure 10. Mapping of Juju to NFV Architecture

Again, as depicted in Figure 6, Juju can be seen as a generic Virtual Network Function Manager (generic VNFM) in the ETSI NFV architecture. It takes care of managing the service lifecycle by using hooks implemented inside Charms (language-aware). Basically, these hooks are scripts and will be executed at the specific point in time of the lifecycle, namely when installing, configuring, starting, upgrading or stopping the service. Also dependencies can be controlled individually by using relation hooks declared by the corresponding Charm. These relationships between Charms are modelled by Juju to provide a topological view of service dependencies (control flows) whereas the data plane is not influenced by Juju (provides only information to support acceleration of the data plane) because it is mainly focused on controlling and coordinating semantics. For this reason, Juju provides also the placement of service units on machines based on a constraint language.

Specifically, in terms of ETSI NFV specifications, Juju supports both Vi-Vnfm and Or-Vi based resource orchestration by passing information to the underlying substrate (defined by the constraint language or even by an external agent) to let Juju place the units explicitly on predefined resources.

Bringing all together: Juju provides a multi-cloud and multi-datacenter infrastructure to deploy and manage complex VNFs including VNFM and VNFO functions. One of the most innovative approaches of Juju is to decouple the logic of managing services from the orchestration frameworks itself. Nevertheless, it does not follow the ETSI NFV information model and does not provide any ETSI NFV compliant interfaces as well. This makes it very hard to extend current functionalities like introducing custom autoscaling or network QoS management.

3.2.1.5 Hurtle and the Mobile Cloud Networking approach

Hurtle [HURTLE] is an orchestration framework developed by the ICCLab in the context of the project Mobile Cloud Networking (MCN) part of the FP7 program. Hurtle supports service composition of different services.
Hurtle provides three main components:

- Service Manager (SM) the endpoint to the Enterprise End User (EEU) providing an API for instantiating new tenant services
- Service Orchestrator (SO) managing the lifecycle of a tenant service instance
- Cloud Controller (CC) middleware providing an abstracted API on the infrastructure layer

The SM, shown in Figure 11, is composed by different functional elements. The CLI and UI interact with the Northbound Interface (NBI) and expose graphical and command line interfaces for allowing the EEU to instantiate service instances. The main role of the SM is to manage the lifecycle of the SO. Indeed, in the MCN terminology, an instance of the SO is instantiated per service instance. Therefore, the SM contains a SO registry (where all the SO bundles are present) and interacts with the CC for deploying the SO on top of the PaaS system.

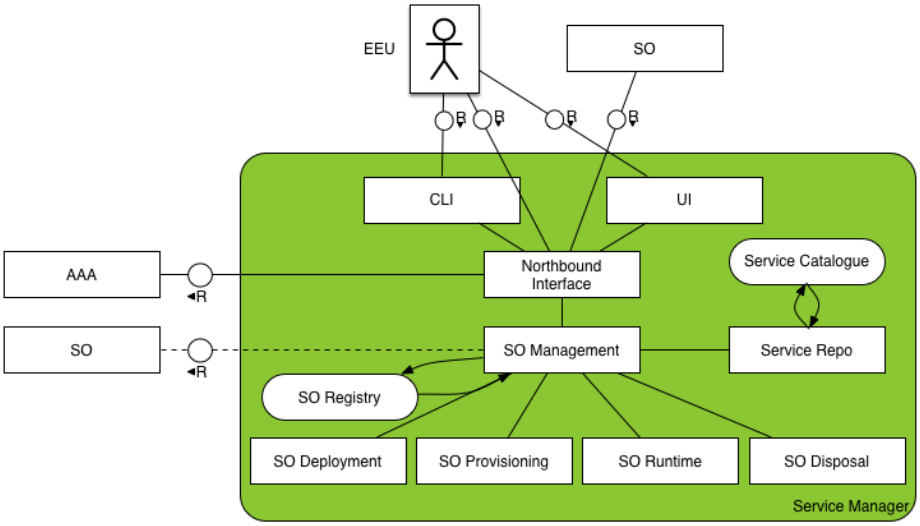


Figure 11. Service Manager (SM) internal architecture

The SO is the component taking care of the lifecycle of services. In particular, hurtle gives only some guidelines on how the SO should be implemented, providing only a general purpose SO based on the Heat IaaS orchestrator. The idea is to describe the services in different template graphs, one describing infrastructure resources (Infrastructure Template Graph) and one describing the service instance components (Service Template Graph). The important thing is that the Northbound Interface of the SO exposes an OCCI interface for the execution of the lifecycle steps described before.

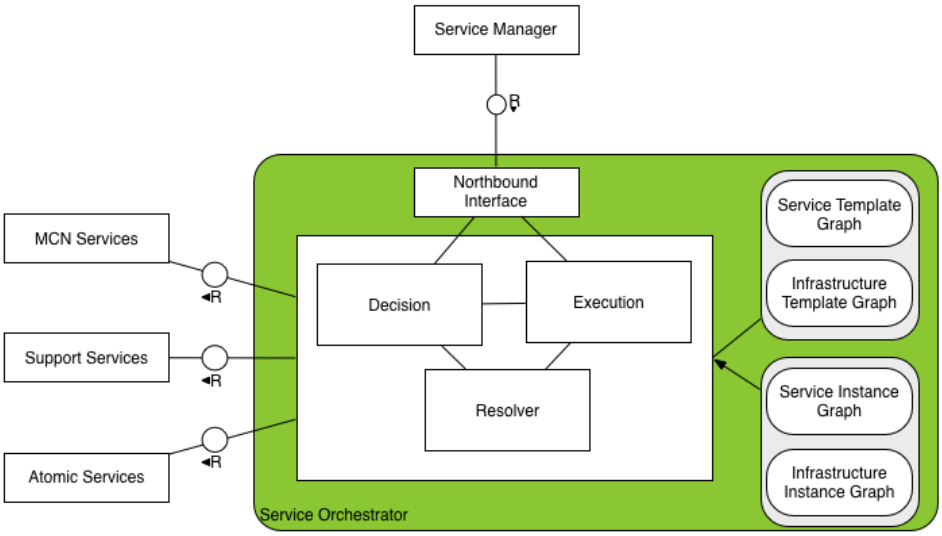


Figure 12. Service Orchestrator (SO) internal architecture

The CC provides a layer of abstraction of the Infrastructure. It provides abstracted API for allowing the SO to deploy the Infrastructure Template Graph on a large scale distributed infrastructure. Additionally it provides an API to the SM for generating the containers on top of which the SO will be executed.

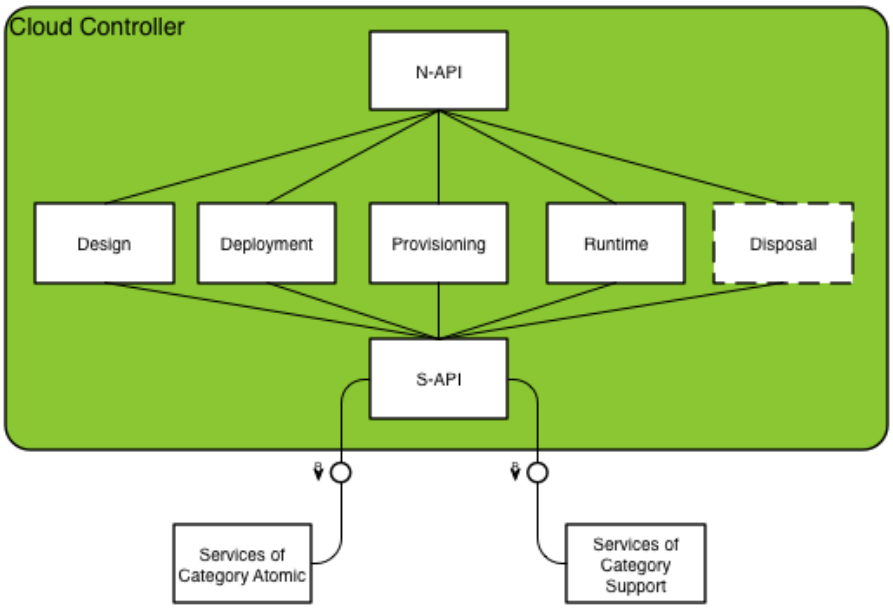


Figure 13. Cloud Controller (CC) internal architecture

Hurtle divides the lifecycle of a service in 6 key phases [HURTLE]:

- Design: where the topology and dependencies of each service component is specified. The model here typically takes the form of a graph.
- Implementation: This is where the developer(s) needs to implement the actual software that will be provided as a service through hurtle
- Deploy: the complete fleet of resources and services are deployed according to a plan executed by hurtle. At this stage they are not configured.
- Provision: each resource and service is correctly provisioned and configured by hurtle. This must be done such that one service or resource is not without a required operational dependency (e.g. a php application without its database).
- Runtime: once all components of an hurtle orchestration are running, the next key element is that they are managed. To manage means at the most basic level

to monitor the components. Based on metrics extracted, performance indicators can be formulated using logic-based rules. These when notified where an indicator's threshold is breached, an Orchestrator could take a remedial action ensuring reliability.

- Disposal: Where a hurtle service instance is destroyed.

The following list shows next steps and the roadmap announced by the developers of hurtle:

- Enhanced workload placement, dynamic policy-based
- Support for docker-registry deployed containers
- Runtime updates to service and resource topologies
- CI and CD support
- safe monitored dynamic service updates
- TOSCA support
- Support for VMware and CloudStack
- User interface to visualise resource and services relationships

3.2.1.6 T-NOVA

T-NOVA is an integrated project co-funded by the European Commission (7FP) where the objective is to design and implement an integrated management architecture containing an orchestrator platform. This orchestrator platform should be capable of provisioning, managing, monitoring and optimizing Virtualized Network Functions (VNFs) over Network/IT infrastructures. Additionally, it exploits and extends Software Defined Networking (SDN) aspects in the way of focusing on the Openflow technology for efficient management of network resources (supports network slicing, traffic redirection and QoS provision).

As depicted in Figure 14, the overall architecture of T-NOVA is basically composed of three platforms: the Orchestrator platform, the Infrastructure Virtualization and Management (VIM) platform and the Marketplace. The Orchestrator Platform, modules and interfaces are shown in Figure 11, is in charge of orchestrating VNFs by offering an automated deployment and configuration of VNFs and a federated management and optimization of networks and IT resources for accommodating network functions. Internally, the Orchestrator platform contains multiple submodules:

- Resource Repository module: includes a network topology service in order to assist the resource mapping procedure
- Resource Mapping module: provides a resource mapping algorithm to decide on resources and placement for the VNF deployment
- Connectivity Management module: instantiation of virtual networks and QoS provision to meet service level agreements (SLAs)
- Cloud Management module: allocation of computing and storage resources for NF instantiation (e.g. on OpenStack)
- NF Management modules: parameterizes the deployed NFs
- Monitoring and Optimization module: monitors networking and computing/storage assets to optimize them (VM resize/migration, virtual network reconfiguration)
- High Availability module: performs detection and forecast of operational anomalies. It triggers fault mitigation procedures (e.g. VM migration, network re-planning) once detected an anomaly.

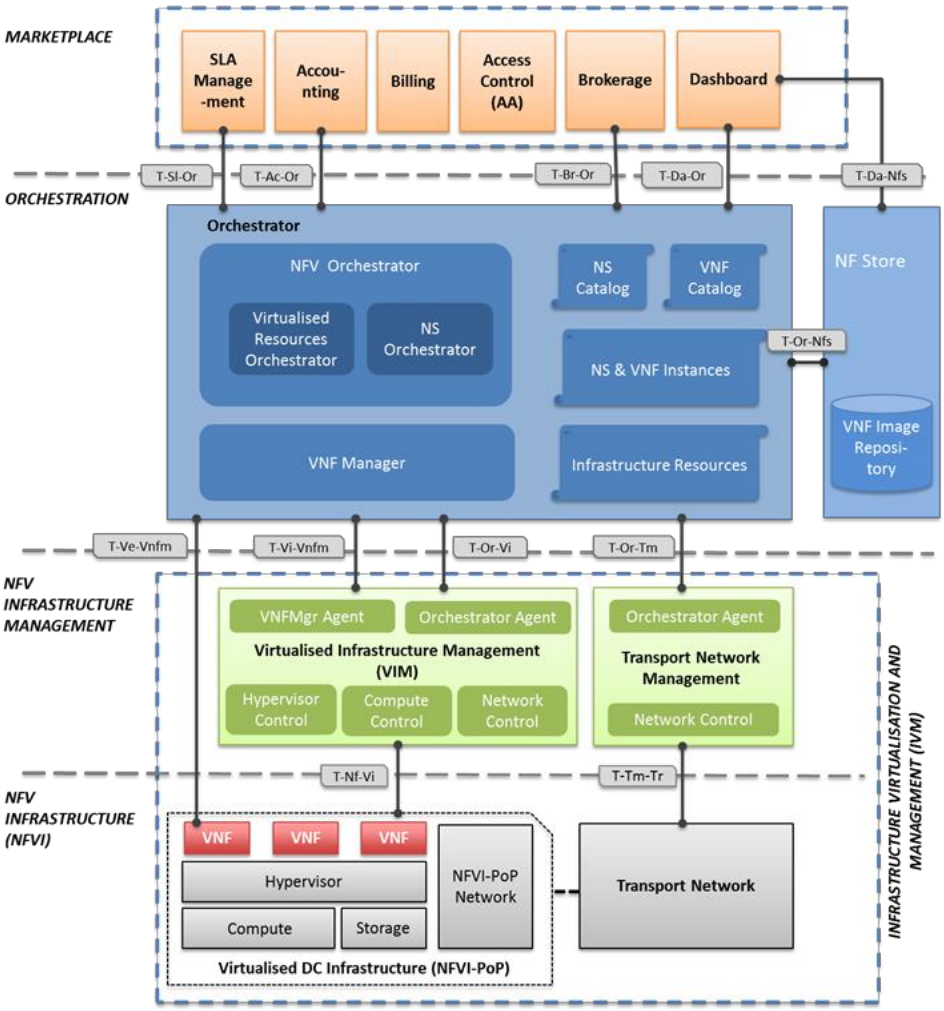


Figure 14. High-level view of overall T-NOVA System Architecture

To enable communication between the orchestration plane and other planes (e.g. service plane, management plane) it provides a set of northbound and southbound interfaces. Northbound interfaces expose functionalities like network and cloud resource listing, function deployment, service provisioning, service monitoring and reconfiguration, and service teardown. Southbound interfaces are used for the allocation and management process of network and cloud resources, in particular, interfacing with OpenStack's RESTful API, Open Cloud Computing Interface (OCCI) or even with a dedicated SDN controller.

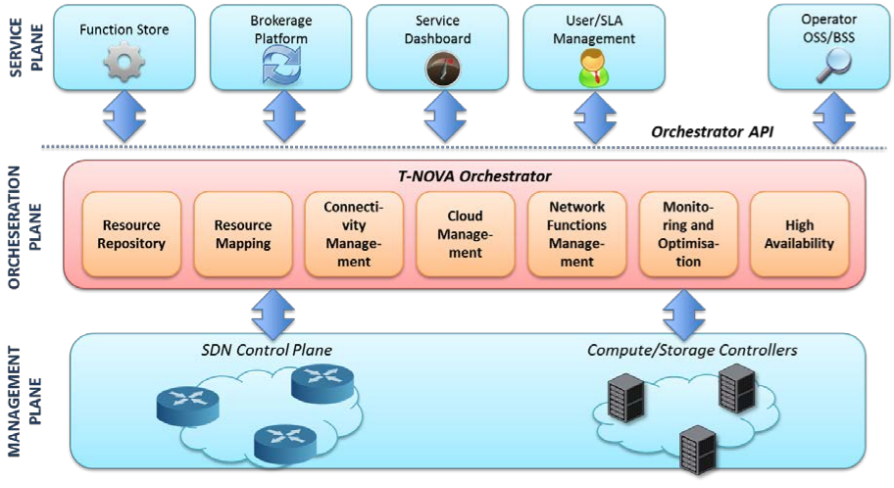


Figure 15. T-NOVA Orchestrator platform, modules and itnerfaces

In terms of the ETSI NFV specification the Infrastructure Virtualisation and Management (IVM) layer can be seen as the VIM. It provides the following functionalities:

- Support separation of control plane, data plane and network programmability
- Allocation, management and release of resource. Additionally, it offers resource isolation, optimisation and elasticity.
- Controlling horizontal and vertical scaling
- Provides physical network connectivity between servers by using L2 Ethernet switched networks (subnets)
- Managing networks by controlling virtual switch or real physical SDN-capable switches
- Managing L3 networks for interconnecting L2 subnets

The Marketplace, in turn, is in charge of offering Network Services and Functions to the community where Services can be published, acquired and instantiated on-demand. These Network Services and Functions are created by a variety of developers and allows to select prepared services and virtual appliances that best match specific needs. This attracts new market entrants to browse through the catalogue of available Services to find existing Services and to simplify Service creation as well. Mainly, the marketplace satisfies three functionalities:

- Publication of resources and NF advertisement
- VNF discovery, resource trading and service matching
- Customer-side monitoring and configuration of the offered services and functions

5G-SONATA

This project is part of the 5G-PPP framework recently launched by the European commission. At the moment of writing this deliverable there were no public documents available describing what is the innovation which this project will bring in the management and orchestration of network functions. The only known information are the ones mentioned on their public website [5G_SONATA]. They aim to:

- Reduce time-to-market of networked services
- Optimize resources and reduce costs of service deployment and operation
- Accelerate industry adoption of software networks

5G Exchange

The 5G Exchange project [5G_EX], launched in the end of 2015, is an European-funded and Ericsson-headed project aiming to deliver a “unified European 5G Infrastructure service market that integrates multiple operators and technologies”. It focuses on supporting cross-domain services orchestration over multiple administrations or multi-domain single administrations being compatible with NFV and SDN technology. So the objective of this project is to design and specify architecture, mechanisms, algorithms and enablers for automated and fast provisioning of infrastructure services in a multi-domain/multi-operator 5G environment. A proof of concept prototype should be implemented by building a working end-to-end system.

Since the project is in a very early state, the documentation is very limited at the moment of writing this deliverable but a first approach of the 5G Exchange architecture was already published as depicted in Figure 16.

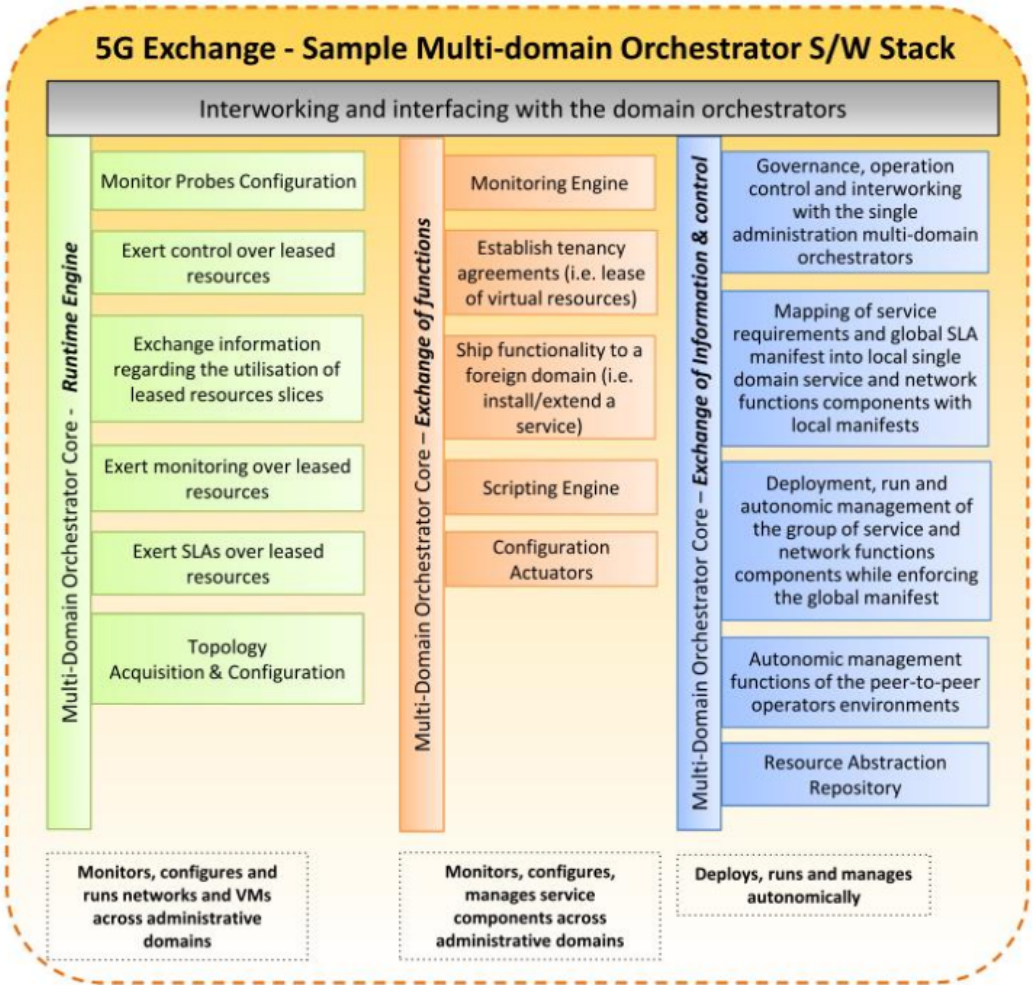


Figure 16. 5G Exchange architectural approach

The main components of the 5G architecture are:

- Runtime Engine: The Runtime Engine is in charge of managing networks and Virtual Machines across multiple domains.
- Exchange of Functions: This component manages service components across multiple domains
- Exchange of Information and Control: This enforces interworking, SLAs, mapping and automatic management of service and network functions.

3.2.2 NUBOMEDIA approach beyond SotA

This section serves as a comparison between the orchestration solutions described above and the requirements that the NUBOMEDIA Media Plane management components need to support. For this reason, it is proposed a taxonomy that includes generalized features of orchestration systems and NUBOMEDIA-related requirements where each of the solutions is compared to. To see if a specific feature is supported or not, it can be differentiated between the following cases:

- Yes, means that this feature is supported out of the box. It requires no further adoptions and is mentioned or even described in the documentation of the corresponding project.
- No, means that this feature is not supported at all. Since most of the projects are fairly new at the moment of doing this comparison, the projects are continuously under development with frequently improvements and new features, so it might be that any not-supported feature will be supported soon, a new feature is not yet announced to the community or at least it is not mentioned explicitly.

- Ext, stands for extendable and indicates if a not-supported feature is supportable by extending, integrating or replacing (re-implementing) a specific component that is in charge of this functionality. Basically, this requires a guideline for developing components with extended functionalities or a comprehensive documentation explaining architectural components and interfaces, relations between them and workflows. In case that a possible extension - to support a feature - is not mentioned explicitly or the documentation is not comprehensive enough or even absent it is seen as not extendible and therefore not supported.

	OpenMANO	OpenBaton	Tacker	Juju	Cloudify	Hurtle / MCN	T-Nova (not yet released)	5G-SONATA	5G Exchange
NFV compliancy	Yes	Yes	No	No	No	No	No	no documentation	no documentation
Provides NFVO	Yes	Yes	Yes	No	Yes	Yes	Yes		
Provides VNFM	No	Yes	Yes	Yes	No	No	Yes		
License type									
Open source	Yes	Yes	Yes	Yes	Yes	Yes	Yes		
Supported cloud infrastructures									
OpenStack	Yes	Yes	Yes	Yes	Yes	Yes	Yes		
Resource abstraction layer									
Compute	Yes	Yes	Yes	Yes	Yes	Yes	Yes		
Storage	No	Ext	Yes	Yes	Yes	Ext	No		
Volume	No	Ext	No	No	Yes	Ext	No		
Network	Yes	Yes	Yes	No	Yes	Yes	Yes		
Management tools									
CLI	Yes	Ext	Yes	Yes	Yes	No	No		
API	Yes	Yes	Yes	No	Yes	Yes	No		
GUI	Yes	Yes	Yes	Yes	Yes	Yes	No		
Core service layer									
Identity service	Yes	Yes	No	Yes	Yes	Yes	No		
Image repository	Yes	Yes	No	Yes	Yes	Yes	No		
Charging and Billing	No	Ext	No	Yes	No	Yes	No		
Logging	Yes	Yes	No	Yes	No	Yes	No		
Monitoring layer	No	Yes	Yes	Yes	Yes	Yes	No		
Basic metrics	No	Yes	Yes	Yes	Yes	Yes	No		
Custom metrics	No	Yes	Yes	Yes	Yes	Yes	No		
Elasticity management layer									
Scaling	No	Yes	No	Yes	Yes	No	No		
Manual	No	Yes	No	Yes	Yes	No	No		
Automatic	No	Ext	No	Yes	Yes	No	No		
Scale-in Termination rules	No	Ext	No	No	No	No	No		
Placement of Media Elements	No	Ext	No	No	No	No	No		

Table 1. Comparison of several orchestration solutions against several features

As you can notice from the previous table, none of the existing solution was satisfying all the NUBOMEDIA requirements. Furthermore, it is important to notice that the

selection was done during release 4 when most of the presented tool did not have source code to be downloaded and executed. Nevertheless, in parallel Fraunhofer FOKUS and TU Berlin launched a new NFV compliant platform (OpenBaton) which was providing almost all the features required in NUBOMEDIA. For this reason, it was decided to invest additional effort coming from WP3 related tasks in extending this platform for implementing the additional features required in NUBOMEDIA.

3.2.2.1 OpenBaton

OpenBaton is an ETSI NFV compliant Network Function Virtualization Orchestrator (NFVO) implemented by Fraunhofer FOKUS and TU Berlin being capable of interoperating with VNF Managers implemented by third parties. This project started in the beginning of 2015 (in parallel with the NUBOMEDIA rel.4) and is highly active with almost weekly improvements and bugfixes. OpenBaton is completely open source and claims to be fully compliant with the ETSI Management and Orchestration specifications. Therefore, it includes a consistent set of features and tools enabling the deployment of VNFs on top of cloud infrastructures.

Since OpenBaton is fully compliant with the ETSI NFV MANO specifications, the architecture of OpenBaton is also aligned to it. Nevertheless, a simplified high-level architectural overview of OpenBaton can be found in **¡Error! No se encuentra el origen de la referencia.** Following this architecture, three main components can be pointed out:

- Orchestrator (NFVO): The Orchestrator is in charge of processing the key functionalities of the MANO architecture. For this reason it maintains an overview on the infrastructure allowing to register and manage multiple Vim instances (also known as Network Function Virtualization Infrastructure Point of Presence or simply NFVI-PoP) dynamically in order to support the deployment of VNFs across multiple datacenters and regions. This enables the NFVO to take care of validation and authorization of NFVI resource requests from the VNF Manager. It is also responsible for the Network instantiation and Network Service instance lifecycle management (e.g. update, query, scaling, collecting performance measurement results, event collection and correlation, and termination). The instantiation of VNFs itself is done in coordination with VNF Managers.
- VNF Managers (VNFMs): The VNF Manager is in charge of managing the lifecycle (e.g. instantiate, start, configure, modify, stop, terminate,) of VNF instances. Therefore, each VNF of a certain type is explicitly assigned to a specific VNF Manager whereas a VNF Manager might be capable of managing multiple VNFs of the same type or even of different types. Depending on the type of the VNF to handle, the VNF Manager might be able of handling generic common functions applicable to any type of VNF or in case of handling specific VNFs the VNF Manager needs to provide specific functionalities for their lifecycle management. A list of main functionalities that each VNF Manager have to provide is listed in the following:
    - VNF instantiation and configuration
    - VNF instance modification
    - VNF instance scaling out/in and up/down
    - VNF instance termination
- Virtualized Infrastructure Manager (VIM): The VIM is in charge of controlling and managing NFVI compute, storage and network resources. A VIM might be capable of handling a certain type of NFVI resource (e.g. compute-only, storage-only, networking only) or even multiple types. At the moment OpenBaton

supports OpenStack as a first example of a VIM for NFV PoPs. Nevertheless, thanks to the "pluggability" it is easily possible to support other types of VIMs by implementing another plugin (vim-driver) that supports the type of infrastructure of choice. Anyway, the main tasks of a VIM of any type is listed in the following:

- Orchestrating the allocation, upgrade, release and reclamation of NFVI resources
- Managing inventory related information of NFVI hardware and software resources, and discovery of the capabilities and features of such resources
- Management of the virtualized resource capacity
- Management of software images (add, delete, update, query and copy)
- Collection of performance and fault information
- Management of catalogues of virtualized resources that can be consumed from the NFVI

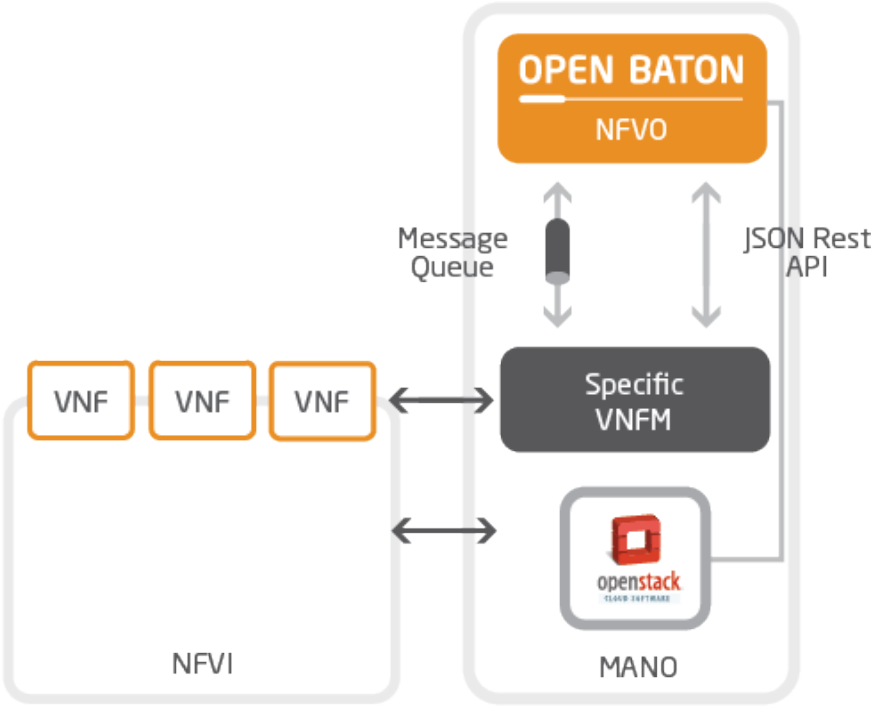


Figure 17. High-level Architecture of OpenBaton

Additionally, as shown in Figure 18, OpenBaton provides already a first version of a VNF Manager, called generic VNFM, supporting generic functionalities. The generic VNFM is tightly coupled with OpenBaton's EMS which runs as a software within deployed Network Functions. Thanks to the EMS, the generic VNFM is capable of executing specific configuration scripts within virtual machine instances at runtime. This VNF Manager is instantiated on demand by the NFVO where the VNF Manager can request from the NFVO the instantiation, modification, starting and stopping of virtual services (or even may request the specific VIM instance directly). Apart from generic functionalities the generic VNFM can be easily extended and customized to support specific types of VNFs.

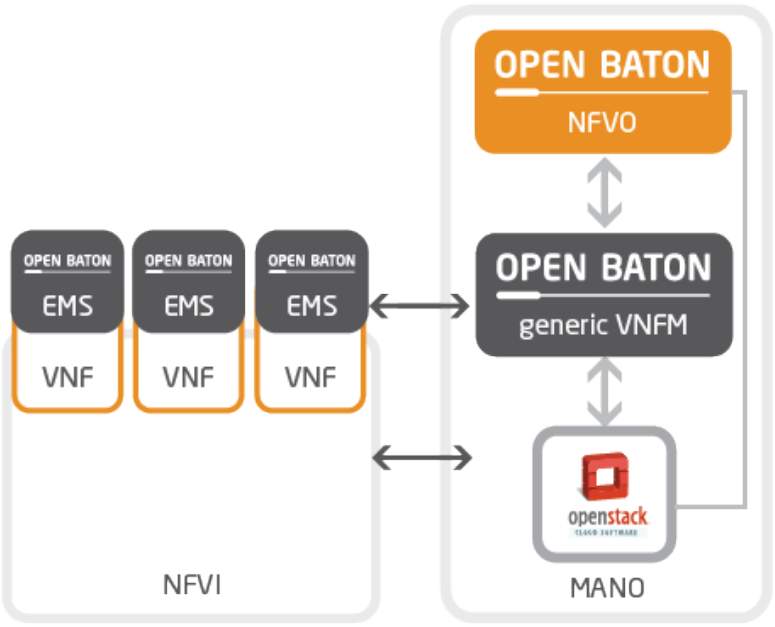


Figure 18. High-level Architecture of OpenBaton and the Generic VNFM inlcuding the EMS

Furthermore, OpenBaton provides some interfaces and frameworks that makes it easy to align the NFVO and VNFMs to certain needs. Some of the most important aspects are listed below:

- Plugin-mechanism: Thanks to the plugin-mechanism, plugins can be easily added to support other types of VIMs and monitoring frameworks. This can be done by using the provided plugin-sdk and implementing the corresponding interface that the plugin needs to satisfy in order to support the functionalities that are mandatory for communicating and managing either the VIM or the monitoring framework.
- openbaton-client: The openbaton-client, also known as the SDK, supports some basic functionalities to communicate with the NFVO. This includes the management of NSDs, NSRs, VIM instances, Virtual Links, Images, Events and Configurations.
- Vnfm-sdk: The vnfm-sdk, providing an AbstractVNFM, can be used to implement a new VNFM with specific functionalities related to specific VNFs. Depending on the communication type of choice it provides also more specific AbstractVNFMs with a built-in communication between NFVO and VNFM (e.g. AbstractVnfmSpringAmqp, AbstractVnfmSpringReST). In the simplest case it is just important to implement the methods that are called during the lifecycle management (e.g. instantiate, modify, configure, start, stop, terminate, etc). But also more complex VNF Managers can be implemented and integrated easily supporting specifc management of VNFs, additional entities, monitoring, autoscaling and fault management – just to mention a few options. The framework for implementing new VNFMs provides also a built-in helper, called VNFMHelper, for supporting a simplified mechanism to communicate with the NFVO in order to send and receive specific messages that may include requests (action-based) to the Orchestrator itself.

3.2.3 NUBOMEDIA outcomes

Extensions to the OpenBaton NFVO and generic-VNFM have been directly pushed on the OpenBaton repositories. The additional components are currently part of the tub-

nubomedia github repository [TUB_GITHUB] and we'll be integrated in the mainstream repository as soon as their implementation is completed and properly tagged.

3.2.4 References

Referenced websites
[5G_EX] https://5g-ppp.eu/5gex/

[5G_SONATA] http://www.sonata-nfv.eu/

[CLOUDIFY] http://getcloudify.org/

[HURTLE] http://hurtle.it/

[JUJU] https://juju.ubuntu.com/get-started/

[OPENBATON.ORG] http://openbaton.org

[OPEN_MANO] https://github.com/nfvlabs/openmano

[TACKER] https://wiki.openstack.org/wiki/Tacker

[TNOVA] http://www.t-nova.eu/

[TUB_GITHUB] https://github.com/tub-nubomedia

Referenced papers or books
[ETSI_WP] ETSI. Network functions virtualisation - introductory white paper, October 2012. At the SDN and OpenFlow World Congress, Darmstadt Germany.

[MANO] Network Function Virtualization Management and Orchestration. (2014). Retrieved December 14, 2015 from http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf

3.3 PaaS for Real-Time Multimedia Applications

Platform as a Service (PaaS) has been one of the vibrant topics in recent cloud computing technology trends. It is foreseen to be a game changer for next generation Real-Time multimedia applications. However, unlike Infrastructure as a Service (IaaS) which is making efforts in consolidating and standardization, the PaaS market is highly fragmented with sets of varying ecosystem capabilities.

The National Institute of Standards and Technology (NIST) Definition of Cloud Computing defines PaaS as "the capability provided to the consumer to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment" [NIST_DEF]

From this well-known and often cited definition, one can already see that the term PaaS be interpreted from different perspectives. In fact, definitions are widely spread and often conducted from contrasting perspectives resulting in diverging PaaS categories missing a common consensus. The result is a crowded market of PaaS offerings that sometimes provide completely different capabilities [FORRESTER_REP]. In general, the definition only requires the ability to deploy applications which have certain dependencies that are supported by the platform environment. Moreover, it demands the abstraction from the underlying cloud infrastructure while possibly granting access to unspecified configuration settings of the PaaS environment.

The NIST defined a Cloud Computing Reference Architecture as presented in Figure 19.

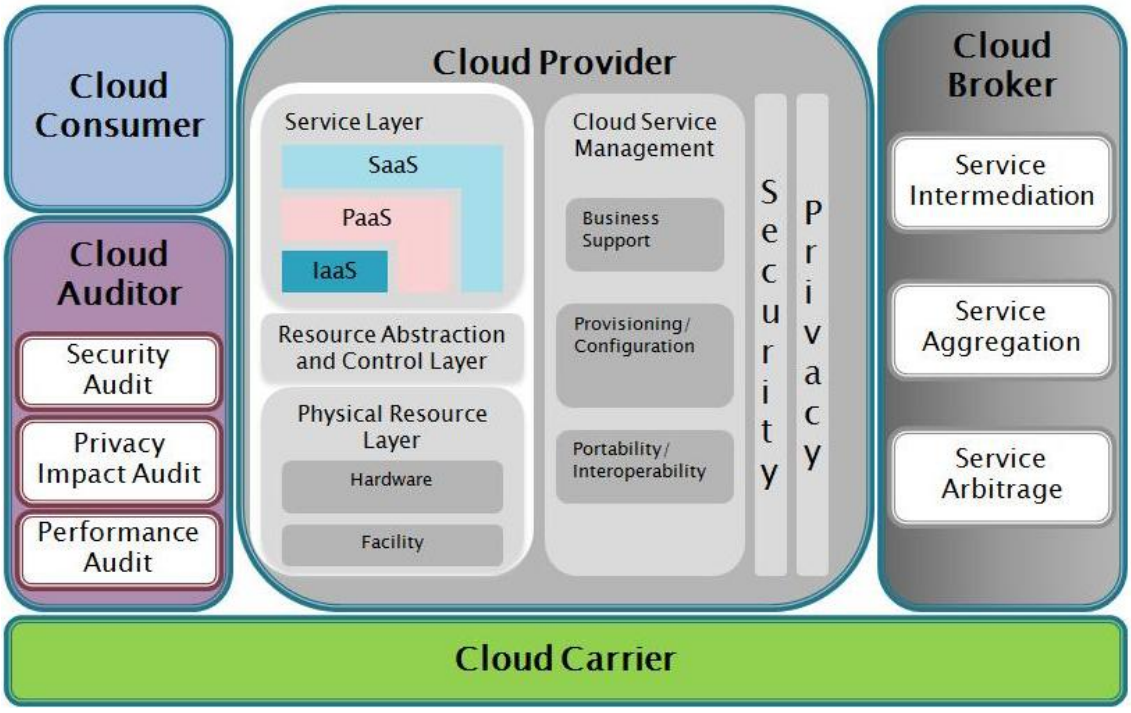


Figure 19 NIST PaaS Reference Architecture

Most notably, this architecture, explicitly defines 5 different roles:

- Cloud Consumer: a person or an organization using a service from the Cloud Provider.
- Cloud Provider: a person or (more likely) an organization making a service available to interested consumers
- Cloud Auditor: a person or (more likely) an organization conducting an assessment of cloud services, information system operations, performance and security of the cloud implementation
- Cloud Broker: a person or (more likely) an organization that manages the use, performance and delivery of cloud services, including negotiations between
- Cloud Providers and Cloud Consumers
- Cloud Carrier: an intermediate organization that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers

While this model clearly describes the various players in a PaaS environment, it is fits more to a description of the PaaS ecosystem. In addition, the model very abstract when it comes to the actual software architecture of the PaaS implementation.

As an evolution for the NIST reference model, [GARTNER_MODEL] defines a reference model of the core components and fundamental architecture of a comprehensive implementation as presented in Figure 20

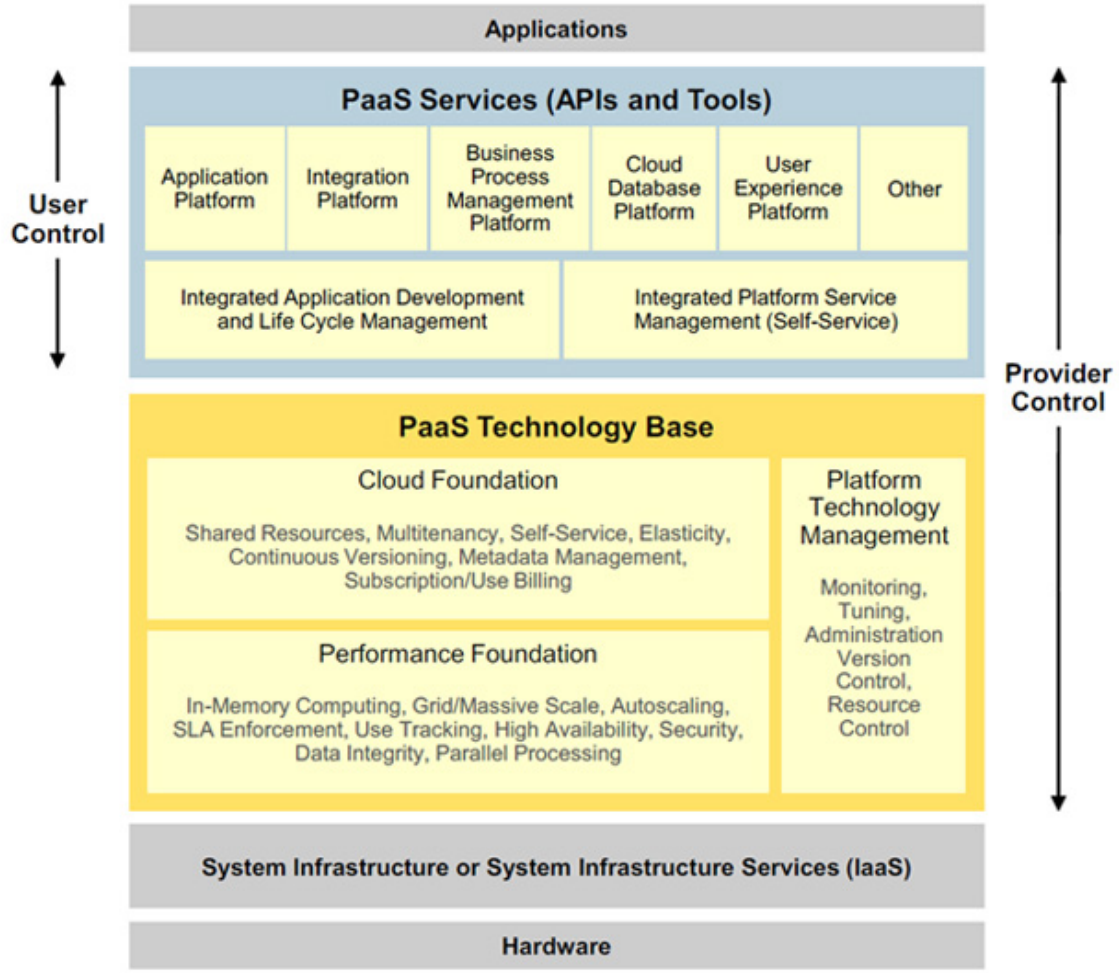


Figure 20 Gartner Refernece Archiecture for PaaS

In particular, this model distinguishes between:

- hardware and infrastructure (network, storage etc.) services that correspond to an IaaS solution
- a sound Technology Base dealing with scalability and elasticity, SLA management, billing, monitoring etc. – in brief with all the fundamental infrastructure services that will be required for any PaaS-based application
- the actual PaaS Layer providing APIs and tools allowing the implementation, integration, delivery and management of specific applications.

Gartner's reference model just mentions the user and the provider as 3rd party entities without going into details as to what they might need. However, when it comes to evaluating a potential PaaS vendor or implementing a PaaS stack, we prefer the

following model that was adopted from the PaaS Reference Architecture by Forrester Research.

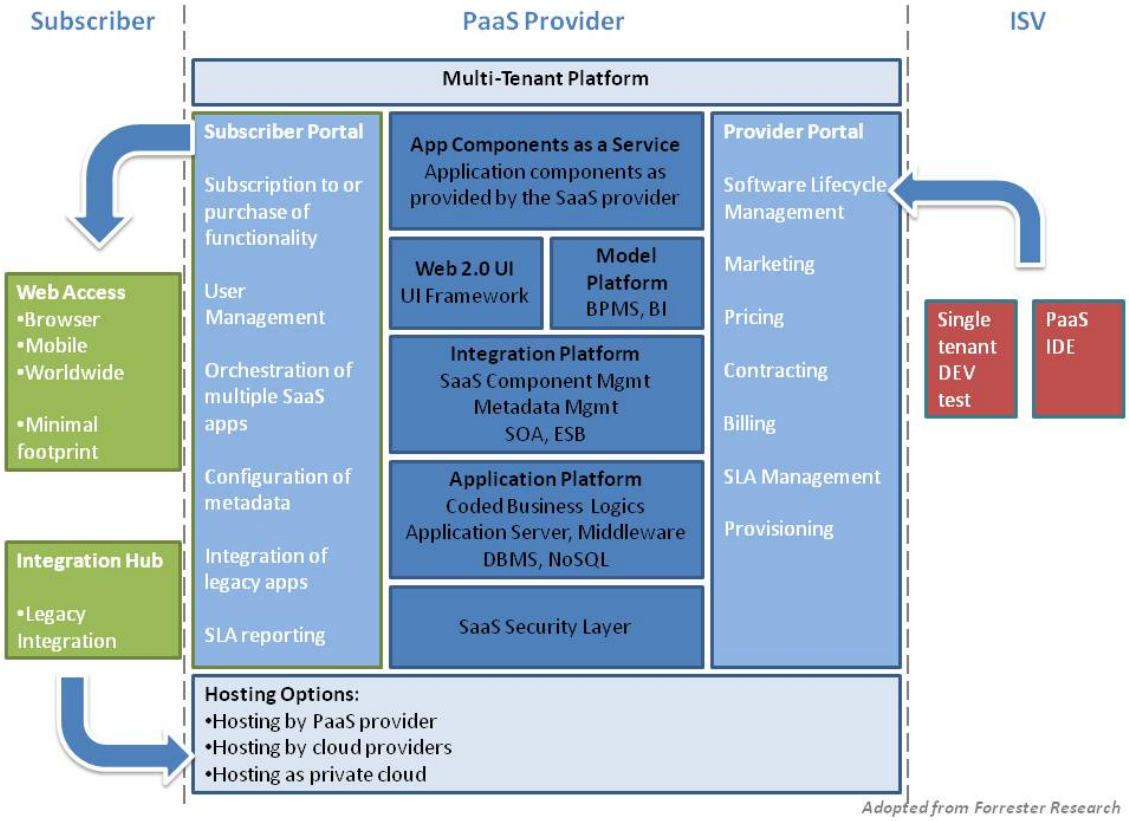


Figure 21 Forrester Research PaaS Reference Architecture

The Forrester based reference architecture for the PaaS shows what is required both from the side of the user/subscriber (green parts) as well as by potential Independent Software Vendors (ISV) that will implement applications on top of the PaaS platform (red parts). The blue parts of the diagram concentrate on what actually has to be implemented by the PaaS provider, most notably (from bottom to top):

- a sound layer ensuring security and data protection across the entire platform
- an application platform based on which specific applications will be implemented, delivered, and maintained; including appropriate middleware and persistence
- an integration platform by means of which various components and services can be managed and orchestrated, data can be exchanged between these etc.
- a UI framework based on which (Web or mobile) user interfaces can be implemented which will then be used by the users/subscribers to access to application
- a model platform that allows for the integration of the PaaS services with other applications and services, typically via BPMS
- some selected application components as they might be provided by the PaaS provider directly, these could serve as demos on how to use the system or really provide useful functionality that can be utilized by the ISVs when implementing other applications.

All these aspects (referring to boxes in dark blue) also apply to a SaaS architecture. In order to provide a PaaS platform, however, some more work is required as shown in the light blue boxes of the diagram:

- Towards the user/subscriber methods and tools for purchasing, reporting, user management etc. are required, of course. But even more, mechanisms will be needed by means of which the subscriber can configure the application in terms of their own metadata, orchestration with other apps being used, and even the integration with legacy systems.
- The ISV, on the right hand side of the diagram, will require tools for Software lifecycle management, including a development environment and sandbox area for testing purposes, but also an tools for marketing, contracting, billing etc. such that the ISV can actually reach out to customers, sell the application and charge for it.

In a certain sense, a PaaS platform thus has to provide more features than a SaaS platform as the latter is typically focused on a single application thus not requiring things like service orchestration or billing for multiple ISVs. Hence, implementing a PaaS platform sounds like a challenging task and requires a sound software architecture for sure.

Nevertheless, to be able evaluate the different PaaS for Real-Time applications, we shall focus on some base criteria that form a common ground model to proceed in analyzing the different PaaS solutions. These parameters include deployment models, platform and management.

***Deployment Features*:**
The NIST Definition of Cloud Computing [15] defines four types of deployment models for cloud computing including PaaS.

- *Private PaaS*. The private PaaS is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.
- *Community PaaS*. The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.
- *Public PaaS*. The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.
- *Hybrid PaaS*. The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

Also included in deployment features are other criteria such as open-source, method of isolation (containers or virtual machines), number of geographical data regions, pricing options, and whether or not a free option is available.

***Platform*:**

The platform is the main deliverable of a PaaS solution which includes the application hosting environment delivered as a service. The best approach in choosing a PaaS starts with the people who actually will be using it: developers. Most important, developers want to know that a given solution will support the software stack of their choice, and these criteria should be at the fort front of any PaaS evaluation.

Two stacks of components are crucial – the runtime stack and the service stack. Both stacks can be mingled to users through bindings. These bindings can be set as environment variables that include important properties of the service for example service routes, credentials and other configuration information.

- *Runtime Stack*: includes the basic runtime offered by the PaaS such as programming languages that application can be written in. Many applications also depend on middleware and frameworks that may be hosted by the PaaS.
- *Service Stack*: includes two types of services namely native and add-on service stacks. Native service stacks are hosted on and operated by the PaaS platform e.g. latency and performance critical core services, databases. Add-ons are 3rd party cloud services that developers can use to immediately extend their applications with a range of functionality such as data stores, logging, monitoring and other utilities.

***Extensibility*:**
Another key functionality of the PaaS platform is extensibility. A modern PaaS should enable developers to add own packages of services or runtimes to the PaaS environment, which can operate as isolated entities that can generate any of the service or runtime stacks capabilities.

Scalability:
An important aspect in considering the PaaS platform is also scalability. Most PaaS solutions provide horizontal and/or vertical scaling of applications.

- *Horizontal Scale*: also referred to as Scale Out is to add more nodes or resources to the network of nodes that are running the application. The advantage of horizontal scaling is that load is distributed amongst each node in the network and also it provides high availability. Also, it does not require downtime as you are adding more nodes without affecting the ones that are running the application
- *Vertical Scale*: or Scale Up is to add more power (CPU, RAM, Disk) to the existing node running the application. The advantage of vertical scale is that many servers do not need to be managed. The disadvantage is that it requires downtime as capacity is being added to a single running node. Also, potential risk of having a single point of failure if in case that single running node goes down.

Management:
In addition to the deployment model and the platform of the PaaS solution, the last key aspect is with regards to the manageability. The management allows control over the deployed applications and the configuration settings of the platform. It also provides the abilities to deploy and manage the lifecycle of the applications. This encompasses pushing, starting, and stopping of applications. Moreover, the provisioning of all native services and add-ons is initiated from the management tier. All available configuration and administration settings for the applications and the PaaS environment can also be controlled. This includes a wide range of functionality like scaling, logging, down to the creation of domain routes and environment variables. The management layer also

covers the resource usage monitoring that is relevant for billing and scaling decisions. All these functionalities are controlled by the management interface. This interface can be a RESTful API, console-based or driven via web user interfaces.

3.3.1 Description of current SoTA

After looking at the common criteria for analyzing PaaS solutions, the next subsection describes some popular PaaS solutions widely used around the world.

3.3.1.1 Cloud Foundry PaaS

Cloud Foundry is an open source PaaS originally developed by VMware and now owned by Pivotal Software [FOUNDRY]. It provides subscribers with a choice of clouds, developer frameworks and application services.

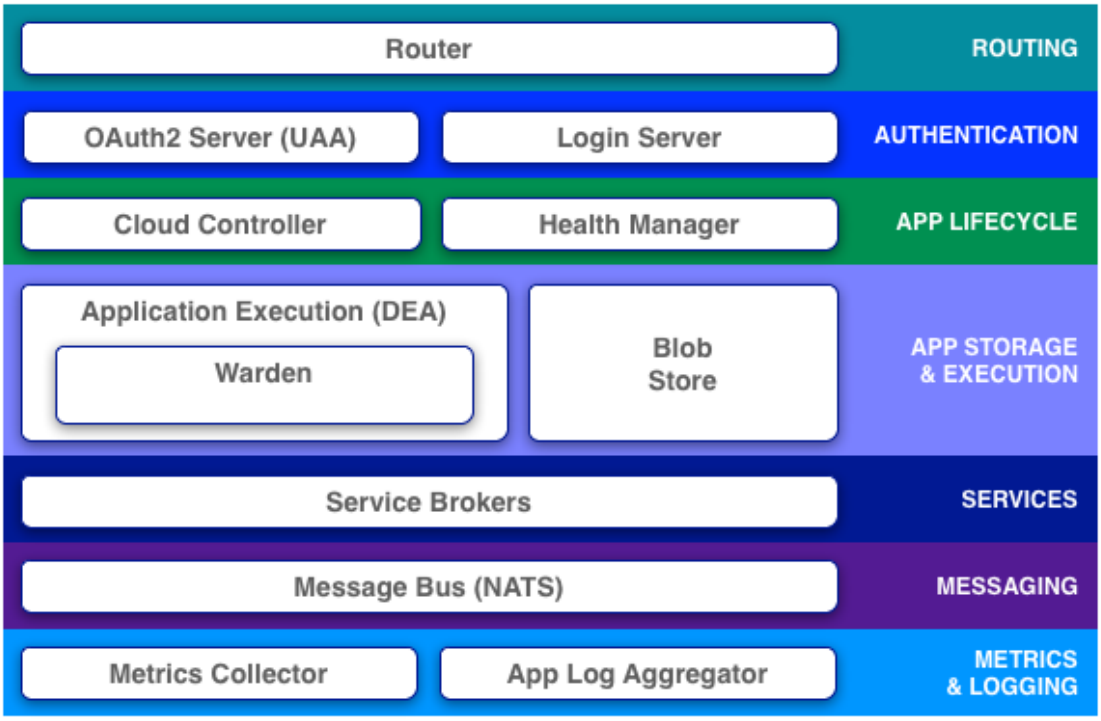


Figure 22 Cloud Foundry Components Overview

Cloud Foundry components include a self-service application execution engine, an automation engine for application deployment and lifecycle management, and a scriptable command line interface (CLI), as well as integration with development tools to ease deployment processes. Cloud Foundry has an open architecture that includes a build-pack mechanism for adding frameworks, an application services interface, and a cloud provider interface.

- Router: routes incoming traffic to the appropriate component, usually the Cloud Controller or a running application on a DEA node.
- OAuth2 Server (UAA) and Login Server: work together to provide identity management.
- Cloud Controller: is responsible for managing the lifecycle of applications. When a developer pushes an application to Cloud Foundry, they are targeting the Cloud Controller. The Cloud Controller then stores the raw application bits, creates a record to track the application metadata, and directs a DEA node to

stage and run the application. The Cloud Controller also maintains records of orgs, spaces, services, service instances, user roles, and more.

- Health Manager (HM9000): has four core responsibilities:
  - Monitor applications to determine their state (e.g. running, stopped, crashed, etc.), version, and number of instances. HM9000 updates the actual state of an application based on heartbeats and droplet exited messages issued by the DEA running the application.
  - Determine applications' expected state, version, and number of instances. HM9000 obtains the desired state of an application from a dump of the Cloud Controller database.
  - Reconcile the actual state of applications with their expected state. For instance, if fewer than expected instances are running, HM9000 will instruct the Cloud Controller to start the appropriate number of instances.
  - Direct Cloud Controller to take action to correct any discrepancies in the state of applications.

  HM9000 is essential to ensuring that apps running on Cloud Foundry remain available. HM9000 restarts applications whenever the DEA running an app shuts down for any reason, when Warden kills the app because it violated a quota, or when the application process exits with a non-zero exit code.
- Application Execution (DEA): Droplet Execution Agent manages application instances, tracks started instances, and broadcasts state messages.
- Blob Store: holds application code, build packs, droplets
- Service Brokers: Applications typically depend on services such as databases or third-party SaaS providers. When a developer provisions and binds a service to an application, the service broker for that service is responsible for providing the service instance.
- Message Bus (NATS): Cloud Foundry uses NATS, a lightweight publish-subscribe and distributed queuing messaging system, for internal communication between components.
- Metrics Collector and App Log Aggregator: collects logging and statistics, while the metrics collector gather metrics from the components. Operators can use this information to monitor an instance of Cloud Foundry.

Deployment features:
Cloud Foundry is a public PaaS which is designed to be configured, deployed, managed, scaled, and upgraded on any cloud IaaS provider. This is achieved by leveraging BOSH, an open source tool for release engineering, deployment, lifecycle management, and monitoring of distributed systems.

Platform:
From the platform runtime parameter, Cloud Foundry offers 8 programming languages with which applications can be deployed. These languages include Go, Groovy, Java, Node, PHP, Python, Ruby, and Scala. In addition it offers Tomcat as middleware platform for deployment of Java (version 6, 7) based applications. In addition, it provides the following frameworks: Grails (for Groovy), Play (Java), Rails (Ruby), Sinatra (Ruby) and spring for the Java runtime.

Scalability:
In terms of scalability, Cloud Foundry provides both horizontal and vertical scalability. Horizontally scaling an application creates or destroys instances of the application. Incoming requests to that application are automatically load balanced across all instances of the application, and each instance handles tasks in parallel with every other

instance. Adding more instances allows the application to handle increased traffic and demand. Vertically scaling an application changes the disk space limit or memory limit that Cloud Foundry applies to all instances of the application.

Management:
The management layer of Cloud Foundry is provided through a command line interface. With the command line interface, users are able to deploy, start and stop applications, access logs, configure application deployment parameters, just to name a few.

3.3.1.2 Red Hat OpenShift Origin

OpenShift is a PaaS solution offered by Red Hat [SHIFT]. Two types of PaaS deployment models are offered by Red Hat namely:

- Public PaaS: OpenShift Online is Red Hat's hosted public PaaS that offers an application development, build, deployment, and hosting solution in the cloud. It lets applications scale automatically in an elastic cloud environment, ensure faster time to market, and it is freely available online for registered users to host their applications
- Private PaaS: OpenShift Enterprise takes the same open source PaaS platform that powers the OpenShift Online hosted service and packages it for customers who want an on premise or private cloud deployment.

The rest of this section however will concentrate on Red Hat's OpenShift Origin because this is the PaaS solution fuelling both the Online and Enterprise version.

OpenShift mission statement is geared principally towards developers. Their main objective is to ease of use and complexity involved in deploying applications on the cloud. OpenShift was recognized by InfoWorld as one of 2015's best tools and technologies for developers, IT professionals, and businesses [INFO_WORLD].

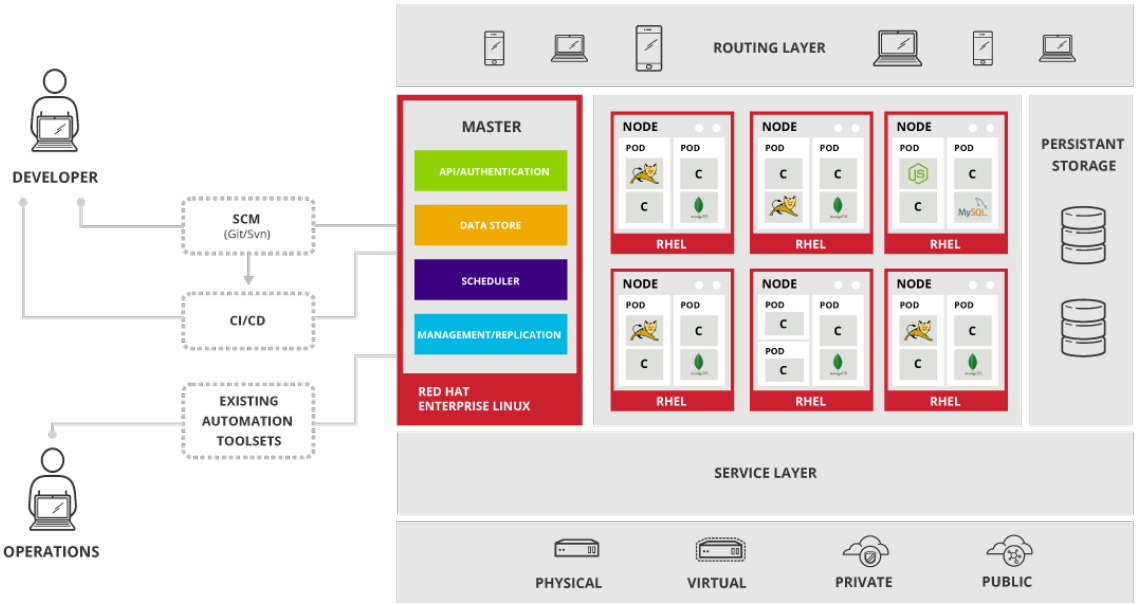


Figure 23 OpenShift Origin Architecture Overview

OpenShift provides a micro-services based architecture of smaller, decoupled units that work together. The architecture is designed to expose underlying Docker and Kubernetes concepts as accurately as possible, with a focus on easy composition of applications by a developer. Docker is an open platform for developers and sysadmins

to build, ship, and run distributed applications, whether on laptops, data center Virtual Machines (VMs), or on the cloud. OpenShift Origin uses it as containers for running applications. Kubernetes is an open source orchestration system for Docker containers developed by Google. OpenShift Origin uses Kubernetes for scheduling and packaging Docker containers onto nodes in a compute cluster and manages it workloads to ensure that their state machines reflects the users declared intentions.

- *Core services*: these include services such as authentication for security of the system; Data Store for keeping persistent metadata of PaaS use's applications available on given nodes and their capabilities; Scheduler service for determining placement of new containers on nodes; Management and replication services to ensure that a specific number of containers ("replicas") are running at all times.
- *Containers*: are the basic units of the applications which uses the concept of Linux Container Technology. This is basically a lightweight mechanism for isolating running processes, such that they are limited to interacting with only designated resources (processes, files, network, database, etc.). OpenShift Origin uses Docker containers which are based on Docker images. A Docker image is a binary that includes all of the requirements for running a single Docker container, as well as metadata describing its needs and capabilities.
- *Node*: is a worker machine which may be a Virtual Machine (VM) or physical machine. Each node has the services necessary to run Pods (a group of containers that make up the application) and is managed by the core components. The node also provides network proxy and load balancing services for scaling the traffic it receives amongst all replicas.
- *Persistent data*: is used to store custom Docker images for building and deploying user's applications.

Deployment features:
As mentioned above, OpenShift Origin is deployed as public and private PaaS

Platform:
From the platform runtime parameter, OpenShift Origin offers officially 6 programming languages with which applications can be deployed. These languages include Java, Node, PHP, Python, Ruby, and Pearl. However, since it is a community driven project, there are many more language support coming from the open source community. In addition it offers JBoss and Tomcat (Java) and Zend Server (PHP) as middleware. Database support (MongoDB, MySQL, PostgreSQL, SQLite). In addition, it provides the following frameworks: Django and Flask (Python), Drupal (PHP), Rails (Ruby), Vert.x (Java) and spring for the Java runtime. To facilitate usage for developers, OpenShift Origin provides numerous quick start templates with pre-created code repositories that allow developers to instantly deploy their favorite application frameworks in one-click.

Scalability:
In terms of scalability, OpenShift Origin provides both horizontal and vertical scalability. Horizontal scaling for applications is accomplished using HAProxy as a load-balancer. Whenever the platform gets a scaling request, it performs all the necessary steps to manage a new instance and configures HAProxy. Vertically scaling with OpenShift is accomplished by providing bigger containers, thus securing more resources.

Management:

The management layer of OpenShift Origin is provided through a command line interface web console and REST APIs. The REST APIs expose each of the core service functionality. Users make calls to the REST API to change the state of the system. Controllers use the REST API to read the user's desired state, and then try to bring the other parts of the system into sync. The controller pattern means that much of the functionality in OpenShift is extensible. By customizing those controllers or replacing them with your own logic, different behaviours can be implemented. From a system administration perspective, this also means the API can be used to script common administrative actions on a repeating schedule. Those scripts are also controllers that watch for changes and take action. OpenShift makes the ability to customize the cluster in this way a first-class behaviour.

3.3.1.3 Google App Engine

Google App Engine [APP_ENGINE] is a PaaS platform for developing and hosting web applications in Google-managed data centers. Applications are sandboxed and run across multiple servers. Integrated within App Engine are the Memcache and Task Queue services. Memcache is an in-memory cache shared across the AppEngine instances. This provides extremely high speed access to information cached by the web server (e.g. authentication or account information).

Task Queues provide a mechanism to offload longer running tasks to backend servers, freeing the front end servers to service new user requests. Finally, App Engine features a built-in load balancer (provided by the Google Load Balancer) which provides transparent Layer 3 and Layer 7 load balancing to applications.

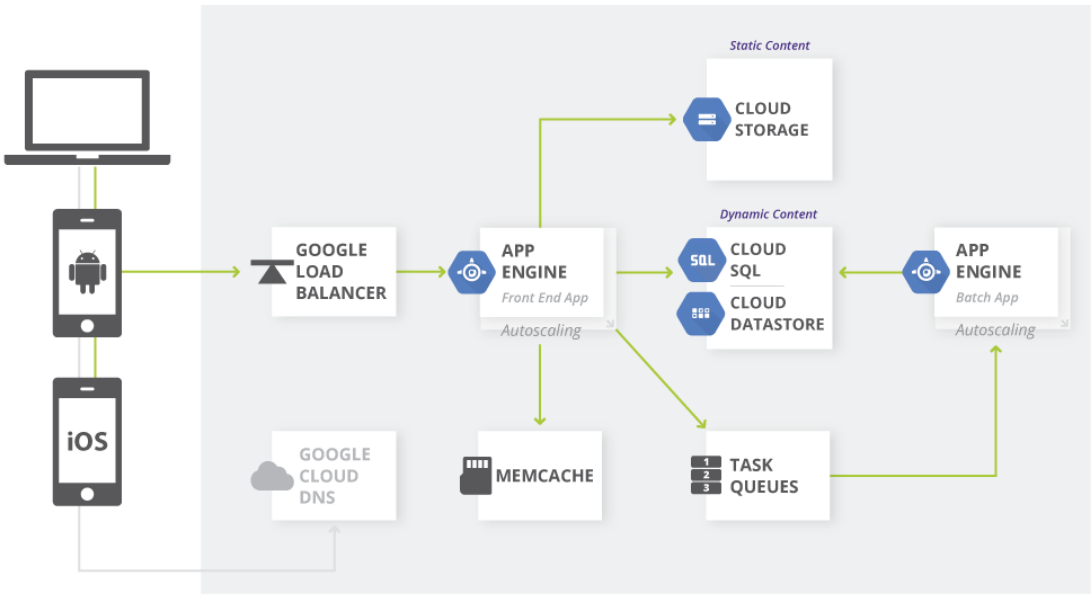


Figure 24Google App Engine High Level Overview

Deployment Features:
Google App Engine is a public PaaS and free up to a certain level of consumed resources. Fees are charged for additional storage, bandwidth, or instance hours required by the application.

Platform:
Google App Engine provides to developers four languages for developing applications; Java, Python, PHP and Go. The Java environment supports other languages that make use of the JRE and there is a SDK for each of the four main supported languages as well as a plugin for Eclipse. For frameworks, Django and Webapp2 for Python are provided.

Regarding service stacks, Google App Engine provides native services and APIs such as Google Cloud SQL, Datastore, Storage and a user authentication API which are common to many applications.

Scalability:
In terms of scalability, Google App Engine provides both auto scaling and vertical scalability. It provides managed infrastructure and runtime environments that are guaranteed to scale, but only if the applications fit the restrictions of Google App Engine

3.3.1.4 Salesforce Heroku

Heroku is one of the early PaaS providers [HEROKU]. It is based on a managed container system, with integrated data services and a powerful ecosystem, for deploying and running modern applications. Heroku is ideal for quick deployments and fits a wide range of distributed applications. Applications that are run from the Heroku server use the Heroku DNS Server to direct to the application domain. Each of the application containers or dynos are spread across a "dyno grid" which consists of several servers. Heroku's Git server handles application repository pushes from permitted users

Deployment Features:
Heroku is available as a public PaaS, located in Ireland - Dublin and in USA-Northern Virginia deployed on Amazon Web Service IaaS

Platform:
Heroku runs applications inside dynos, which are smart containers on a reliable, fully managed runtime environment. It provides to developers nine languages for developing applications; Java, Python, PHP, Go, Groovy, Clojure, Node, Ruby and Scala. The system and language stacks are monitored, patched, and upgraded, so it's always ready and up-to-date. The runtime keeps apps running without any manual intervention. For frameworks, Django, Flask, Grails, Play and Rails are available. Developers deploy directly from popular tools like Git, GitHub or Continuous Integration (CI) systems.

Regarding service stacks, Heroku supports Cloudant, Couchbase Server, MongoDB and Redis databases in addition to its standard PostgreSQL, both as part of its platforms and as standalone services. Outstanding difference to the other PaaS solutions is the huge number of add-on services (179) provided. Heroku Elements let developers extend their applications with add-ons, customize their application stack with Buildpacks and jumpstart their projects with Buttons. Heroku provides two fully-managed data service Add-ons: Heroku Postgres and Heroku Redis.

Scalability:

Heroku uses a process model that scales up or down applications instantly from the command line or Dashboard. Each application has a set of running dynos, managed by the dyno manager

Management:
The management layer of Heroku is provided through an intuitive web-based Dashboard that makes it easy to manage applications and monitor application performance. Also available is a command line interface.

3.3.1.5 Comprehensive comparison of PaaS solutions

The above sections have briefly described some of the popular PaaS solutions in the market today. To give a complete description of all solutions will span this document to eternity. Nevertheless, this section provides a close up side by side comparison of the above described and other PaaS solution with the criteria model provided above. This comparison is based on [SOLUTIONS_REVIEW] report on 2016 Comparison Matrix Report on PaaS. For simplicity's sake, the Solutions Review PaaS Comparison Matrix only includes runtimes, frameworks, middleware, and services that are native to, or fully supported by, each solution. However, it should be noted that 60 percent of the solutions listed are extensible, and can add new runtime and framework support via community buildpacks at varying degrees of difficulty to the user.

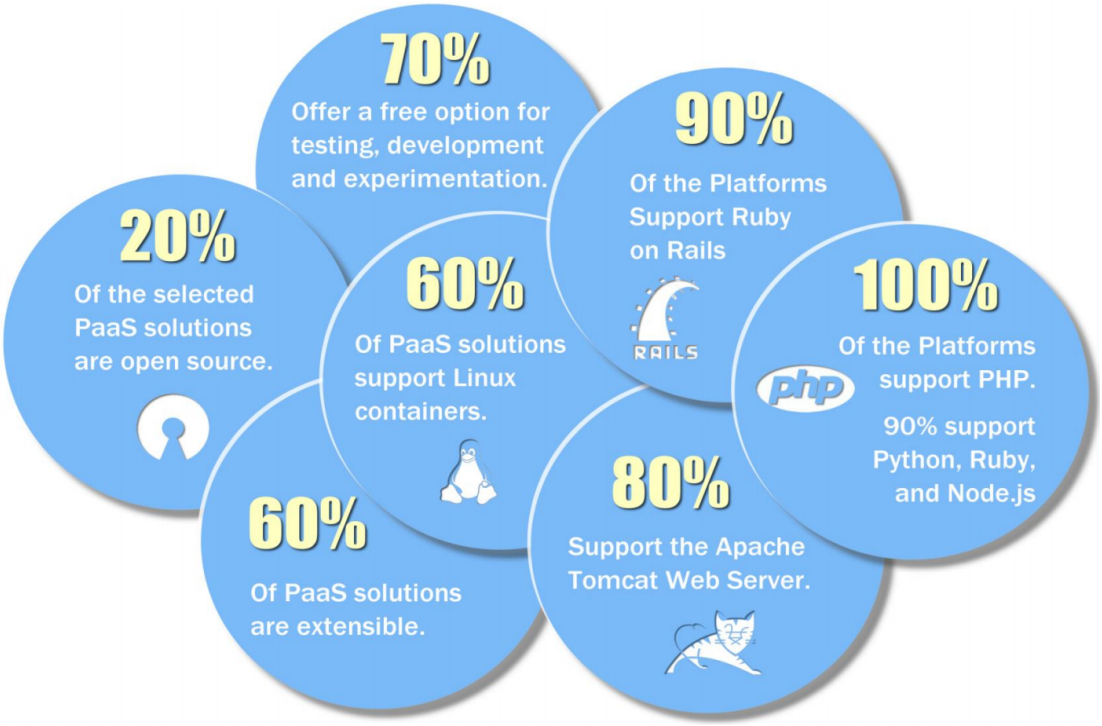


Figure 25 Solutions Review report on 2016 Comparison Matrix Report

Table 2 PaaS comparison Deployment Features

Name	Hosting	Open Source	Virtual Machines	Linux Containers	Metered Pricing	Monthly Pricing	Free option	Infrastructure
Pivotal Cloud Foundry	private	yes	no	yes	yes	yes	yes	private

Red Hat OpenShift Online	public	yes	no	yes	yes	yes	yes	Europe, North America
Red Hat OpenShift Enterprise	private	yes	no	yes	yes	yes	yes	private
Salesforce Heroku	public	no	no	yes	yes	yes	yes	Ireland USA
Google App Engine	public	no	yes	no	no	yes	yes	Europe, USA
Microsoft Azure	public	no	yes	no	yes	yes	yes	Asia, Europe, USA, Australia, Brazil
IBM Bluemix	public	no	yes	no	yes	yes	no	Europe, USA
Centurylink Appfog	Public & private	no	yes	yes	no	yes	no	Asia, Europe, USA
Engine Yard	public	no	yes	no	no	yes	no	Europe, USA, Australia, Brazil
Amazon Elastic Beanstalk	public	no	yes	no	no	yes	yes	Europe, USA, Australia, Brazil

Table 3 PaaS comparison Platform - Runtime

Name	Clojure	.NET	Go	Groovy	Java	jRuby	Node	PHP	Python	Ruby
Pivotal Cloud Foundry			yes		yes		yes	yes	yes	yes
Red Hat OpenShift Online*										
Red Hat OpenShift Enterprise[1]					yes		yes	yes	yes	yes
Salesforce Heroku*	Yes			yes	yes		yes	yes	yes	yes
Google App Engine			yes		yes			yes	Yes	
Microsoft Azure*		yes			yes		yes	yes	yes	Yes
IBM Bluemix*			yes		yes		yes	yes	yes	yes
Centurylink Appfog		yes			yes		yes	yes	yes	yes
Engine Yard						yes	yes	yes	yes	yes
Amazon Elastic Beanstalk		yes	yes		yes		yes	yes	yes	yes

Table 4 PaaS comparison Platform - Middleware

Name	Gunicorn	HAProxy	JBoss	Jetty	Nginx	Passenger	Rack	Puma	Tomcat	Unicorn
Pivotal Cloud Foundry			yes			yes		yes	yes	yes
Red Hat OpenShift Online*			yes						yes	

[1] *=Extensible

Red Hat OpenShift Enterprise[2]			yes						yes	
Salesforce Heroku*	yes			yes		yes		yes	yes	yes
Google App Engine			yes	yes	yes	yes				
Microsoft Azure*			yes	yes	yes				yes	
IBM Bluemix*			yes							
Centurylink Appfog	yes								yes	
Engine Yard		yes		yes	yes	yes	yes	yes	yes	yes
Amazon Elastic Beanstalk		yes	yes		yes		yes	yes	yes	

Table 5 PaaS comparison Platform - Framework

Name	Gunicorn	HAProxy	JBoss	Jetty	Nginx	Passenger	Rack	Puma	Tomcat	Unicorn
Pivotal Cloud Foundry			yes			yes		yes	yes	yes
Red Hat OpenShift Online*			yes						yes	
Red Hat OpenShift Enterprise[3]			yes						yes	
Salesforce Heroku*	yes			yes		yes		yes	yes	yes
Google App Engine			yes	yes	yes	yes				
Microsoft Azure*			yes	yes	yes				yes	
IBM Bluemix*			yes							
Centurylink Appfog	yes								yes	
Engine Yard		yes		yes	yes	yes	yes	yes	yes	yes
Amazon Elastic Beanstalk		yes	yes		yes		yes	yes	yes	

3.3.2 NUBOMEDIA approach beyond SotA

The last section presented the concept of PaaS and some of the most popular PaaS solutions and their limitation. This section presents the approach taken for the NUBOMEDIA project. The NUBOMEDIA project deals specifically with Real-Time multimedia applications.

An obvious limitation from the solutions presented above is the focus on Real-Time applications. Their infrastructure is generalized to web application as a whole and little emphasize on Real-Time Communication services.

One of NUBOMEDIA's intent is to address this limitation by providing an open source PaaS platform, which makes it possible for developers to implement and host advanced multimedia applications.

- Providing service stacks for facilitating the development of Real-Time applications

[2] *=Extensible
[3] *=Extensible

- Reducing complexity in deploying Real-Time applications by providing simple APIs that can be used from the web and programmatically.
- Provide cloud Media pipelines which offer specific APIs to define media processing topologies such as augmented reality filters and visual computing toolkits for seamless integration
- Providing add-ons to support real-time interoperability with other multimedia networks

With this objectives, and from the analysis of the existing solutions above, we have identified a couple of requirement the NUBOMEDIA PaaS system will need to meet.

3.3.2.1 Requirements on NUBOMEDIA PaaS Platforms

While reading these requirements, it is worth mentioning to keep in mind the fact that, the PaaS is somewhere in between IaaS and SaaS. So not only will developers access services running in the cloud but also they need to be able to ship and manage their applications via the platform. In addition to standard software engineering principles, this results in a couple of requirements for PaaS:

- As many different developers should be served, the platform needs to be multi-tenant. This not only relates to a clear separation of all data structures, account data, and service orchestration of all tenants but also includes ways of managing and setting up new developers easily such as in the case of evaluation periods.
- Zero footprint requirement which means that the user should not have to install any components locally. This, of course, implies that the NUBOMEDIA PaaS platform is fully designed for the Web supporting access via the browser and, ideally, mobile devices.
- Specifically and needless to say that scalability plays a major role in Real-Time applications. The PaaS must be able to scale at least horizontally the deployed application. But the PaaS platform should not just be able to grow in case of higher demands but also to shrink during off-peak periods. In other words the platform should be elastic in the sense that resources are managed wisely depending on demand and it is easy to scale up and down.
- Given that developers will implement and ship their specific applications via the PaaS platform, a programming model as well as support for popular frameworks and programming languages are required. This also includes mechanisms for software lifecycle management such that application providers can easily and safely develop, test and ship their applications via the PaaS platform.

Given these requirements and the numerous open source versions of popularly used PaaS solution, the intent of NUBOMEDIA is not to yet develop a new PaaS solution from scratch, but utilise and enhance an existing open source solution to fit our demands and requirements for PaaS for a Real-Time applications.

For this project, we decided to use the open source version of Red Hat's OpenShift - OpenShift Origin v3. This is the community driven implementation of the PaaS which

introduces the use of Docker and Kubernetes. In addition to the advantages offered by Docker and Kubernetes, there are several reasons, why we decided for this software component:

- Provision of over the top functionalities for augmented deployment, orchestration and routing.
- Provision of groups of Docker containers called pods that simulate a single virtual machine with a single IP address, shared file system and common security settings. This ability provides the advantage to deploy scalable applications with shared local resources.
- Extensibility. Through the REST APIs to core components services, alternative implementations are applicable.
- Flexibly linked feature; Other PaaS platforms are limited to only web frameworks and rely on external services for other component types. OpenShift Origin v3 provides more application topologies. Developers can have a project in which many components are linked with each other. In this manner, we can link any two arbitrary components together through exporting and consumption of environment variables. This way, we can link together any two components without having to change the images they are based on.

Given the above advantages, we can build anything on OpenShift by offering a platform built on containers that allows building entire applications in a repeatable lifecycle. To address the complexity of deploying on Open Shift Origin, Fraunhofer FOKUS has developed and abstraction layer – NUBOMEDIA PaaS Manager on top of Open Shift Origin that uses a web based GUI interface and REST APIs to interact with Open Shift micro services in building, deploying, scaling and hosting NUBOMEDIA multimedia applications.

Remember the fact that, the PaaS is somewhere in between IaaS and SaaS, thus the NUBOMEDIA PaaS Manager also provides connection and interaction with OpenBaton (an ETSI NFV complaint network orchestrator) for requesting and instantiating virtual network resources (e.g. media servers)

3.3.3 NUBOMEDIA outcomes

The PaaS Manager and PaaS API extensions to the OpenShift Origin PaaS have been implemented, tested and pushes to the fhg-fokus-nubomedia github repository [FOKUS_GITHUB]. The associated software artifacts have ben releases as part of the NUBOMEDIA Open Source Software Community and shall be disseminated along the project. At the time of this writing, a scientific paper has been submitted to a relevant journal featuring our PaaS architecture.

3.3.4 References

Referencing websites
[5G_EX] https://5g-ppp.eu/5gex/

[5G_SONATA] http://www.sonata-nfv.eu/

[CLOUDIFY] http://getcloudify.org/

[FOUNDRY] https://www.cloudfoundry.org

[SHIFT] https://www.openshift.com

[INFO_WORLD] http://www.infoworld.com/article/2871935/application-development/infoworlds-2015-technology-of-the-year-award-winners.html

[APP_ENGINE] https://cloud.google.com/solutions/architecture/webapp

[HEROKU] https://www.heroku.com

[SOLUTIONS_REVIEW] 2016 Comparison Matrix Report - Platform as a Service (PaaS) http://solutions-review.com/dl/2016_Solutions_Review_Buyers_Matrix_Cloud_PaaS_NYE16.pdf

[FOKUS_GITHUB] https://github.com/fhg-fokus-nubomedia

Referencing papers or books
[NIST] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," NIST Special Publication 800-145, September 2011.

[FORRESTER_REP] S. Ried, "Multiple PaaS Flavors Hit The Enterprise," Forrester, Tech. Report., August 2012, http://www.forrester.com/Multiple+PaaS+Flavors+Hit+The+Enterprise/fulltext/-/E-RES78101

[GARTNER_MODEL] Y. V. Natis, "Gartner Reference Model for PaaS" Gartner, Tech. Rep., September 2011, http://www.gartner.com/id=1807820

3.4 Media monitoring in cloud infrastructures

Monitoring is a terminology that depends a lot on the context it refers but in the case of computer systems is a process to of becoming aware of the current state of a computer system. Current IT system are very complex and managing and monitor the state of them are needed real-time systems to gather metrics, store them in a time-series database and interpret the readings. Monitoring a system means finding out about complications before they develop into problems and keep high SLAs. They also provide historical data for developing predictive solutions.

In general monitoring begins with reading a specific metric, store it and measure against thresholds. Important about the monitoring metrics is the resolution of the values they are stored. If metrics are stored at 5 minutes intervals they will not help a developer troubleshoot the latency on his multimedia application. Is possible that latency to

increase and when the monitoring system is measuring the value to be decreased and in this way deceives the developers that has the impression that latency is in parameters.

Monitoring of a computer systems involves multiple fields like storage systems, real-time processing, statistics and data analysis. A monitoring system is usually a set of software components that collects data, process data and then presents the data to an user. Time-series data are basically chronologically ordered lists of data points. A metric and his properties in a data structure are stored in so called time-series databases.

Benefits of using monitoring systems are immediate, they provide early detections of problems, helps business to make decisions. Some examples are Netflix[INFONETFLIX] which monitors all the company cloud infrastructure and provide open-source tools for large scale monitoring. By leveraging data from monitoring, Netflix understands the quality of the service is provided to customers and what aspects of the service needs improving.

Data collection in a monitoring system is responsible with gathering meaningful data from systems by using agents. The collection is a continuously process that reads the data at intervals that are not affecting performance of the monitored system. Then data is stored at specific intervals, the previously mentioned resolution or granularity. Common intervals are 1, 5, 15 and 60 minutes. Some systems have fixed granularity but others have a hybrid approach where data points are stored for a period in a fixed granularity but later are in a different. This hybrid approach helps saving costs for storing the metrics.

Collecting data can be active or passive. An active monitoring is adding a cost to the monitoring for example by doing a ping request is added a small network load. So intervals of collection are critical to not increase the network costs. A passive monitoring is reading statistics from current flow of data without adding any cost for collecting the data.

Now that we have an understanding what a monitoring system is we'll describe in next section current monitoring technologies.

3.4.1 Description of current SoTA

This section will describe current monitoring solutions with time-series databases with their features and characteristics that we followed when we investigated to use them for NUBOMEDIA.

3.4.1.1 InfluxDB

Is a young open-source project that is building a time-series database which aims to solve the problem with scalability of large number of writes from different sources. Being built from scratch with latest technologies and specifically for time-series databases allows InfluxDB [INFLUXDB] to achieves his goals of high scalability and performance.

InfluxDB has powerful APIs that allows developers to push data via REST APIs and perform queries which are expressive and SQL-like.

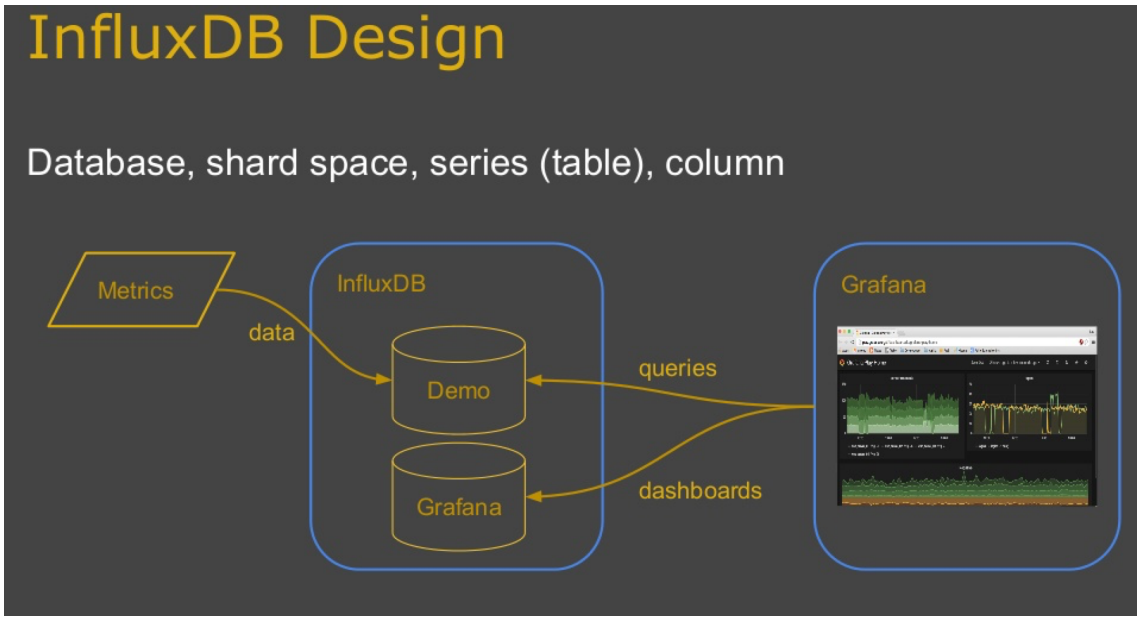


Figure 26 InfluxDB design

As shown on Figure 26, the design of database is distributed and data is sharded, where it borrowed concepts from other databases. Being a time-series database, most of the data is sharded by timestamp.

As shown in Figure 27 InfluxDB is managed through a web interface and most of the functions are available in the interface like creating new databases and performing queries and displaying the stored data. Data stored on InfluxDB can be tagged so queries can target aggregated data from multiple sources. All the data is indexed by tags and time.

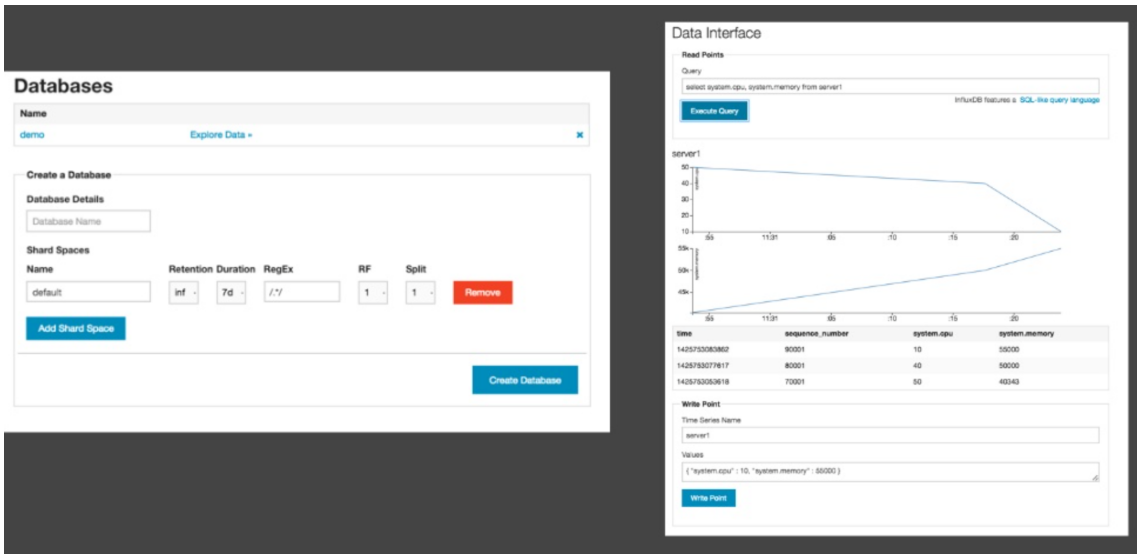


Figure 27. InfluxDB Web Interface for management

An issue with InfluxDB is that the database is distributed which makes it more complex so inherently harder to manage it in deployment scenarios and multiple systems should be monitored. This was the main reason why InfluxDB was not considered for NUBOMEDIA as it added a lot of complexity and the project is quite young(latest version is 0.9).

3.4.1.2 Graphite

From current solutions, Graphite [GRAPHDOCS] is one of the oldest and most used open source projects dedicated to monitoring. It was started in 2006 by an online agency company and released as an open source project in 2008 [MONITGRAPHITE]. Having a simple interface to push metrics lots of contributors were involved in Graphite by adding improvements to performance and addressing an use where results where needed in real-time. This is very important for NUBOMEDIA which real-time is the most important requirement for a multimedia application.

Many businesses use or used Graphite [MONITGRAPHITE] as their monitoring platform to store metrics and taking business decisions on the data it collected. Booking.com one of the Internet's busiest online travel agency is using Graphite as their primary system for storing performance data for systems, network and application. Graphite enabled them to store easily data from multiple sources and correlate data across all of them and understand the full impact of a code, infrastructure or feature change.

Other users of Graphite are GitHub, which manage software development pipeline of many commercial and open source projects. Their culture of data driven investigation powered by Graphite enabled them to interact with data in a manner that was visible for all employees of the company, from ops to marketing.

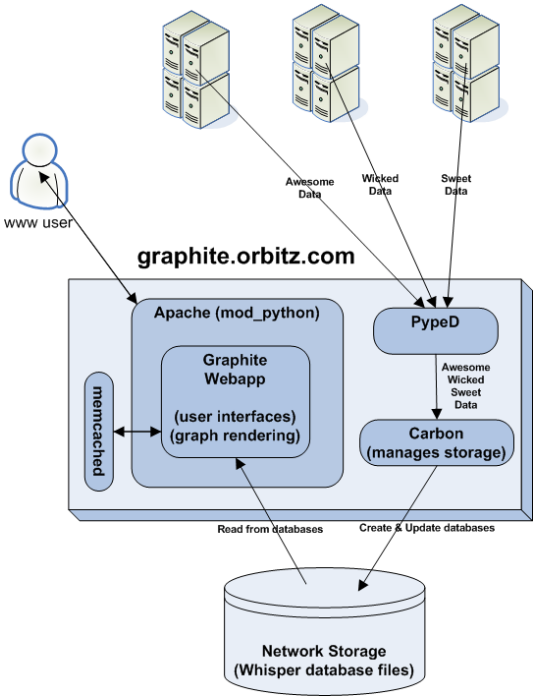


Figure 28. Graphite architecture

As shown on Figure 28, Graphite is composed from multiple components [MONITGRAPHITE]:

- Carbon: A Python based daemon that listens to the data and is storing it on Whisper storage. Metrics are submitted in a simple format with a name delimited by dots, a epoch timestamp and a value. Carbon can be installed on a different machine than rest of Graphite components so it can be used an optimized machine for Carbon which is CPU bound, compared to Whisper

which will require a server optimized for IO operations. Also, Carbon can aggregate values for use cases like monitoring a server with 8 cores and some operators will prefer an aggregated display of CPU usage.

- Whisper: is a time-series database that is storing the metrics to one-second precision. Each datapoint will be stored as float with an epoch timestamp. Is based on RRD and improved by allowing data to be stored at different intervals and backfilling old data. Each metric will create a file on the filesystem and an empty Whisper database will have prepopulated all the datapoints that will arrive for that metric with null values. These null values are replaced as data arrives and stored by Carbon. Retention of metrics are optimized by specifying the precision needed and period. For example 1sec metrics to be stored for 15 days, then datapoints to be roll up on at 1min interval. This is achieved by using an aggregation method which can be defined. Default method of aggregation is average of values. So for our example, from 60 datapoints for each second are aggregated in a single value, an average of those 60 values. If needed aggregation method can be sum, min, max or last.
- Graphite Web application: A Django/Python application that is serving the data from Whisper to users via a WEB interface or REST APIs. Web application is using Memcached to optimize the performance of Graphite. Memcached is an in-memory key-value store that stores the results form the backend.

Figure 29 displays how all Graphite components interact. Hosts and applications are pushing metrics to Carbon via TCP sockets or REST API. The metrics are stored on Whisper database and server to Graphite web application for display.

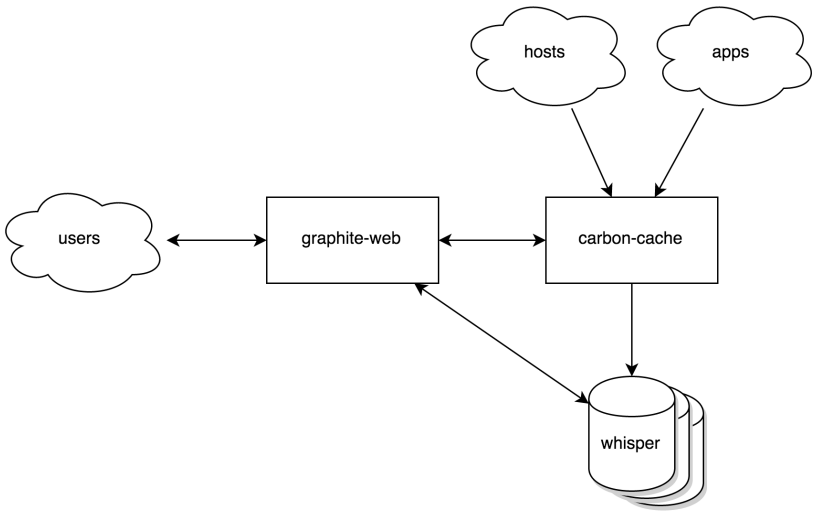


Figure 29. How Graphite components interact

Being a mature project and filling requirements needed for NUBOMEDIA project, we choose Graphite as the monitoring tool for storing metrics from applications and resource usage of servers, containers used on NUBOMEDIA platform.

3.4.1.3 Prometheus

Started in 2012 by SoundCloud company and released as an open source project, Prometheus [PROMETHEUS] is a monitoring system with a feature-rich time-series database with no reliance on distributed storage.

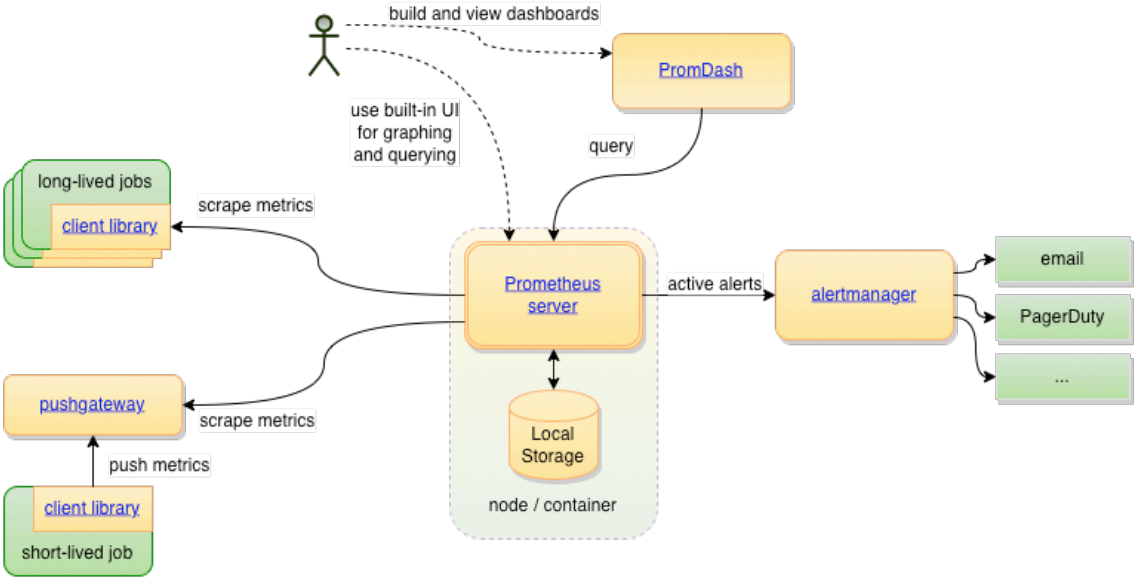


Figure 30. Prometheus architecture

Compared to previous systems, Prometheus uses a pull model over HTTP as shown inFigure 30 . The server is querying clients for data, in this way the server decides what to pull and when to pull the data. For example a server under heavy load can throttle better the network traffic as the clients will store the metrics until server is ready.

Prometheus has a rich data model (time series identified by metric name and key/value pairs) and with an advanced query system that can filter, group data based on labels from metadata. Prometheus query system can be accessed with a HTTP API.

Main disadvantage of Prometheus is that the events are not stored per events but they are preprocessed on the clients. For example, on Prometheus will be stored that for 1 minute a server had 300 requests and we'll not store each individual request. This limits the use cases where can be used, but for NUBOMEDIA requirements this was not the case.

The reason we didn't used Prometheus as monitoring platform for NUBOMEDIA is that is not supporting down-sampling and metrics storage can become costly for a large number of servers in long term. Down-sampling on monitoring is the process of removing entry-points at different intervals, for example if latency was stored initially at each second, after 15 days it can be downscaled and latency to be stored at each minute. In this way older data that is not used anymore will use less storage.

3.4.2 NUBOMEDIA approach beyond SotA

All the solutions mentioned at 3.4.1 have a granularity of data for 1, 5, 15 or 30 minutes, which is not satisfying the requirements of NUBOMEDIA where real-time multimedia applications are deployed and require close to real-time monitoring information.

Considering current monitoring tools offerings and the real-time requirements of NUBOMEDIA, we decided that Graphite and his flexible time-series database satisfies the requirements for the project and we integrated it in the platform. Also we fine-tuned the solution to be able to monitor metrics at a granularity that helps developers for real-time applications but also reduce the costs for storing the monitoring data.

3.4.3 NUBOMEDIA outcomes

We integrated Graphite into NUBOMEDIA platform and added on top of it an API for a better experience for developers to submit metrics. As part of the integration we optimized storage component to store time-series data at granularity that make the system useful for real-time applications.

The development was integrated in NUBOMEDIA as part of deliverable D3.2.
The optimizations performed will be presented to Distributed Systems scientific seminar from dec 2016: https://www.eed.usv.ro/SistemeDistribuite

3.4.4 References

Referencing websites

[INFONETFLIX] http://www.infoq.com/presentations/netflix-monitoring-system

[INFLUXDB] https://influxdata.com/time-series-platform/influxdb/

[GRAPHDOCS] https://graphite.readthedocs.org

[PROMETHEUS] http://prometheus.io

Referencing papers or books

[MONITGRAPHITE] J. Dixon, Monitoring with Graphite, O'Reilly Media, 2015

3.5 Deploying and installing media cloud infrastructures

OpenStack is a cloud operating system that provides support for provisioning large networks of virtual machines, pluggable and scalable network and IP management, and object and block storage. OpenStack is a complex system of components, each requiring expertise to deploy and manage. The system administrators require the same agility and productivity from their hardware infrastructure that they get from the cloud.

Infrastructure-as-Code (IaC) automates the process of configuring and setting up the environment (e.g., servers, VMs and databases) in which a software system will be tested and/or deployed through textual specification files in a language like Puppet or Chef [HUMBLE2010].

Given that testing and deploying a cloud computing platform like OpenStack requires continuous configuration and deployment of virtual machines, OpenStack makes substantial use of IaC, adopting both Puppet [PUPPET] and Chef [CHEF] to automate infrastructure management [JIANG2015].

More recently, OpenStack has started collaborating both with Chef [CHEF.IO] and Puppet [PUPPETLABS.COM] to create new means to configure and deploy fully-

functional OpenStack environments on bare-metal hardware, as well as on Vagrant virtual machines. The combination of a system management tool, like Chef or Puppet, and Vagrant can be used to setup a virtualized experimentation environment. Puppet Enterprise offers solutions to deploy OpenStack and scale it horizontally on demand.

3.5.1 Description of current SoTA

The current available technologies for system administration and management are the following:

- Chef https://www.chef.io/chef/
- Puppet Enterprise https://puppetlabs.com/puppet/puppet-enterprise
- AWS OpsWorks http://aws.amazon.com/opsworks/
- JUJU https://juju.ubuntu.com/

These tools provide domain-specific declarative languages (DSL) for writing complex system configurations, allowing developers to specify concepts such as "what software packages need to be installed", "what services should be running on each hardware node", etc. More recently, OpenStack has started collaborating both with Chef [CHEF.IO] and Puppet [PUPPETLABS.COM] to create new means to configure and deploy fully-functional OpenStack environments on bare-metal hardware, as well as on Vagrant virtual machines [AFFETTI2015]. The combination of a system management tool, like Chef or Puppet, and Vagrant can be used to setup a virtualized experimentation environment.

AWS OpsWorks [AWSAMAZONCOM] is an Amazon product that allows resources to be provisioned via the EC2 service and configured using Chef. Chef recipes can be applied at a number of defined lifecycle stages (Setup, Configure, Deploy, Undeploy, Shutdown) [MEYER2013].
OpsWorks allows deployments to be built as stacks composed of the individual components of the deployment, such as webservers and databases. Cookbooks can be retrieved from SVN and GitHub repositories, HTTP addresses and S3 buckets. Recipes can be executed directly in virtual machine instances, eliminating the need for a dedicated Chef server. OpsWorks is a propriety hosted service and can be used only with the Amazon EC2 cloud service.

Juju is an open-source service configuration and deployment tool developed by Canonical [JUJU]. It is limited to deploying systems running the Ubuntu Linux distribution. It is compatible with Linux containers, physical machines and cloud environments, such as Amazon EC2, Eucalyptus, OpenStack, HP Cloud and RackSpace. Juju service configuration scripts are called *charms* and are shared through a public catalogue called the charm store. The charms are stored on the Ubuntu Launchpad platform and can be written in any language that can be interpreted by Ubuntu [MEYER2013]. This makes it possible to use it in combination with Puppet or Chef, as Juju will just call the agent to run a script or to connect to a Puppet Master or Chef server. Currently it is only possible to assign one charm per instance. Instance specification parameters, such as memory, can be used as prerequisites for charms.

A number of commercial systems, such as IBM Tivoli, BMC Bladelogic Automation Suite, HP OpenView and Microsoft Center Configuration Manager are also available [MEYER2013]. Access to script catalogues for these systems, if they exist, is typically restricted to paying clients. Ad-hoc sharing of scripts can also occur on forums and mailing lists.

These are complex sysadmin tools that require:

- strong technical skills;
- specialized personal;
- complex setups;
- aimed to large enterprises.

The currently available FOSS solutions used are:

- Puppet open source - https://puppetlabs.com/puppet/puppet-open-source
- Chef Open source - https://github.com/chef/chef
- Saltstack - http://saltstack.com
- Bash scripts https://www.gnu.org/software/bash/
- Python application https://www.python.org/
- Ruby application https://www.ruby-lang.org/en/

The two most popular open source automated configuration management (CM) software tools all follow the declarative approach: Puppet and Chef. The scripts are developed in a collaborative manner on social coding sites and are shared through publicly accessible catalogues or community forums.

Puppet is an automated configuration management (CM) and service deployment system that has been in development since 2005. Puppet configuration scripts are referred to as *manifests*. It is implemented using the Ruby programming language, and hence requires an installation of Ruby to be present on machines that it manages. For cloud environments Puppet has a suite of command-line tools to start virtual instances and install puppet on them without logging in manually.
Puppet Labs provide a manifest sharing platform, Puppet Forge [FORGE] that is used by the community to promote re-use and collaborative development of manifests. Analysis of the manifests available on Puppet Forge indicates that the user base is focused primarily on Linux, especially Ubuntu and RHEL based systems.
The manifests available on Puppet Forge are open source and address a wide variety of service deployment and administration tasks. The manifests themselves are written either in a Puppet description language (a DSL using simplified Ruby), or directly in Ruby.
Two web interfaces are available for Puppet. The first, Puppet Dashboard, is developed by Puppet Labs and is used as the interface in the commercial version, although a community version with a reduced set of functionality is freely available. The second interface is Foreman, which has more detailed views and integrated support for compute resources, such as cloud service providers. Foreman is an open source project that is built and maintained by a community of volunteers with assistance from Red Hat.

Chef is an open source CM tool and framework developed by Opscode, a company founded in 2008 [EWANT2013]. Furthermore, it exposes APIs and libraries that can be used to build custom tools. The configuration scripts are called *cookbooks* that consist of *recipes*. Chef is built on a client-server model, where the Chef client connects to the Chef server to pull the cookbooks and to report the current state. The server is also the reference implementation of the Chef API. The client is written in Ruby and therefore depends on the Ruby runtime. Recipes can be implemented either in pure Ruby or a reduced Ruby DSL.
Chef is divided into multiple components. The multi-purpose command line tool Knife uses the Chef framework to facilitate system automation, deployment and integration. It provides command and control capabilities for managing virtual, physical and cloud resources.

Open source social coding communities, such as GitHub [DABBISH2012], are often used to share scripts for open source automated CM tools and encourage collaborative development.
For example, the "openstack-chef-repo" repository is an example project for deploying an OpenStack architecture using Chef, and the "puppetlabs-openstack" project is used to deploy the Puppet Labs Reference and Testing Deployment Module for OpenStack, as described in the project profile on GitHub [GITHUB-PUPPETLABS], [GITHUB-CHEF].

These solutions have the following limitations

- no gui tools.
- no solution for configuring OpenBaton and the PaaS manager

The modular architecture of these management tools enables the FOSS community to collaborate and share plug-ins to support any operating system and any boot sequence.

Puppet Labs and EMC collaboratively developed Razor, a next-generation physical and virtual hardware provisioning solution [PUPPETLABS.COM]. Puppet Enterprise with Razor automates every phase of the IT infrastructure lifecycle, from bare-metal to fully deployed applications. Razor will automatically deploy the correct operating system or hypervisor to the appropriate hardware by matching hardware profiles to a defined policy. This tool will provide unique capabilities for managing hardware infrastructure, including:

- Auto-discovered real-time inventory data for every hardware node, eliminating inefficient, error-prone manual processes;
- Dynamic image selection which allows to removing the need for manual intervention whenever there is a change in hardware configuration;
- Policy-based provisioning allowing specifying the desired state of each hardware node and its operating system, automatically tracks provisioning progress toward this state, and can even decide when to re-provision.
- RESTful open APIs and plug-in modular architecture which gives full programmatic control of the rules and models that govern operating system image selection and hardware provisioning.

FutureGrid [LASZEWSKI2010], now FutureSystems [FUTURESYSTEMS] is a project funded by NSF, USA that set out to develop an environment that researchers could use to experiment with cloud infrastructures and various kinds of HPC platform services (e.g., Hadoop). Initially, FutureGrid supported IaaS deployments through Nimbus, OpenStack, and Nebula and now the FutureSystems supports IaaS deployments only through OpenStack. FutureSystems is a well-known project allowing fast access to a functional cloud installation, focused to experiment the potential behavior of a specific application across different cloud providers.

OpenStack APIs has multiple SDKs for Ruby but none of them is a native one, making it not the best solution because of the capabilities that might not be implemented on the third party SKDs.

Using the OpenStack Python native APIs we can develop the NUBOMEDIA platform autonomous installer having access to all the capabilities the IaaS can provide in an easy and direct manner.

3.5.2 NUBOMEDIA approach beyond SoTA

For adapting to the specific requirements of NUBOMEDIA, we created a free open source autonomous installer based on Python that allows the deployment of the NUBOMEIDA platform on top of an IaaS based on OpenStack and a PaaS based on OpenShift.. For R7 we plan to further extend the autonomous-installer in order to support deployment of the NUBOMEDIA platform on other private and public clouds like Rackspace or AWS.

3.5.3 NUBOMEDIA outcomes

We published the autonomous installer [AUTONOMOUS-INSTALLER] on a public github repository[4] so that anyone who wants to use NUBOMEDIA on it's private data center to be able deploy it in just few minutes without having to run installation and configuration scripts. The results of task T3.5.2 will be disseminated by publishing a research paper at the DAS conference 2016 [DAS conference 2016].

3.5.4 References

Referencing websites

[FUTURESYSTEMS] https://portal.futuresystems.org/ ;
[CHEF] "Chef Software, Chef - IT Automation for Speed and Awesomeness", https://www.chef.io/chef/, 2015 ;
[PUPPET] "Puppet Labs, Puppet - IT Automation Software," http://puppetlabs.com/.
[CHEF.IO] https://www.chef.io/solutions/openstack/ ;
[PUPPETLABS.COM] https://puppetlabs.com/solutions/openstack/ ;
[GITHUB-PUPPETLABS] github.com/puppetlabs/puppetlabs-openstack ;
[GITHUB-CHEF] github.com/stackforge/openstack-chef-repo ;
[FORGE] forge.puppetlabs.com/ ;
[AWSAMAZONCOM] "AWS OpsWorks DevOps cloud application management solution", aws.amazon.com/opsworks/ ;
[JUJU] juju.ubuntu.com/ ;
[AUTONOMOUS-INSTALLER] https://github.com/usv-public/nubomedia-autonomous-installer .
[DAS conference 2016] http://www.dasconference.ro/ ;

Referencing papers or books

[EWANT2013] Ewart, J. , "Instant Chef starter", Birmingham: Packt, (2013).
[DABBISH2012] Dabbish, L.; Stuart, C.; Tsay, J. and Herbsleb, J.,"Social coding in GitHub: transparency and collaboration in an open software repository", *ACM 2012 Conference on Computer Supported Cooperative Work* (*CSCW '12)*, New York, NY, USA: ACM, (2012):1277–1286.
[JIANG2015] Jiang, Yujuan; Adams, Bram, "Co-evolution of Infrastructure and Source Code - An Empirical Study", *IEEE 12th Working Conference on Mining Software Repositories*, (2015): 45-55.
[HUMBLE2010] Humble, J. and Farley, D., "Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation", *1st ed. Addison-Wesley Professional*, 2010.
[MEYER2013] Meyer, Stefan; Healy, Philip; Lynn, Theo; Morrison, John, "Quality Assurance for Open Source Software Configuration Management", *15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, (2013): 454-461.

[4] https://github.com/usv-public/nubomedia-autonomous-installer

[LOPEZ2014] Lopez, Luis, et al. "Authentication, Authorization, and Accounting in WebRTC PaaS Infrastructures: The Case of Kurento." *Internet Computing, IEEE* 18.6 (2014): 34-40.
[AFFETTI2015] Affetti, L., Bresciani, G., and Guinea, S., "aDock: A Cloud Infrastructure Experimentation Environment based on OpenStack and Docker", *IEEE 8th International Conference on Cloud Computing (CLOUD 2015)*, June 27 - July 2, 2015, NY, USA, (2015): 203-210.
[LASZEWSKI2010] Laszewski, G. Von; Fox, G. C.; Wang, F.; Younge, A. J.; Kulshrestha, A.; Pike, G. G.; Smith, W.; Voeckler, J.; Figueiredo, R. J.; Fortes J. et al., "Design of the Futuregrid Experiment Management Framework," *Gateway Computing Environments Workshop*, (2010): 1–10.

4 RTC media server technologies

4.1 RTC media servers

4.1.1 RTC media servers: an overview

In the context of RTC systems, a media server is a software stack (i.e. a subsystem) that provides media capabilities. Media capabilities are all the features related to the media itself (i.e. related with the bits of information representing audio and video). These may include:

- Media transport following specific RTC protocols
- Media recording and recovery
- Media transcoding
- Media distribution
- Media analysis
- Media augmentation and transformation
- Media mixing
- Etc.

In the RTC area, sometimes media servers are frequently named depending on the specific capabilities they provide. For example, we speak about streaming media servers when dealing with media servers capable of streaming multimedia content, or we say recording media server when we have the capability of recording the audio and video received, etc.

The concept of media server is quite old and has been used in the RTC literature for long time. In the last few years, RTC media servers are living again a new gold era due to the emergence of WebRTC technologies. Nowadays, WebRTC is the main trending topic in the multimedia RTC area and this is why WebRTC needs to be supported by any RTC media server wishing to play a role in the market. WebRTC services commonly require the presence of media servers, which are very useful when creating services beyond the standard WebRTC peer-to-peer call model. Some common examples of server-side media plane elements that could be used in WebRTC include:

- WebRTC media gateways – typical on services requiring protocol or format adaptations (as happens when integrating WebRTC with IMS)
- Multi Point Control Units (MCUs) and Selective Forwarding Units (SFUs) – used to support group communications
- Recording media servers – helpful when one needs to persist a WebRTC call
- Etc.

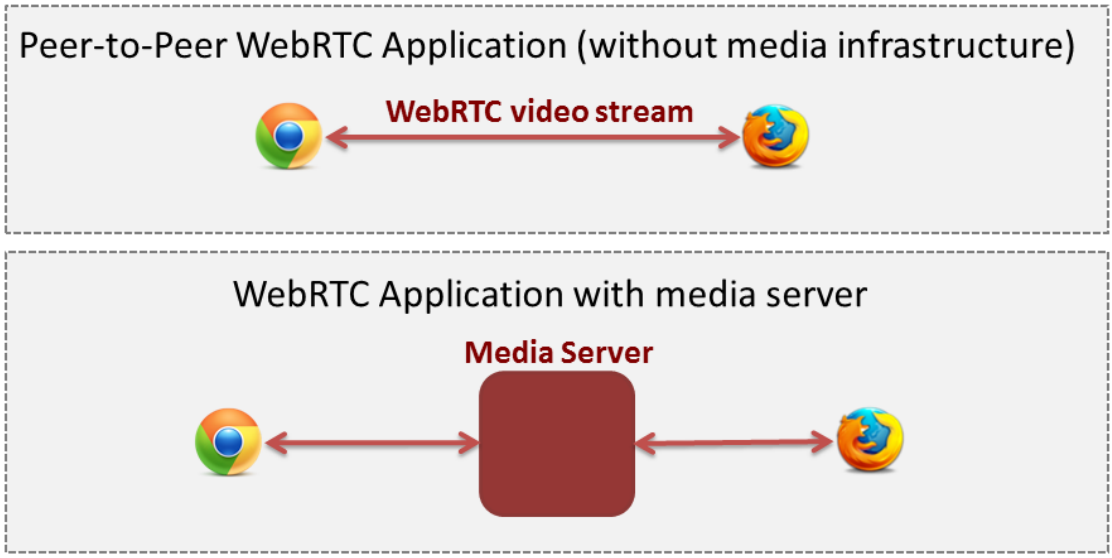


Figure 31. RTC applications, in general, and WebRTC applications, in particular, may use two different models. As shown at the top, the peer-to-peer model is based on direct communication among clients. This model provides minimum complexity and latency, but it also has important limitations. At the bottom, the infrastructure-mediated model, where a media server is mediating among the communicating clients. This model has higher latency and complexity, but it makes possible to enrich RTC services with additional capabilities such as transcoding (i.e. interoperability), efficient group communications (i.e. MCU or SFU models), recoding and media processing.

Currently, there are a large number of WebRTC capable media servers out there. Just for illustration, Table 6 shows the list of the me most popular open source ones.

Server	Description
Medooze[5]	Meedoze it is a MCU Video Multiconference Server with WebRTC support. It can be integrable in any SIP infrastructure. It provides web streaming and recording of conferences supported, custom layouts and continous presence, and web management interface.
Licode[6]	The Licode media server provides MCU videoconference rooms and recording features totally compatible with the WebRTC standard.
Jitsi[7]	Jitsi Videobridge is a WebRTC compatible Selective Forwarding Unit (SFU) that allows for multiuser video communication. Unlike expensive dedicated hardware videobridges, Jitsi Videobridge does not mix the video channels into a composite video stream. It only relays the received video flows to all call participants.
Intel CS for WebRTC Client SDK[8]	The Intel CS for WebRTC Client SDK builds on top of the W3C standard WebRTC APIs to accelerate development of real-time communications (RTC), including broadcast, peer-to-peer communications, and conference modes.
telepresence[9]	The open source media server of the Doubango initiative which provides group communications through a MCU mixing model and some additional features which include recording, 3D sound, etc.
Janus[10]	Janus is a WebRTC Gateway developed by Meetecho conceived to be a general purpose one. As such, it doesn't provide any functionality per se other than implementing the means to set up a WebRTC media communication with a browser, exchanging JSON messages with it, and

[5] http://www.medooze.com/
[6] http://lynckia.com/licode/
[7] https://jitsi.org/
[8] https://software.intel.com/en-us/webrtc-sdk5
[9] https://code.google.com/p/telepresence/
[10] https://janus.conf.meetecho.com/

	relaying RTP/RTCP and messages between browsers and the server-side application logic they're attached to. Any specific feature/application is provided by server side plugins that browsers can then contact via the gateway to take advantage of the functionality they provide. Example of such plugins can be implementations of applications like echo tests, conference bridges, media recorders, SIP gateways and the like.
Red5[11]	Red5 is an open source media server for live streaming solutions of all kinds. It is designed to be flexible with a simple plugin architecture that allows for customization of virtually any VOD and live streaming scenario. Built on the open source Red5 Server, Red5 Pro[12] allows you to build scalable live streaming and second screen applications.

Table 6. Open source software WebRTC media servers.

In the proprietary arena, there are currently many vendors offering different types of WebRTC Media Servers providing the above mentioned features, a non-exhaustive list containing some of them (in no particular order) is the following:

- The Dialogic Power Media[13] product line offers different types of WebRTC capable media servers providing all the common media server capabilities (i.e. transcoding, MCU, recording, etc.)
- Some of the Radisys MRF[14] products claim to offer common media server capabilities enabled for WebRTC endpoints.
- Oracle, rebranding the acquired Acme Packet media server product line, is also offering a product line[15] containing different types of WebRTC middleboxes including media servers and session border controllers.
- Italtel seems also to have adapted its media server[16] product line for providing WebRTC capabilities.
- Flahsphoner also provides now a WebRTC capable media server[17] for group communications.
- The NG Media media server seems also to have introduces WebRTC[18] endpoint capabilities.
- Ericsson offers through its Web Communication Gateway[19] different WebRTC media server capabilities.

[11] http://red5.org/
[12] https://red5pro.com/
[13] http://www.dialogic.com/en/landing/webrtc.aspx
[14] http://www.radisys.com/products/mrf/mpx-12000/
[15] http://www.oracle.com/us/industries/communications/oracle-enterprise-web-rtc-wp-2132263.pdf
[16] http://www.italtel.com/en/media-center-eng/press-room/item/italtel-launches-embrace-webrtc-solution
[17] http://flashphoner.com/webrtc-streaming-server-for-live-broadcasting-and-webinars/
[18] http://n-g-media.com/ng-media-reveals-the-support-of-webrtc/?lang=en
[19] http://www.ericsson.com/us/ourportfolio/products/web-communication-gateway

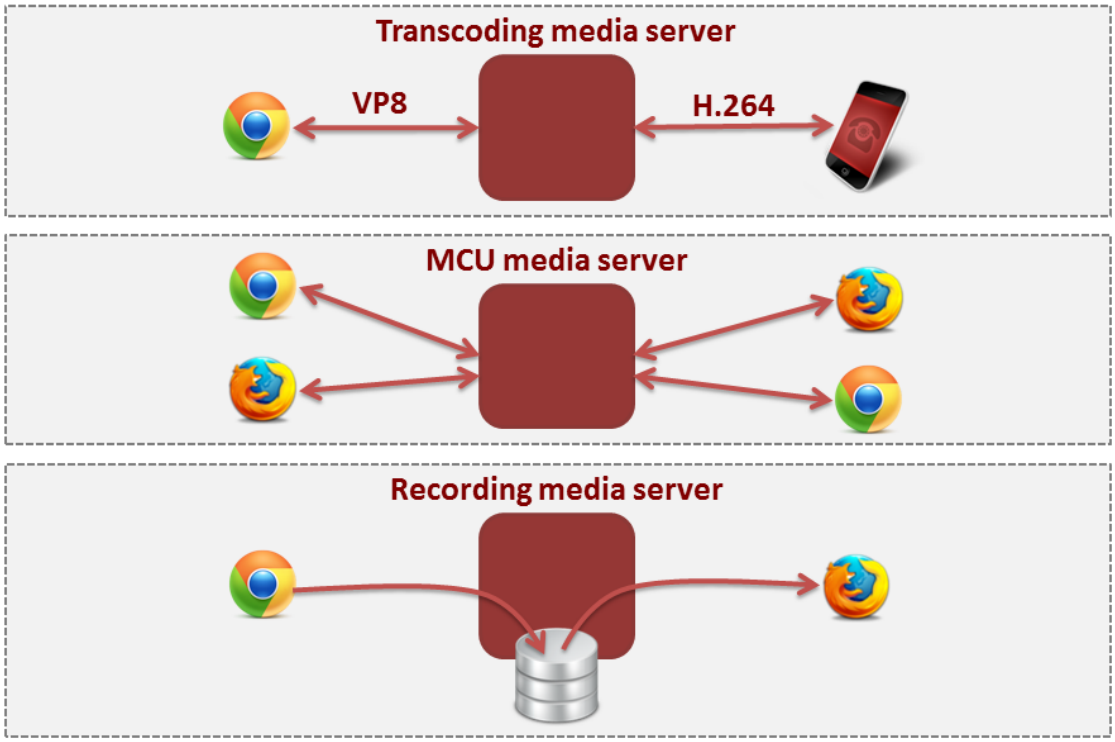


Figure 32: Media capabilities provided by state-of-the-art media server include: transcoding (top), group communications (middle) and archiving (bottom).

However, as NUBOMEDIA's vision shows, there are many interesting things we can do with the media beyond the basic three capabilities enumerated above (i.e. transcoding, group communications, recording). Why not enrich it using augmented reality? Why no analyze it using computer vision or deep speech analysis? Why can't we blend, replicate or add special effects to the media as it is travelling? These kinds of capabilities might provide differentiation and added value to applications in many specific verticals including e-Health, e-Learning, security, entertainment, games, advertising or CRMs just to cite a few.

Due to this, NUBOMEDIA requirements cannot be satisfied with any of the above mentioned solutions. Hence, for the execution of the project we need a more flexible technology where advanced processing capabilities could be plugged seamlessly. For this, we have re-architected the only open source media server enabling such types of capabilities: Kurento Media Server (KMS).

KMS vision is based on transforming traditional media server technologies, which are tied to a set of specific features, into modular software having the ability of extending the provided features just adding further plug-ins. In this direction, KMS is just a container providing a number of basic building blocks for implementing low level operations that provide plumbing capabilities enabling to move media from one place to another. Modules are just specialized classes that are capable of receiving and sending media to other modules through that plumbing.

Due to this vision, KMS is a flexible technology capable of providing all types of features one can image for the media, which is a clear advantage and justification for NUBOMEDIA choosing it. However, KMS also has drawbacks. The first one is paid in terms of complexity. For example, the software complexity, in terms of source code lines, classes, files, dependencies or any other metric you may wish to use, of KMS is significantly higher to the ones shown by the rest of open source media server

technologies described above. The second one is on performance. The required degree of flexibility makes the architecture to require on all modules the provision of capabilities that may be memory and CPU consuming, but which are not always used on a given specific application.

4.1.2 Description of current scientific and engineering SoTA

There is not a formal definition of what a media server is and different authors use the term with different meanings. In this document, we understand that a media server is just the server side of a client-server architected media system. In the same way DD.BB. (Data Base) servers provide data persistence capabilities to WWW applications, media servers provide multimedia capabilities to media-enabled applications.

Following this definition, media server technologies emerged in the 90's catalyzed by the popularization of digital video services. Most of these early media servers were specifically conceived for providing multimedia content distribution and transport in two main flavors: streaming media servers and RTC (Real-Time Communication) media servers.

Streaming media servers [DASHTI2003, LAURSEN1994, LI2013] provide media distribution capabilities through transport protocols designed for reliability that tend to relax latency requirements [SEUFERT2014]. This means, among other things, that these protocols typically recover from packet loss in the network through iterative retransmissions. The QoS provided through them is convenient for one-way applications such as VoD (Video on Demand) or live streaming, where the media information travels from sources to destinations but there is no real-time feedback communication.

RTC media servers [SCHULZRINNE1999, LU2010] in turn, are designed for bidirectional communications. Due to this, the transport protocols they implement are designed for low latency and packet retransmissions are not always useful for recovering form losses. In these services, the full duplex media information exchange is used to provide conversational interactions among users. Due to this, this type of media servers are typical in audio and video conferencing systems. In this document, we concentrate our attention on this latter type of media servers.

During the last two decades, RTC media servers evolved through different types of standards. One of the most remarkable ones is H.323[THOM1996], where the media server is performed by an architectural module called MCU (Multipoint Control Unit). The IMS (IP Multimedia Subsystem) architecture also standardized a generic media server function as the MRF (Media Resource Function) [KOUKOULIDIS2006]. Other standardization bodies worked in the same direction: the IETF (Internet Engineering Task Force), with specifications such as MGCP, MSCML, MSML or the JCP (Java Community Process) with API standards such as the Jain MEGACO API or the Media Server Control API. All is can be observed, all these efforts were concentrated on the standardization of the media server control interfaces more than on generating architectural recommendations or implementation guidelines

Most of these standards were issued by telecommunication operators and vendors to solve their very own needs. Due to this, RTC media servers remained as niche technologies used only on those areas. However, in the last few years, the emergence of WebRTC [LORETO2012] is bringing RTC services to mainstream WWW and mobile

developers worldwide. WebRTC enables high quality real-time multimedia communications on the Internet in a standardized, universal and interoperable way. WebRTC makes it possible for developers to incorporate RTC as an additional feature of WWW and mobile applications in a seamless way. As a result, WebRTC is permeating into many application domains including e-learning, e-health, CRM (Customer Relationship Management), gaming, robotics, IoT or video surveillance, just to cite a few.

Plain client WebRTC implementations just enable peer-to-peer communications. However, in the last few years users demand for richer services is exploding. This is making (Web)RTC media servers to become a critical asset for the success of applications. In particular, most available RTC media servers are designed and optimized for providing one of the following capabilities:

- Group communication capabilities: These refer to the ability of enabling groups of users to communicate synchronously. Most group videoconferencing services require them.
- Media archiving capabilities: These are related to the ability of recording multimedia streams into media repositories and of recovering them later for visualization. Services requiring communication auditability or persistence use them.
- Media bridging: This refers to providing media interoperability among different network domains having incompatible media formats or protocols. These are used, for example in WebRTC gateways, which interconnect WebRTC browsers with and legacy VoIP systems, and on IMS architectures.

Following this, in this document we introduce a next generation (Web)RTC media server that contributes to the state of-the-art by following a holistic architectural approach, meaning that it has been designed for providing, in an integrated way, all types of server-side media capabilities. This includes the three specified above but also further ones, such as augmented reality, media content analysis or arbitrary media processing; which enable novel use cases for rich person-to-person, person-to-machine and machine-to-machine communications. The next sections are devoted to specifying how Kurento contributes to these areas in detail.

4.1.2.1 Modularity in media server architectures

In software development, the concept of module is typically used for referring to a specific functional block that is hidden behind an interface. Following this definition, many modern RTC media servers are modular [AMIRANTE2014]. However, beyond the intuitive idea of a module, we are interested in the stronger concept of modularity [BALDWIN2001].

From a developer's perspective, an RTC media server is just a framework (i.e. a system providing capabilities to a system designer for programming application on top.) When bringing the concept of modularity to frameworks, the main idea is to split systems into smaller parts (i.e. modules) so that there is strong cohesion within modules and loose coupling among them. Moreover, for having full modularity they should comply with the following additional properties:

- Isolation: when modules operate together in a system, the internal status of a module should not directly affect the status of others.
- Abstraction: the internal state of a module should be hidden for both the system designer and for other modules. In software, abstraction is typically achieved

through interfaces that limit how the module interacts with the external world through a limited set of primitives.
- Composability: those abstract interfaces should enable components to recombine and to reassemble in various combinations to satisfy specific users requirements. In other words, modules should behave as building blocks and the system designer's role is to create the appropriate interconnection topology among them to provide the desired logic.
- Reusability: if a module provides a specific capability, any system requiring such capability might reuse the module without the need of re-implementing again the capability.
- Extensibility: the platform should provide seamless mechanisms for creating and plugging additional modules extending the exposed capabilities.

Modularity brings relevant benefits for developers including cost reduction, shorter learning times, higher flexibility, augmentation (i.e. adding a new solution by merely plugging a new module), etc. On the other hand, a downside to modularity is that low quality modular systems are not optimized for performance. This is usually due to the cost of putting up interfaces between modules.

In this context, and to the best of our knowledge, there is a single scientific reference dealing with modular RTC media servers: the Janus WebRTC Gateway [AMIRANTE2014]. Being Janus indeed modular, it does not comply with all modularity requirements, in particular in what refers to composability (i.e. Janus modules provide specific application logic and cannot be assembled among each other) and reusability (i.e. in Janus, different modules need to provide similar capabilities due to the lack of composability).

4.1.2.2 Media servers for group communications

Since their early origins, group communications were one of the most popular applications of videoconferencing services [ELLIS1991]. Due to this, today many RTC media servers concentrate on providing such capability [AMIRANTE2014, GROZEV2015, NG2014]. In current state-of-the-art, there are two widely accepted strategies for implementing group communications on RTC media servers [WERTERLUND2016]: media mixing and media forwarding.

Media mixing is performed through the Media Mixing Mixer (MMM) Topology, in which multiple input media streams are decoded, aggregated into a single output stream and re-encoded. For audio, mixing usually takes place through linear addition. For video, the typical approach is to perform video downscaling plus composite aggregation to generate a single outgoing video grid. Hence, a participant in a group RTC session based on MMM sends one media stream (its own audio and/or video) and receives one media stream (the mixed audio and/or video of all participants). As MMM were the common topology on H.323, MMM are sometimes informally called MCUs in the literature. MMM are useful when client devices have bandwidth or CPU restrictions. However MMM have several drawbacks. The first is that they implement CPU intensive operations making difficult to media server infrastructures to scale. The second is that, due also to these operations, they may increase the infrastructure latency degrading he overall QoE (Quality of Experience).

Media forwarding is typically performed through a Selective Forwarding Middlebox topology. RTC media servers implementing this topology are sometimes called Selective Forwarding Units (SFU) in the literature. An SFUs clones and forwards (i.e.

routes) the received encoded media stream onto many outgoing media streams. This means that a participant in a group RTC session based on SFUs sends one media stream (its own audio and/or video) and receives N-1 media streams (the audio and/or video of the rest), being N the total number of participants. For the sake of simplicity, RTC media servers implementing the Media Switching Mixer (MSM) topology might also be called SFUs. SFUs and MSM share most of its properties being the only difference that each SFU outgoing stream is uniquely mapped to one of the incoming streams. In the RTP jargon, this means that an SFU just projects to each of its outputs a given input SSRC and the only allowed operation is to switch on or off that projection. However, MSMs may change the mapping of the outgoing SSRCs among different incoming SSRCs through a switching procedure. This makes MSM to be more complex to implement, but in exchange, they provide bigger flexibility for implementing features such as Last N/dominant speaker or simulcast without a significant performance decrease.

In current state-of-the-art, RTC most media servers just implement one of the above mentioned topologies [AMIRANTE2014, GROZEV2015, NG2014] and do not provide flexible mechanism for using the rest. Some solutions [AMIRANTE2014] enable the ability to use different topologies as pluggable modules, so that, different applications may use different topologies by consuming different module capabilities.

4.1.2.3 Transparent media interoperability

Transcoding media servers exist since long time ago due to the need of achieving interoperability among incompatible media systems [AMIR1998, AHMAD2005]. Different media systems tend to have different requirements and, due to this, specific formats and codecs were conceived during the years for satisfying them. In addition, the existence of different commercial interests in the multimedia arena contributed to the emergence of a multiplicity of incompatible codecs [SINGH2014]. As a result, transcoding RTC media servers have been traditionally necessary as soon as a service requires interoperability among different communication domains.

This situation has become even more complex with the arrival of WebRTC given that developers typically need to interoperate WebRTC services with legacy IP infrastructures and with the phone system [BERTIN2013, AMIRANTE2013]. Due to this, many RTC media servers provide transcoding capabilities. However, for managing them and create interoperable services, developers need to explicitly manage format and codec conversions. This is in general a very cumbersome process which is error prone and requires deep knowledge about low level details of media representation.

4.1.2.4 Systems and tools for advanced media processing

Multimedia processing technologies have been used in RTC during the last 20 years for applications such as media compression, speech recognition, or DTMF (Dual Tone Multi Frequency) detection [COX1998]. However, in the last few years, the spectrum of media processing capabilities has been enlarged thanks to the emergence of novel techniques, which include Computer Vision (CV), Augmented Reality (AR) or Computer Generated Imagery (CGI). The volume of scientific publications in those areas is currently overwhelming with novel tools and algorithms emerging almost daily. However, most of these results do not arrive to be used in real applications. Probably, the main reason for this is complexity. Leveraging most of these results typically require deep understanding on low level media details and there is a lack of tools and architectures enabling their seamless integration into the frameworks used by common developers for creating applications.

In the area of CV, for example, there are relevant initiatives for creating software tools and frameworks suitable for simplifying the development and integration efforts of CV technologies. These include libraries such as OpenCV [PULLI2012] , VLFeat [VEDALDI2010] or SimpleCV [DEMAAGD2012]. All these, simplify significantly the task of managing CV algorithms as software artifacts. However, at the time of integrating CV mechanisms into RTC applications, and very particularly into WebRTC applications, they do not provide the appropriate level of abstraction and the appropriate capabilities. As a result, the number of details and complexities to manage by developers is so high that using CV technologies in RTC applications is impractical for most typical scenarios. However, many previous works demonstrate that enabling a seamless convergence of RTC and CV might open new horizons in many application domains including video surveillance [LIMNA2014] or video sensor networks [FIDALEO2004]

When considering AR technologies [CARMIGNIANI2011] the situation is not so different. There is a plethora of technologies, libraries and tools for creating AR applications [AJANKI2011, OLSSON2011, MACINTYRE2011]. However, in most of them, AR is an isolated experience that happen on the user's. As a result, AR technologies are rarely integrated into RTC services.

4.1.2.5 Generalized multimedia: media beyond audio and video

Traditionally, multimedia communication services have been based on the transmission of audiovisual information. However, in the last few years, there is an increasing need for generalizing this notion toward a multisensory multimedia model. Multisensory multimedia typically involves several human senses beyond sight and hearing. Multisensory multimedia may also leverage multiple types of sensors beyond cameras and microphones. Multisensory multimedia streams are currently being used in different application domains such as haptic RTC interfaces [ZHANG2014] virtual reality interfaces [SUTCLIFFE2003] or entertainment [ZHOU2004].

Due to this, novel RTC media standards are providing mechanisms for low-latency transmission of arbitrary sensor data. In particular, the WebRTC standardization bodies have defined the DataChannel protocols and interfaces for such purpose [LORETO2012]. DataChannels are currently being used as a transport mechanism for peer-to-peer content distributions [NURMINEN2013] and for low-latency transport of chat text messages. However, there are very few initiatives implementing multisensory services leveraging DataChannels support and even less enabling full DataChannel support in media infrastructures.

4.1.2.6 Developer tools

The objective of RTC media servers is to make possible for developers to create applications leveraging media capabilities. Due to this, RTC media servers expose interfaces typically in the form of APIs. Indeed, APIs are the mechanism preferred by developers for assembling different types of capabilities into applications. However, programming with APIs is just a part of what developers need to do for creating applications. Many studies show that a relevant fraction of the development effort is devoted to diagnosing problems and fixing software bugs [MEYER2014, MAALEJ2009, ROEHM2012]. A critical ingredient for simplifying these efforts is the provision of just-in-time visual information of software status at runtime. Due to this, most development frameworks and IDEs (Integrated Development Environment) are shipped with runtime debugging tools. However, in the case of RTC media servers, the

distributed nature of applications joined to the inherent real-time nature of the software logic, makes most of these tools useless. Moreover, from the perspective of RTC media server vendor, such runtime information is not typically considered as relevant, reason why most available solutions simply disregard this problem.

	Jitsi	Licode	Telepresence	Janus	Medooze	Red5	**Kurento**
Transport							
WebRTC	Yes	Yes	Yes	Yes	Yes	No	Yes
RTP	No	No	No	Yes(P)	Yes	No	Yes
RTSP	No	No	No	No	No	No	Yes
HTTP	No	No	No	No	No	No	Yes
Archiving							
File Recording	Yes(P)	Yes	NA	Yes(P)	Yes	NA	Yes
HTTP Recording	No	No	NA	No	NA	No	Yes
File playing	Yes(P)	Yes	NA	Yes(P)	Yes	NA	Yes
HTTP playing	No	No	NA	No	No	No	Yes
Play Seek	No	No	No	No	No	Yes	Yes
Media Interoperability							
Transcoding	No	No	No	Yes(P)	Yes	No	Yes
Agnostic	No	No	No	No	No	No	Yes
Architecture							
Modular	No	No	No	Yes	No	No	Yes
Composable	No	No	No	No	No	No	Yes
RTP Topologies							
SFU	Yes	Yes	Yes	Yes	No	No	Yes
MSM	No	No	No	No	No	No	Yes
MMM	No	No	No	Yes(P)	Yes	No	Yes
Media Processing							
CV/VCA	No	No	No	No	No	No	Yes
AR	No	No	No	No	No	No	Yes
Alpha mixing	No	No	No	No	No	No	Yes
Multisensory							
DataChannels	Yes	Yes	NA	Yes	NA	No	Yes
Synchronized metadata	No	No	No	No	No	No	Yes
Development Tools							
Instrumentation	No	No	No	No	No	No	Yes
Inspector	No	No	No	No	No	No	Yes
Cloudification							
QoS metrics	No	No	No	NA	No	No	Yes
Modular config	No	No	No	NA	No	No	Yes
Development APIs							
Media Server API	No	No	No	NA	Yes	No	Yes
XMPP		No	No	No	No	No	No
SIP	No	No	NA	Yes(P)	No	No	No

Table 7. This table compares the features of the NUBOMEDIA media server (aka Kurento) with some of the most popular open source solutions in the area of real-time-media infrastructures. Cells marked as "Yes" indicates that the feature is supported off-the-shelf. "Yes(P)" indicates that the feature requires an external plug-in or capability. "No" means that the feature is not supported. "NA" indicates that we have not found the appropriate information for evaluating whether the feature is supported. This table is a best effort performed basing on incomplete documentation provided by the cited projects and may contain mistakes.

4.1.2.7 Adaptation to cloud environments

The cloudification of RTC services is currently a very hot topic due to the benefits it brings to network operators and service providers. Most telcos are already working on the direction of IMS virtualization for providing novel scalable solutions of signaling functions [CARELLA2014]. This trend is also permeating into WebRTC services, where PaaS (Platform as a Service) clouds are becoming one of the main exploitation mechanisms [LOPEZ2014, RODRIGEZ2016]. Current state-of-the-art in cloud computing point to a full automation of the runtime tasks through capabilities such as autoscaling, deployment and provisioning and service orchestration.

Due to this, RTC media servers need to adapt for complying with the requirements that these novel clouds technologies demand. In particular, for successful cloud deployment the following features seem to be necessary. First, to provide monitoring capabilities enabling the management of autoscaling algorithms. Second, to incorporate the

appropriate logic for brokering media server capabilities among applications in a seamless and robust way. Third, to offer suitable mechanism for automating he deployment and provisioning of media server instances. Fourth, to hold the appropriate traffic assessment and control mechanism guaranteeing the appropriate QoS adaptation to virtual networking environments. To the best of our knowledge, no current state-of-the-art media servers provide all these capabilities.

4.1.3 NUBOMEDIA approach beyond SotA

In relation to the above-specified topics shortcomings and limitations, the NUBOMEDIA approach is to re-architecture KMS technology for creating a NUBOMEDIA Media Server enabling the following progresses beyond SotA.

4.1.3.1 Modularity in media server architectures

One of the most relevant contributions of NUBOMEDIA is that in the project we have designed and implemented a full modular architecture complying with all the above-mentioned modularity requirements. In particular, to the best of our knowledge, our media server is the only one enabling full composability and reusability of components thanks to its unique design based on the notion of Media Elements and Media Pipelines.

4.1.3.2 Media servers for group communications

In this area, the contributions of NUBONEDIA are straightforward: following an holistic approach, our media server offers all topologies in an integrated way. This enables a novel feature that we could define as “topology as an API”, in the sense that application developers do not need to be aware of the complexities or internal details of MMM, MSM or SFUs. They just need to use the media server API modularity features to interconnect the appropriate building blocks in the appropriate way for satisfying application requirements. The media server takes care of translating the API calls into the corresponding topologies combinations and of implementing the required optimizations and media adaptations enabling the appropriate mechanism (i.e. MMMs, MSMs or SFUs.) to be used at the right places.

4.1.3.3 Transparent media interoperability

A very relevant contribution of NUBOMEDIA is to introduce a novel capability, that we call the agnostic media, which performs fully transparent transcoding and format adaptations for developers over a wide variety of codecs. To understand how this happens, observe that the modularity requirements specified above mandate our modules to be composable. This means that application developers should be able to freely interconnect them for generating the desired media processing logic. From a practical perspective this means that, for example, a WebRTC endpoint module receiving video encoded with the VP8 codec can be connected with an RTP endpoint sending H.264 encoded video. Of course, this type of module pipelining requires the appropriate transcodings which, in our case, take place without even requiring developers to know they are happening: the agnostic media detects the need of a transcoding and performs it transparently in the most optimal possible way.

4.1.3.4 Systems and tools for advanced media processing

In this context, the main contribution of NUBOMEDIA is to provide a seamless mechanism an a technological stack enabling the integration of advanced media processing capabilities, and very particularly CV and AR technologies, as modular components into a RTC media server. This mechanism has two advantages. The first is that the abstraction properties of our architecture are suitable for hiding all the complex details of such technologies enabling non-expert application developers to control them

through simple and intuitive interfaces. The second is that the composability of our APIs makes possible to assemble such advanced capabilities among each other and with other RTC features making possible to generate innumerable combinations of rich media processing topologies reusing the implemented algorithms and mechanism as basic building blocks for system creation.

4.1.3.5 Generalized multimedia: media beyond audio and video

The main contribution of NUBOMEDIA in this area is the integration of WebRTC DataChannels as a first citizen of a RTC media server. This has two implications. First, that our WebRTC implementation fully supports the DataChannel transport and negotiation mechanisms, for which we have been one of the main contributions to the corresponding GStreamer repositories. Second that our modular architecture enables all modules to manage three types of information: video, audio and data. Hence, module implementations may leverage arbitrary sensor data received from the external word for enhancing media processing or may generate arbitrary data streams from media semantics. This opens novel possibilities for application developers such creating multisensory Augmented Reality services where the augmentation logic is controlled by external sensors, or implementing mechanisms translating audiovisual streams into machine-understandable streams exposing rich semantic information about the media content.

4.1.3.6 Developer tools

Another relevant contribution of NUBOMEDIA is to introduce a number of mechanisms and APIs enabling the inspection of the runtime behavior of the RTC media server. In particular, we have created a visual debug tool capable of depicting graphs representing the modules involved in an application, their internal state and their interaction topology. This tool has the objective of minimizing the effort required by developers for understanding application runtime behavior and for gathering the appropriate diagnose information in case of problems.

4.1.3.7 Adaptation to cloud environments

The NUBOMEDIA WebRTC media server has been designed to comply with all the above-specified requirements for adapting to cloud environments. In particular, the following specific capabilities have been enabled. First, we have created a metric publishing system enabling custom QoS metrics to be gathered through a coherent API basing on W3C WebRTC Stats API specification draft. Second, the media server includes a Media Resource Broker function (MRB) that performs load balancing and pipeline scheduling among media server groups basing on operating system and internal QoS metrics. Third, we have created a modular configuration mechanism enabling modules to expose all their parameterizable properties through them, so that the cloud provisioning mechanism may modify and adapt them to the specific needs of each deployment. Fourth, the media server provides a traffic shaping mechanism suitable for avoiding overloading the virtual network interfaces of cloud environments.

4.1.4 NUBOMEDIA outcomes

The main outcome of NUBOMEDIA in the area of RTC media servers has been to create a full re-architected version of Kurento Media Server (KMS). For this version we have successfully implemented a fully compliant WebRTC protocol stack basing on GStreamer capabilities. In addition, we have created the appropriate artifacts for complying with the specific requirements of the project and for addressing all the above mentioned progresses beyond SotA. In particular, at the time of this writing, the following objectives have been successfully achieved:

Modularity in media server architectures
The modularity properties of the media server have been fully implemented. Media Element and Media Pipeline mechanism have been defined and specified through interfaces, as described in the corresponding deliverable devoted to the NUBOMEDIA Media Server.

Media servers for group communications
At the time of this writing, KMS provides MSM capabilities in a natural way just through Media Element connectivity interfaces. This mechanism makes possible to generate arbitrary and dynamic topologies where the any incoming SSRC can be forwarded to any outgoing SSRC seamlessly and freely by application developers in correspondence with their application logic. In addition, MMM are also naturally provided through the Composite hub, as described in the NUBOMEDIA Media Server deliverable document. At the time of this writing, an SFU capability providing simulcast support has been also developed and is made available through a Media Element interface. This capability is currently under incubation for improving its performance and stability.

Transparent media interoperability
At the time of this writing, KMS provides a fully functional agnostic media capability suitable for adapting media codecs and formats in a transparent way, as described in the NUBOMEDIA Media Server deliverable document.

Systems and tools for advanced media processing
The modular mechanism enabling the integration of CV and AR capabilities as modules of KMS has been fully developed and validated.

Generalized multimedia: media beyond audio and video
The integration of DataChannel support into the KMS WebRtcEndpoint is fully functional and validated. The support on KMS for data tracks on Media Element interfaces is fully functional and validated. The support for synchronous metadata buffers is also fully functional.

Developer tools
In the context of the NUBOMEDIA project, we have created an instrumentation mechanism enabling to access KMS runtime information. This mechanism is based on the ServerManager interface, which exposes an API suitable for obtaining the list of media capabilities (i.e. Media Elements and Media Pipelines) that are in place at a given time. This API also provides subscription primitives to be notified when runtime changes take place. In addition, runtime QoS metrics are exposed my Media Elements through different types of interfaces, as specified in the NUBOMEDIA Media Server deliverable.

Adaptation to cloud environments
As part of the results of this project, we have created several mechanisms for adapting KMS to automated cloud environments. At the time of this writing, many KMS capabilities, including the WebRtcEndpoint, provide custom QoS metrics suitable for evaluating in a precise way the load status of the server. In addition, the modular configuration mechanism has been fully developed and validated enabling cloud provisioning mechanisms to automate fine grained parameterization of KMS execution.

In addition to this, the following results are expected for the last year of the project:

Media server for group communications
We plan to evolve our current SFU architecture towards a full featured and validated SFU supporting differentiated quality through simulcast and scalable video codec mechanisms supporting stream multiplexing based on unified plan.

Developer tools
We plan to create a visual graphical user interface consuming KMS instrumentation APIs for representing runtime information in a seamless and intuitive way.

Adaptation to cloud environments
We plan to perform relevant research on the definition and gathering of metrics suitable for improving autoscaling automation mechanism. This may include the creation of predictive algorithms suitable for detecting media server overload before it happens, both for network and CPU.

All in all, these results have been useful for generating the following outcomes.

- Creation of a strong and successful Open Source Software community around KMS technologies.
- Contributions to relevant open source software projects including GStreamer, OpenWebRTC and Chrome.
- Publication of one journal paper and two conference papers, as shown in the NUBOMEDIA Communication and Dissemination Results deliverable.
- Submission of two journal papers, as shown in the NUBOMEDIA Communication and Dissemination Results deliverable.
- Contributions to relevant industrial conferences as shown in the NUBOMEDIA Communication and Dissemination Results deliverable.

As part of the last year of the project, we plan to generate the following outcomes:

- Creation of a startup leveraging KMS for creating an innovative business model for RTC in the cloud.
- Submission of, at least, one scientific publication to a top journal in the area of multimedia tools and applications presenting the final KMS architecture and benchmarks on its internal workings.

4.1.5 References

[AHMAD2005] Ahmad, Ishfaq, et al. "Video transcoding: an overview of various techniques and research issues." Multimedia, IEEE Transactions on 7.5 (2005): 793-804.

[AJANKI2011] Ajanki, Antti, et al. "An augmented reality interface to contextual information." Virtual reality 15.2-3 (2011): 161-173

[AMIR1998] Amir, Elan, Steven McCanne, and Randy Katz. "An active service framework and its application to real-time multimedia transcoding." ACM SIGCOMM Computer Communication Review. Vol. 28. No. 4. ACM, 1998.

[AMIRANTE2013] Amirante, Alessandro, et al. "On the seamless interaction between webRTC browsers and SIP-based conferencing systems." Communications Magazine, IEEE 51.4 (2013): 42-47.

[AMIRANTE2014] Amirante, A., et al. "Janus: a general purpose WebRTC gateway." Proceedings of the Conference on Principles, Systems and Applications of IP Telecommunications. ACM, 2014.

[BALDWIN2001] Baldwin, Carliss Young, and Kim B. Clark. Design rules: The power of modularity. Vol. 1. MIT press, 2000., Sullivan, Kevin J., et al. "The structure and value of modularity in software design." ACM SIGSOFT Software Engineering Notes. Vol. 26. No. 5. ACM, 2001.

[BERTIN2013] Bertin, Emmanuel, et al. "WebRTC, the day after." Procs. 17 th International Conference on Intelligence in Next Generation Networks, ICIN. 2013.
Johnston, Adrian, John Yoakum, and Karam Singh. "Taking on WebRTC in an Enterprise." Communications Magazine, IEEE 51.4 (2013): 48-54.

[CARELLA2014] Carella, G. et al. Cloudified IP Multimedia Subsystem (IMS) for Network Function Virtualization (NFV)-based architectures. In Computers and Communication (ISCC), 2014 IEEE Symposium on, Vol. Workshops. 1–6.

[CARMIGNIANI2011] Carmigniani, Julie, et al. "Augmented reality technologies, systems and applications." Multimedia Tools and Applications 51.1 (2011): 341-377.

[COX1998] Cox, Richard V., et al. "On the applications of multimedia processing to communications." Proceedings of the IEEE 86.5 (1998): 755-824.

[DASHTI2003] Dashti, Ali Kim Seon Ho Shahabi, Roger Zimmermann, and Seon Ho Kim. Streaming media server design. Prentice Hall Professional Technical Reference, 2003.

[DEMAAGD2012] Demaagd, Kurt, et al. Practical Computer Vision with SimpleCV: The Simple Way to Make Technology See. " O'Reilly Media, Inc.", 2012.

[ELLIS1991] Ellis, Clarence A., Simon J. Gibbs, and Gail Rein. "Groupware: some issues and experiences." Communications of the ACM 34.1 (1991): 39-58.

[FIDALEO2004] Fidaleo, Douglas A., Hoang-Anh Nguyen, and Mohan Trivedi. "The networked sensor tapestry (NeST): a privacy enhanced software architecture for interactive analysis of data in video-sensor networks." Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks. ACM, 2004.

[GROZEV2015] Grozev, Boris, et al. "Last N: relevance-based selectivity for forwarding video in multimedia conferences." Proceedings of the 25th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video. ACM, 2015

[KOUKOULIDIS2006] Koukoulidis V, Shah M (2006) The IP multimedia domain: service architecture for the delivery of voice, data, and next generation multimedia applications. Multimedia Tools and Applications 28(2):203-220.

[LAURSEN1994] Laursen A, Olkin J, Porter M (1994). Oracle media server: providing consumer based interactive access to multimedia data. In ACM SIGMOD 23(2):470-477

[LI2013] Li, B., Wang, Z., Liu, J., and Zhu, W. 2013. Two decades of Internet video streaming: A retrospective view. ACM Trans. Multime- dia Comput. Commun. Appl. 9, 1s, Article 33 (October 2013)

[LIMNA2014] Limna, Thanathip, and Pichaya Tandayya. "A flexible and scalable component-based system architecture for video surveillance as a service, running on infrastructure as a service." Multimedia Tools and Applications (2014): 1-27.

[LOPEZ2014] Lopez, L. et al. Authentication, Authorization, and Accounting in WebRTC PaaS Infrastructures: The Case of Kurento. Internet Computing, IEEE 18, 6 (Nov 2014), 34–40

[LORETO2012] Loreto, Salvatore, and Simon Pietro Romano. "Real-time communications in the web: Issues, achievements, and ongoing standardization efforts." IEEE Internet Computing 5 (2012): 68-73.

[LU2010] Lu, Yue, et al. "Measurement study of multi-party video conferencing." NETWORKING 2010. Springer Berlin Heidelberg, 2010. 96-108

[NG2014] Ng, Kwok-Fai, et al. "A P2P-MCU Approach to Multi-Party Video Conference with WebRTC." International Journal of Future Computer and Communication 3.5 (2014): 319

[NURMINEN2013] Nurminen, Jukka K., et al. "P2P media streaming with HTML5 and WebRTC." Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on. IEEE, 2013.

[MAALEJ2009] Maalej, Walid, and Hans-Jörg Happel. "From work to word: How do software developers describe their work?." Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on. IEEE, 2009.

[MACINTYRE2011] MacIntyre, Blair, et al. "The Argon AR Web Browser and standards-based AR application environment." Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on. IEEE, 2011.

[MEYER2014] Meyer, André N., et al. "Software developers' perceptions of productivity." Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, 2014

[OLSSON2011] Olsson, Thomas, and Markus Salo. "Online user survey on current mobile augmented reality applications." Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on. IEEE, 2011

[PULLI2012] Pulli, Kari, et al. "Real-time computer vision with OpenCV." Communications of the ACM 55.6 (2012): 61-69.

[RODRIGEZ2016] Rodríguez, P, et al. Materialising a new architecture for a distributed fMCUg in the Cloud. Computer Standards Interfaces 44 (2016), 234 – 242.

[ROEHM2012] Roehm, Tobias, et al. "How do professional developers comprehend software?." Proceedings of the 34th International Conference on Software Engineering. IEEE Press, 2012.

[SCHULZRINNE1999] Schulzrinne H, Rosenberg J (1999) The IETF internet telephony architecture and protocols. Network, IEEE, 13(3):18-23,

[SEUFERT2014] Seufert, Michael, et al. "A survey on quality of experience of HTTP adaptive streaming." Communications Surveys & Tutorials, IEEE 17.1 (2014): 469-492.

[SINGH2014] Singh, Harjit Pal, et al. "VoIP: State of art for global connectivity—A critical review." Journal of Network and Computer Applications 37 (2014): 365-379.]

[SUTCLIFFE2003] Sutcliffe, Alistair. Multimedia and virtual reality: designing multisensory user interfaces. Psychology Press, 2003.

[THOM1996] Thom, G. (1996). H. 323: the multimedia communications standard for local area networks. IEEE Communications Magazine 34(12):52-56

[VEDALDI2010] Vedaldi, Andrea, and Brian Fulkerson. "VLFeat: An open and portable library of computer vision algorithms." Proceedings of the international conference on Multimedia. ACM, 2010.

[WERTERLUND2016] Westerlund, M. et al. "RTP Topologies". 2016. Internet Draft. Internet Engineering Task Force. Available online in https://tools.ietf.org/html/draft-ietf-avtcore-rtp-topologies-update-10

[ZHANG2014] Zhang, Longyu, Jamal Saboune, and Abdulmotaleb El Saddik. "Development of a haptic video chat system." Multimedia Tools and Applications (2012): 1-24.

[ZHOU2004] Zhou, Zhiying, et al. "Multisensory musical entertainment systems." MultiMedia, IEEE 11.3 (2004): 88-101.

4.2 Real-time Video Content Analysis on the cloud

Computer Vision (CV) is just one of the technological fields with higher growth and with growing popularity nowadays. Wikipedia defines computer vision as:

"... a field that includes methods for acquiring, processing, analyzing, and understanding images and, in general, high-dimensional data from the real world in order to produce numerical or symbolic information, e.g., in the forms of decisions.

Therefore, we can say that its objective is to determine what is happening in front of a camera and use that understanding to control a computer or system, or to provide people with new images that are more informative or aesthetically pleasing than the original camera images. Nowadays, Computer Vision applications can be used for a variety of use cases across many industries. Some examples of usage of computer-vision technology include video surveillance, biometrics, automotive, photography, movie production, Web search, medicine, augmented reality gaming, new user interfaces, and many more.

There are several examples which show the increasing popularity of this technology.

In 2014, Google launched the Project Tango Smartphone [TANGO] a project that uses computer vision and other technologies to achieve the goal of giving mobile devices a

human-scale understanding of space and motion. This technology tracks the motion of the device in 3-D and at the same time creates a map of the environment. According to the Project Tango website, the sensors in Project Tango devices allow the device to make over a quarter million 3-D measurements every second, updating its position and orientation in real time, combining that data into a single 3-D model of the space around you. Google also acquired Jetpac, a startup that has created a mobile application that uses computer vision to extract and analyze data from public Instagram photos. The data generated from the photos is used to create city guides. The Jetpac City Guides app is a visual guide for more than 6,000 cities worldwide.

But Google is not the only tech giant who is investing ion Computer Vision. Oculus VR, the Facebook subsidiary responsible for the latest virtual-reality revolution, just acquired the technology startup Surreal Vision [SURREAL]. Surreal Vision is focused on real-time 3D scene reconstruction – generating an accurate representation of the real world in the virtual world. Great scene reconstruction will enable a new level of presence and telepresence, allowing you to move around the real world and interact with real-world objects. Another example which shows the bet by computer vision has been the acquisition by Facebook of Instagram. Microsoft is also investing efforts in computer visions, as it is shown in one of their last research reports [MSIMGCLASS], where they claim that one Microsoft's team developed a system that can classify images of the ImageNet database with an error rate lower than human's.

But tech giants are not the only ones who are betting on this technology. It is increasingly common to find SMEs and startups that make the computer vision one of their core competences. Apart from those companies who have been already acquired by large players, we can find others who still survive on their own. Some examples are as follows

- Tire check [TIRE] who tries to detect whether the wheels of the vehicles have proper air pressure through a picture taken with your smartphone
- Tyris software [TYRIS] who uses CV to extract depth measures, track objects or identify people and implements some of these algorithms in the cloud.
- Seene [SEENE] who has developed a portfolio of advanced computer vision algorithms designed from the scratch for use on mobile devices in real-time applications.

Not only the industrial sector is very interested in CV, but also it is a very active research field. For example, if we search in Google Scholar the term "computer vision", we find 73,800 results only in 2015. To name a few:

- Motion compensation based fast moving object detection in dynamic background [ZHANG2015]
- Practical matters in computer vision [JAIN2015]
- FaceNet: A Unified Embedding for Face Recognition and Clustering [SCHROFF2015]

However, Computer vision is computationally expensive. Computer vision or Video content analysis algorithms (VCA) consume a prohibitive amount of resources in particular memory and CPU. That is the reason why many problems in the field cannot be solved properly. For example, complex algorithms require many functions operating in parallel on a single image to extract all relevant content, and in many cases the users require that the algorithm processes the images in less than 40 milliseconds.

The emergence of media servers in a cloud environment in the last years is considered to be one of the enabling technologies that allow some complex VCA algorithms to be used in real applications. Running in the cloud VCA algorithms that can inter-operate with each other and that are accessible through comprehensive yet simple APIs can open new horizons to computer vision applications.

Some of the benefits that cloud computing technology offers to computer vision are:

- Computational power and storage.
- Allows for having VCA algorithms or services on demand.
- Publishing algorithms for massive use through simple APIs

Nevertheless, as we can see in the rest of this section, it does not appear to exist any open source mature solution for running video content analysis (VCA) algorithms in media servers on the Cloud.

4.2.1 Description of current SoTA

In this section, we focus on the current available commercial products, FOSS solutions and scientific SoTA which run computer vision algorithms in the cloud. Prior to that, we will describe the main libraries which can be used as base technology for these cloud-based products or services. Although many companies or communities have partially based their developments in their own technology, however in many cases these solutions rely on a set of common libraries. The most popular libraries are as follows.

- **OpenCV (Open** Source **C**omputer **V**ision) [OPENCV] is a library of programming functions mainly aimed at real time computer vision. It is probably the most famous and used library. It is cross-platform and can run over different platforms such as Windows, Linux, android iOS and MacOS. The library is written in C and C++. It also contains wrappers for other languages such as Python and Java. OpenCV has a specific module which provides GPU acceleration in order to run more accurate and sophisticated algorithms in real-time on higher resolution images. This library is free for use under the open-source BSD-license. It helps developers to build different applications such as facial recognition, gesture recognition, segmentation, stereo vision, motion tracking, object identification and motion robotics.
- SimpleCV (Simple Computer Vision) [SIMPLECV] is an open source framework for building computer vision applications. With it, you get access to several high-powered computer vision libraries such as OpenCV – without having to learn about bit depths, file formats, color spaces, buffer management or matrix versus bitmap storage. This is “computer vision library made easy.” SimpleCV is free to use, and it is open source. It is written in Python, and runs on Mac, Windows, and Ubuntu Linux. It is licensed under the BSD license.
- PointCloud Library (or PCL)) [PCL] is a library for 2D/3D image and point cloud processing. The PCL framework contains numerous state-of-the art algorithms including filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation. These algorithms can be used, for example, to filter outliers from noisy data, stitch 3D point clouds together, segment relevant parts of a scene, extract key points and compute descriptors to recognize objects in the world based on their geometric appearance, and create

surfaces from point clouds and visualize them -- to name a few. PCL is released under the terms of the 3-clause BSD license and is open source software. PCL is cross-platform, (runs on Linux, MacOS, Windows, and Android/iOS).

- VLFeat [VLFeat] is an open source library which implements popular computer vision algorithms in image understanding, local features extraction and matching. It is written in C for efficiency and compatibility, with interfaces in MATLAB for ease of use, and detailed documentation throughout. It supports Windows, Mac OS X, and Linux. Some of the algorithms included are: Fisher Vector, VLAD, SIFT, MSER, k-means, hierarchical k-means, agglomerative information bottleneck, SLIC superpixels, quick shift superpixels, large scale SVM training, and many others.
- BoofCV is an open source Java library for real-time computer vision and robotics applications. Written from scratch for ease of use and high performance. Its functionality covers a wide range of subjects including, optimized low-level image processing routines, camera calibration, feature detection/tracking, structure-from-motion, and recognition. BoofCV has been released under an Apache 2.0 license for both academic and commercial use.
- CCV is another Computer Vision library which tries to be much easier to deploy, the code is better organized with a bit more dependency hygiene. It now runs on Mac OSX, Linux, FreeBSD, Windows, iPhone, iPad, Android, Raspberry Pi. In fact, anything that has a decent C compiler probably can run it. CCV includes different functionalities that will help the developer to build applications requiring image classifiers, frontal face detectors, cars and pedestrian detectors, text detectors, and object tracking. Its source code is distributed under BSD 3-clause License.

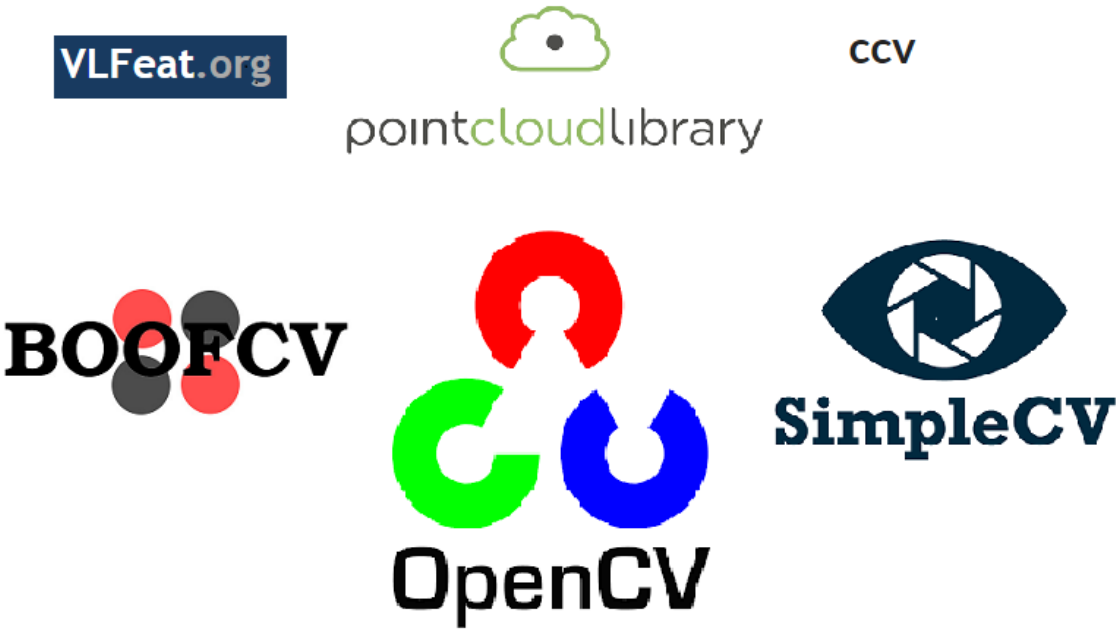


Figure 33: The most popular Computer Vision libraries

Defining better the technology and interfaces of cloud media servers will be crucial to see different options of computer vision libraries supporting cloud computing.

Commercial computer vision products and solutions in the cloud

First of all, it is important to highlight that every developer could create a media server with computer vision functionalities on the cloud **from the scratch**. For that purpose,

developers can use one of the **libraries** that we have just explained and **deploy** it in a cloud provider such as **Amazon web services**, **Microsoft azure** or **Google cloud**. Maybe, the most difficult part will be to send the video stream from the client side to the cloud provider. For the **video streaming** you can use **GStreamer** which is a multimedia framework written in C which can serve as a base to create many types of multimedia applications such as streaming media broadcasters, media players and video editors. Here [OPENCVAMAZON], you can find a post which explains how to run OpenCV services using PHP on Amazon cloud.

As for the commercial solutions, the **Microsoft Windows Azure** may be one of the most famous platforms which offer computer vision processing on the cloud. The computer vision part is based on a Microsoft's special project called Microsoft Project Oxford [MSOXFORD] which offers a set of services for understanding data and adding 'smart features' to your application. They offer a number of APIs related to:

- Faces: Face APIs provide state-of-the-art algorithms to process face images, like face detection with gender and age estimation, face recognition, face verification through which they can match the face of a person in two different images, face grouping through which they can classify a set of unknown faces based on similarities. A famous example of an application built using this API is the web site www.how-old.net which is a demo that detects faces and estimates the person age.
- Image analysis: with this API, the user can extract low-level features such image categories, dominant color and more from the input image's visual content.
- Get Thumbnail: given an input image this API generates a high quality image and stores an efficient thumbnail. This functionality uses smart cropping for thumbnails that are different than the aspect ratio of your original image to preserve the region of interest.
- OCR: Optical Character Recognition detects text in an image and extracts the recognized characters into a machine-usable character stream.

Figure 34: Microsoft Project Oxford

Visual Tools had the opportunity to create a simple application to test the use of this APIs over Microsoft Windows Azure to find some relevant limitations. Specifically, we have created an application to detect faces and the result is that it takes several seconds to process a single image. Another important thing is that we have not found how to use this API with a video stream.

The past December 2, 2015, Google launched a Computer Vision API [GVISIONAPI] over its cloud, Google Cloud platform. The Google Cloud Vision API allows developers to build powerful applications that can see, and more importantly understand the content of images. All the functionalities offers to the users are exposed through an easy-to-use REST API. So far the prices to use this API over the Google Cloud have not been set, Google offers a limited preview, since it is an API that just came out. The API offers the following features:

- **Label/Entity Detection** picks out the dominant entity (e.g., a car, a cat) within an image, from a broad set of object categories. You can use the API to easily build metadata on your image catalog, enabling new scenarios like image based searches or recommendations.
- **Optical character recognition**, to retrieve text from an image. Cloud Vision API provides automatic language identification, and supports a wide variety of languages.
- **Safe Search Detection**, to detect inappropriate content within your image. Powered by Google SafeSearch, the feature enables you to easily moderate crowd-sourced content.
- **Facial Detection** can detect when a face appears in photos, along with associated facial features such as eye, nose and mouth placement, and likelihood of over 8 attributes like joy and sorrow. The API does not support facial recognition and does not store facial detection information on any Google server.

- **Landmark Detection** to identify popular natural and manmade structures, along with the associated latitude and longitude of the landmark.
- **Logo Detection** to identify product logos within an image. Cloud Vision API returns the identified product brand logo.

Another example of commercial products in this field is provided by vision.ai [VISIONAI]. Vision.ai is a company who builds computer vision products and services. They offer a product call VMX vision server which provides object recognition, detection and tracking. They give you the opportunity to make a local installation or use a Docker container to deploy it on the cloud.

A good example that reinforces the theory that IT companies are investing heavily in computer vision is IBM. IBM acquired the Alcheamy company [ALCHEAMY] last March (2015). Among other things Alcheamy offers two different computer vision products running over Bluemix, the IBM cloud platform. These products offer:

- Face Detection and recognition API, through this services when an image or URL is provided, the system returns the estimated age, gender, and in the case of the celebrities the identities of the people in the photo.
- Image Tagging, through this service when an image or URL is provided, the system returns keywords summarizing scenes, objects and stylistic features. The API is capable of identifying 3D objects such chars, dogs, building, recognize scenes for example streets, stores, beaches, landscapes, mountains and detecting people and faces. We have tested this functionality through the demo of its web site, and we have to say that works pretty well.

However, these applications have also the same limitation already identified in other solutions, i.e. they do not give support to video streams. In addition, it takes a little time to process the images. In the same line as Alcheamy, we found Aylien [AYLIEN] which also provides an API for tagging images.

Another solution which brings computer vision services to the cloud has been launched by Meta Mind [METAMIND]. MetaMind was founded in December 2014 with the goal of providing artificial intelligence-as-a-platform that were easily accessible and easy to use. The company is focusing on the development of a new type of natural language processing, image understanding and knowledge base analytics platform. MetaMind utilizes groundbreaking technology called Recursive Deep Learning. The MetaMind API provides image classification, but in the same way as the others examples we can use the API only with single images.

Having reviewed commercial products and solutions we will review **FOSS solutions** here after (free and open source software solutions).

FOSS (Free Open Source Software) solutions in the cloud

There are some existing initiatives providing computer vision in the cloud. The first one which is worth to highlight is CloudCV [CLOUDCV] a large-scale distributed Computer Vision as a cloud service. The CloudCV is an open source project coming out of the Graphlab [GRAPHLAB] project in the summer 2012. The first version of CloudCV was available in summer 2013. This first version only provided an image stitching algorithm in order to combine multiple photographic images with overlapping fields of view to produce a segmented panorama or high resolution image. After this

first version the CloudCV has introduced new algorithms. Currently, the algorithms which can be executed are:

- Image stitching
- Object detection
- Object classification, through which different objects in the image can be automatically identified.
- Decaf: A deep convolutional activation feature for generic visual recognition.
- Face detection and recognition. In fact, they have a demo capable of finding important people on images.

The following figure depicts a basic scheme of CloudCV.

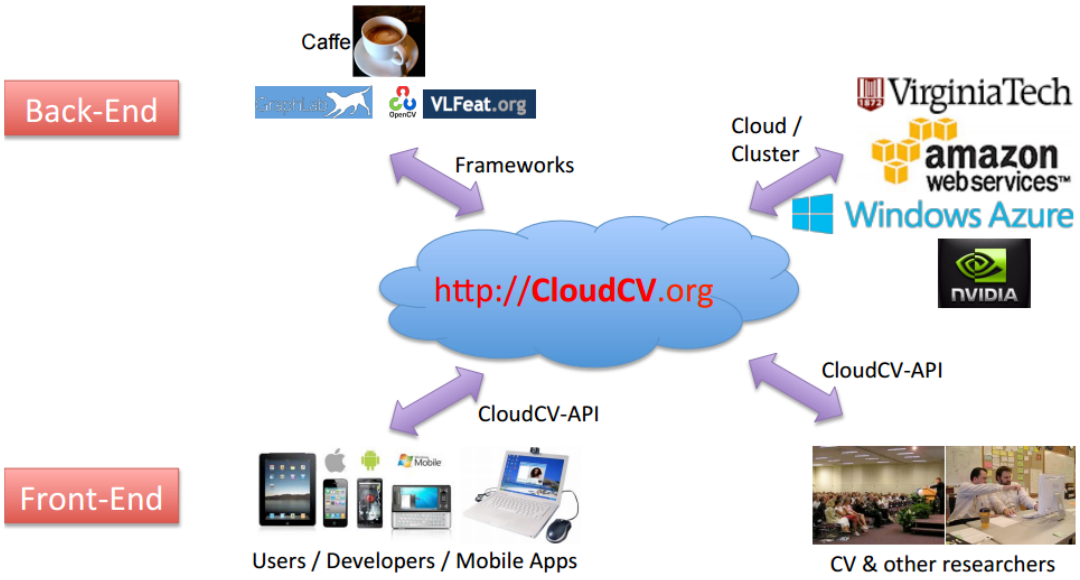


Figure 35: CloudCV Architecture

CloudCV provides "computer vision algorithms as a service" to researchers, students and app developers through its Matlab, python and web APIs. GraphLab, OpenCV, VLFeat and Caffe are the libraries and frameworks which provide a high level programming interface, allowing a rapid deployment of distributed machine learning algorithms. The deployment is done over different cloud platforms, Amazon web services [AWS], Windows Azure [AZURE] and Virginia Tech clusters [VIRGINIATECH].

In the following figure we can see an example of how this platform works. The web-servers are responsible for listening to incoming job requests and sending real-time updates to the user. A job scheduler takes these incoming jobs and distributes them across a number of worker nodes. Here, it is important to highlight that if the job scheduler detects a task requiring a lot of processing power, it will send it to a node with graphics processing units (GPU) based on Nvidia hardware in order to improve the performance of the task.

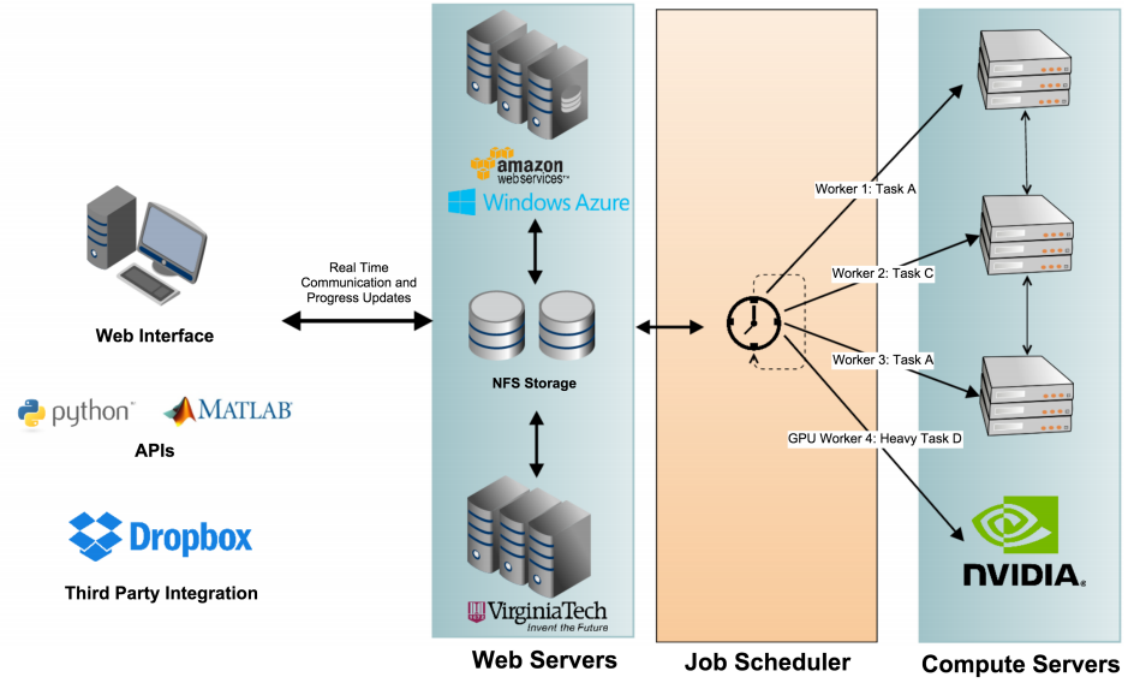


Figure 36: CloudCV backend

After reviewing the CloudCV platform, we are going to present another platform which tries to bring computer vision algorithms to the cloud. This platform is called **Rubix.io** [2]. At the time of this writing Rubix.io is still in beta. This software can be used through a Ruby API. A JavaScript API is under development. In this beta version, you can use through its API the following algorithms:

- Object Detection.
- Text recognition (OCR).
- An algorithm to detect Image similarities.
- They are currently working on a face recognition algorithm.
- Vibrand is a product for google glasses which allow you to collect images of brand logos and products and provide relevant information about them.

However, at the time of this writing Rubix.io does not seem under active development, e.g. more than a year has passed without any commit to its repository at Github.

Scientific publications

As we have explained on the previous section a lot of interesting research in computer vision is underway. A search in Google Scholar provides quite some results. In contrast, the number of scientific publications on computer vision AND cloud computing is much less. The most relevant ones are described here after.

Applications or prototypes on face recognition or license plate recognition over cloud are some of the research examples. In particular, one can find many examples of Face Recognition in the net. For instance, in the paper "*Face Recognition for Social Media with Mobile Cloud Computing*" [INDRAWAN2013] a cloud solution is proposed for face recognition using mobile devices. In this solution the mobile devices are in charge of detecting the face on the image. Once the face has been detected, the part of the image where the face has been detected is sent to the cloud service to perform the face

recognition. Another example of facial recognition in the cloud can be found in the paper "*Cloud-Vision: Real-Time Face Recognition Using a Mobile-Cloudlet-Cloud Acceleration Architecture*" [SOYOTA2013]. The challenge of this study lies with how to perform task partitioning from mobile devices to cloud and distribute computing load among cloud servers to minimize the response time considering communication latencies and server computing powers. However, in this challenge the image processing is carried out in the mobile devices or some internal servers which are found in the previous step to send information to the cloud. In fact, with the aim to avoid becoming very network-intensive, the system only sends metadata corresponding to the HAAR features of the face. This metadata will be used to make the matching between the info sent and the corresponding databases of faces. A further example that follows the line of using facial recognition in the cloud is explained in the paper "*Biometric Authentication and Data Security in Cloud Computing*" [MASALA2015]. This paper presents a new Cloud platform designed to support basic web applications and guaranteeing secure access. The platform is built using the OpenStack architecture, while the user authentication is based on an original biometric such as face or fingerprint recognitions. In the same way as the previous examples, the system does not send any image to the cloud platform, but the mathematical model computed by the biometric device.

Regarding License Plate Recognition, the paper "*Cloud Based Anti Vehicle Theft by Using Number Plate Recognition*" [GAETHA2014] presents a specific system for efficient automatic theft vehicle identification by using the vehicle number plate. The proposed algorithm follows the typical phases in license plate recognition: Vehicle identification, Extraction of number plate region, Recognition of plate characters and OCR. In this system the only part which is executed on the cloud is the OCR. Another example of a system based on the four steps on license plate recognition (LPR) mentioned above is an LPR in the cloud developed Visual Tools. It consisted of a vehicle identification system developed in the Itea project "Web Of Objects" [WOO]. The system was designed to operate without real-time requirements and only with images and not video streams.

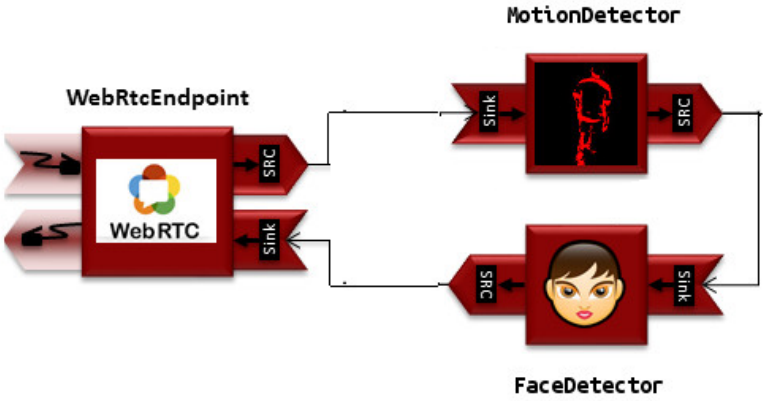
4.2.2 NUBOMEDIA approach beyond SotA

As we saw in the previous section we have seen that the computer vision is fast growing technology having large companies, SMEs and researchers behind the scene. However, we have realized that there is a considerable **lack of information** about **computer vision in the cloud**. This may be due to Cloud computing being a relatively new technology with high potential but it still needs to be further exploited in certain fields like computer vision. In the examples we have seen related to specific solutions, some applications process images outside of the cloud and then send the metadata or isolated images at a specific time but not as a continuous stream. Other systems work over images and not over a video stream and they do not support real-time requirements. As for the platforms that try to bring cloud-computing algorithms to the cloud, Rubix.io seems not to be very active. It does not provide many algorithms and does not specify if the platform also works for a video stream. In the same way, they do not mention anything about real-time results. Finally, CloudCV also provides too few computer vision algorithms. However, it seems have some capabilities to process video streams with real-time results. But even if there is somewhat more information on this platform, it is still scarce and we have many issues to solve such as whether you can apply various algorithms over the same video stream, whether they will develop more algorithms in the future or what is the communication standards supported for real-time

communication. Maybe, CloudCV can be a competitor for the NUBOMEDIA project. Therefore, the consortium will be alert to follow the evolution of that platform.

As a result of the previous review, NUBOMEDIA emerges as a remarkable solution to create a cloud platform specifically designed for real-time interactive multimedia services supporting VCA and Augmented Reality. Apart from the benefits of cloud technology on VCA algorithms, the main positive impacts that the NUBOMEDIA project can have are:

- Providing a simple to use API encapsulating and abstracting the complexities of VCA technologies. This API makes possible to create applications just by chaining individual media functions known as "Media Elements." The creation of such chains with different VCA services is suitable for tackling complex problems. For example, combining a motion detector media element and a face recognition media element can be done with the media pipeline (chain of media elements) shown below. In this way, every time the motion detector detects motion in an image the face detector will try to detect faces on that particular image.



- NUBOMEDIA enables many intelligent operations on video to be executed in real-time, an essential feature in many application areas such as video surveillance or video games.
- NUBOMEDIA is Free Open Source Software (FOSS). This guarantees that the platform is open and can be openly accessed in order to create a community of contributors. Therefore, the number of VCA services, elements or algorithms could be widely increased generating a big library of computer vision functionalities.
- Combining VCA with Augmented Reality will enrich the number of useful applications to build up.
- Ease of chain creation and composition which allow for fast development of customizable applications

In addition, building VCA systems is beyond the reach of people without deep understanding of image processing, modeling and expertise in computer programming. These prerequisites limit the ability of creating VCA-based applications to a reduced community of researchers and advanced programmers and serve as a high barrier of entry for developers without skills in these areas.

4.2.3 NUBOMEDIA outcomes

On this section we will describe the different outcomes of NUBOMEDIA related to Computer Vision. At the time of writing this deliverable (end of December 2015), the filters/algorithms supported by NUBOMEDIA is as follows.

- NuboFaceDetector
- NuboMouthDetector
- NuboNoseDetector
- NuboEyeDetector
- NuboEarDetector
- NuboTracker (Object Tracker)
- NuboVfence (perimeter intrusion detector)
- NuboMotion (Motion detector)

Apart from these filters, we expect to develop at least one filter more within the current release (January 31st, 2016).

Based on these filters and the characteristics of the NUBOMEDIA platform and taking into account the solutions and platforms described in this document, we can conclude:

- NUBOMEDIA contains a higher number of algorithms, compared to open source platforms and different business solutions studied on this deliverable.
- NUBOMEDIA allows for great flexibility in creating applications, by providing the ability to create different pipelines with the filters provided. This outcome is particularly relevant, since we have not found any similar capability on other open source platforms and applications studied.
- NUBOMEDIA does not only provide Computer Vision over the cloud, it also provides Augmented Reality and media capabilities enabling users to develop more complete solutions.
- Unlike other solutions analyzed above, NUBOMEDA gives support to video streams.

4.2.4 References

[TANGO] https://www.google.com/atap/project-tango/
[SURREAL] http://surreal.vision/
[TIRE] http://tirecheckapp.com/
[TYRIS] http://tyris-software.com/
[SEENE] http://seene.co/
[OPENCV] www.opencv.org
[SIMPLECV] http://simplecv.org/
[PCL] http://pointclouds.org/
[VLFEAT] http://www.vlfeat.org/
[BOOFCV] http://boofcv.org/
[CCV] http://libccv.org/
[OPENCVAMAZON] https://abhishek376.wordpress.com/category/computer-vision/
[MSOXFORD] https://www.projectoxford.ai/
[GVISIONAPI] https://cloud.google.com/vision/
[VISIONAI] http://vision.ai
[ALCHEAMY] http://www.alchemyapi.com/
[AYLIEN] http://aylien.com/
[METAMIND] https://www.metamind.io/

[AWS] https://aws.amazon.com
[AZURE] https://azure.microsoft.com/en-us/
[VIRGINIATECH] http://www.phys.vt.edu/facilities/clusters.shtml
[CLOUDCV] www.cloudcv.org
[RUBIX] www.rubix.io
[GRAPHLAB] http://graphlab.com/index.html
[WOO] http://www.web-of-objects.com

[ZHANG2015] Wei Zhang, et al. "Motion compensation based fast moving object detection in dynamic background" Volume 547 of the series Communications in Computer and Information Science pp 247-256.

[JAIN2015] Lakhmi C. Jain , et al. "Practical matters in computer vision" Volume 547 of the series Communications in Computer and Information Science pp 247-256. Computer Vision in Control Systems-2 Volume 75 of the series Intelligent Systems Reference Library pp 1-10

[SCHROFF2015] Florian Schroff, et al. " FaceNet: A Unified Embedding for Face Recognition and Clustering" Google. Computer Vision and Patern Recognition Conference 2015.

[INDRAWAN2013] Prasetyawidi Indrawan, et al. "Face recognition for social media with mobile cloud computing." *Journal on Cloud Computing*: Services and Architecture (IJCCSA),Vol.3, No.1, February 2013.

[SOYOTA2013] Prasetyawidi Indrawan, et al. "Cloud-Vision: Real-time Face Recognition Using a Mobile-Cloudlet-Cloud Acceleration Architecture" Computers and Communications (ISCC), 2013 IEEE Symposium.

[MASALA2015] G.L. Masala, et al. "Biometric Authentication and Data Security in Cloud Computing" International conference on security and management (SAM'2015)

[GAETHA2014] Gaetha B.G , et al. "Cloud Based Anti Vehicle Theft by Using Number Plate Recognition" International Journal of Engineering Research and General Science Volume 2, Issue 2, Feb-Mar 2014 ISSN 2091-2730.

4.3 Augmented Reality capabilities on real-time media servers

4.3.1 Description of current SoTA

In augmented reality (AR) digital objects are overlaid to the real world. The extended real world is viewed through a device having a display and camera. The rendered virtual objects on top of the real world view can contain etc. 2d/3d images, video, text, sound. The technology used in AR contains various enablers, but in extreme simplification of the process, only two things are required: recognition of the point where the object is rendered and the rendering of the object.

Starting at year 2000, the AR started to evolve strongly towards consumer applications, such as digital extension into a printed advertisement. The history of AR applications is more concentrated on mobile device applications for a personal user rather than a technology for video communication. During the year 2015 there has been remarkable

acquisitions of companies developing AR technology: Vuforia was bought by PTC (a global provider of technology platforms and solutions that transform how companies create, operate, and service the "things" in the Internet of Things, IoT) 2015, Metaio was bought by Apple 2015 and 13th Lab bought by Facebook (Oculus Rift Division). Also a company developing new era AR, Magic Leap, raised over 500 M USD for developing their AR technology. This builds expectations for wider adoption of the AR technologies. Same trend is supported from the vast development of new generation see trough AR glasses, including Sony SmartEyeGlass, MS Hololens and Magic Leap.
Since AR is researched quite a long time there are quite a few solutions for AR libraries that are either commercial or open source. Naturally the commercial libraries are often more advanced compared to the opens source libraries and during the last few years the gap between them has become larger. The commercial solutions for providing AR technology can be found from Table 8 and open source solutions from Table 9.

In general, for creating AR experience applications, native software for each platform is required. For this reason, all the libraries, especially the commercial ones, support many platforms, from Windows and iOS to all possible mobile operating systems. There is a slight difference is between commercial and open source versions that almost all the commercial products support also upcoming AR glasses. There exist some extensions of AR libraries that are trying to tackle the supporting of multiple platforms. Web based approaches are done by Cordova having a plugin from Wikitude and ARToolkit's JavaScript version JSARToolkit. Total Immersion's technology is used in In2AR that supports Adobe AIR's native application development for multiple platforms.

As earlier said, the simplest AR system would contain only tracking and recognition. To obtain any reasonable user experience for AR application actually the tracking of the position where the object will be rendered is required. The traditional approach for detecting the position of augmented item is to use specific markers, but it can also be a planar (any image). In mobile devices, this is supported often via multiple cues: gps, gyroscopes, compass, and most advanced use camera based 3D tracking (*Simultaneous Localization And Mapping,* SLAM). As the 3D range sensors will miniaturize they will become common as well for augmented reality purposes. The support for recognition and tracking of images, shapes, positions via different sensors and rendering possibilities also vary with in the existing AR libraries.

Commercial solutions for AR	Platforms	Special feature	
Total Immersion http://www.t-immersion.com/	Desktop, iPhone, iPad and Android Phones and Tablets). Adobe Flash	**Marker Less tracking as well as unmatched Face Tracking capabilities.** 2D and 3D content	D'Fusion is the world's most widely-used commercial Augmented Reality solution.
Alvar http://virtual.vtt.fi/virtual/proj2/multimedia/alvar/	Mobile SDK for iOS, Android, Symbian, Maemo, Flash and Silverlight platforms,	**Support for full 3D tracking**	
Layar	iPhone,	2d/3d animation	

www.layar.com	android,, BlackBerry	tools	
Aurasma http://www.aurasma.com		animation, video, audio, and 3D content.	
BlipAR	iOS, Android, Windows and wearables,		Uses technology by Layar
Wikitude: http:// www.wikitude.com	iOS, Android, Windows and wearables,	3d with SLAM	
In2AR	iOS , android,	AR by Flash and AIR SDK and content creation support to Unity 3d	

Table 8. Commercial solutions for Augmented reality

It is worth of noting that within few years many commercial AR players have bought by big companies, which means that expectation for the technology is high. Below are the famous examples:

- Vuforia bought by PTC (a global provider of technology platforms and solutions that transform how companies create, operate, and service the "things" in the Internet of Things (IoT).) 2015
- Metaio bought by apple (2015)
- 13th Lab bought by Facebook (Oculus Rift Division)

Open-Source AR technology	Platforms	Special Features	Licence	
Argon http://argon.gatech.edu/	Argon is currently designed for the iOS (3.1); an Android version is in progress	javascript framework for adding augmented reality content to web applications	Apache 2.0 Open Source License.	Uses Vuforia Tracking
Alvar http://virtual.vtt.fi/virtual/proj2/multimedia/alvar/	Linux, windows		GNU Lesser General Public License, version 2.1	
ArToolkit http://www.hitl.washington.edu/artoolkit/	Windows, Linux, MacOs also Android, Flash or Silverlight		dual-license: GPL, commercial	First open-source AR

JSARtoolkit https://github.com/kig/JSARToolKit	HTML5 + webRTC	JavaScript port of FLARToolKit	GNU General Public License	Inherits from ARtoolkit (FLARToolKit is the Flash Actionscript (v3) version of ARToolKit)/ demo with webRTC support exists
ArUco http://www.uco.es/investiga/grupos/ava/node/26	Windows, Linux	Marker based	BSD	Based on OpenCV, Trivial integration with OpenGL and OGRE.
JavaCV https://github.com/bytedeco/javacv	Java / Android interface		GPLv2	Based on ArToolkit, Open CV
ATOMICAuthoringTool http://sourceforge.net/projects/atomic-project/	Microsoft Windows, Ubuntu and Mac OS X.		GNU GPL	
GoblinXNA https://goblinxna.codeplex.com/	Windows XP, Vista, or 7 and Windows Phone 7.5		Goblin XNA License.rtf Microsoft Permissive License.rtf (Latest release 27.6.2012)	Based on ALVAR marker-based camera tracking package, InterSense hybrid trackers. supports the Vuzix iWear VR920
GRATF http://www.aforgenet.com/projects/gratf/		Marker based, 2D/3D augmentation	GNU GPL v3 (Latest release 06.03.2012)	
Mixare http://www.mixare.org/	Android/iPhone	Marker based, gps 2D augmentation	GPLv3 (latest commit August 2012)	
PTAM, Parallel Tracking and Mapping for Small AR Workspaces https://github.com/Oxford	Linux, Win32, OSX	SLAM	GPL (latest commit 5.11.201	Only for non commercial use

-PTAM/PTAM-GPL			3)	
DroidAR https://github.com/bitstars/droidar	Android	Location, marker based, 3d objects	GPL	
GeoAR https://wiki.52north.org/bin/view/Projects/GeoAR	Android	Geospatial data , 2d support	Apache 2.0 License (latest version March 2013)	
BeyondAR http://beyondar.com/home	Android and google glass		Apache 2.0 License	

Table 9. Opensource AR software

Augmented Reality technology is used in many ongoing research projects. The closest to NUBOMEDIA approach are EU projects FIWARE and Compeit, where media server technologies and AR are developed and utilized.

- FIWARE, where an open source platform for and supporting sustainable ecosystem around it is developed. AR is part of the platform, Java Script support is provided by JSAR open source library as well as VTT ALVAR commercial library.
- Compeit, concentrates creating highly interactive, personalised, shared media experiences

Other on going or recent projects example for AR are

- iAM project http://www.iam-project.eu/, heritage and tourist domain
- Satifactory, http://www.satisfactory-project.eu/, industry 4.0
- Lara http://lara-project.eu/index.php/project-overview, navigation and position technology
- Target Training Augmented Reality Generalised Environment Toolkit, http://cordis.europa.eu/project/rcn/194852_en.html, gaming
- Insiter http://www.insiter-project.eu/, construction, refurbishment and maintenance of energy-efficient buildings
- Venturi https://venturi.fbk.eu/, pervasive AR, user centric design

4.3.2 NUBOMEDIA approach beyond SotA

Currently most of the AR applications and platforms offer their solutions so that the processing of all required functions are performed locally. This due to the fact that the one of the fundamental ideas in AR solutions has been consumer applications where the user itself is looking the world via mobile device and gets new view augmented by virtual elements. Transferring the augmented view for others has not been the main emphasis in the field with the exception of adaptive advertising in sport broadcast events. The first AR solutions were based on markers, but now more and more technology is available for recognizing e.g. planars, buildings and objects from real 3D

world via camera and other sensors. This requires much more computational power and thanks to new era of cloud computing the trend is to split tasks on the client and server side. For example Gammeter & al [2] published a system for mobile AR where object recognition is performed on the server side and tracking on the client side. In planar (image based) type city AR where recognizable object can be merely anything, this is the only possible approach to search a match from millions of objects. The tracking is implemented utilizing both image and sensor based methods. Distributed architectures are also used by commercial solutions, e.g. Wikitude provides cloud based recognition of images.

The consideration of using AR in videoconferencing or telepresence applications opens totally different view on designing the architecture for such applications. In video communication cases the video stream is processed by the media server and this allows extending more complicated processing to be done at the server side, such as rendering which currently almost always done at the client side. In [1] there is a proposed an architecture for enhancing a video service. In this architecture the AR overlay is performed on the server side and higher rate output of video as well as more accurate positioning is expected as there is more computational power available.

Rendering at the server side is not common in AR applications. But for example Playstation already has a cloud gaming service, where all the processing for interactive virtual content is done at the cloud server. The game is then streamed as a video to the viewing device. One benefit of that is to allow very wide selection of mobile devices with incompatible properties, e.g. hardware for 3D acceleration to enable PS gaming. In [1] they also made a comparison of device-side and server-side AR videophone, table[3] below from [1].

	Device-side	Server-side
Performance		✓
Efficiency	✓	
Reusability		✓

We can also identify other benefits of the server-side AR

- AR content management eases out, as the required material for rendering does not need to be locally stored, also no need to download all the content to be rendered
- Larger virtual models (e.g. 3d models) can be handled at the server side
- Content can be changed on the fly
- In advertisement type of scenario, the content management is at the server side and easily controlled
- Video streams are always supported and incompatibles devices for rendering have access to augmented video stream
- Customization of the rendered image -> in multipoint communication the rendered content can vary between each participant

One of the well-known bottlenecks in AR has been introduction of the AR solutions for the end user. Web-RTC protocol is changing the playground for AR and communication systems by bringing standardized means of real time communication to the web applications. For the user WebRTC supported service means, that there is no (necessary) need for a special application download for the AR enabled service. Only access to internet and browser is enough. AR enhanced videoconferencing system

would need only opening a web page, not always downloading and installing specific application for this, but only opening a link at the browser. This will tackle down one obstacle for AR to become mainstream as supported device population is extremely large. For the creators and developers this means web based development of applications.

OoVoo (http://www.ooVoo.com) is a company providing WebRTC enabled communication services and SDK for developing them. They already have elementary video filters for enhancing the video communication with video effects or e.g. by adding a 2d images over/top of the face based on face position. In their solution the rendering is performed on the client side.

4.3.3 NUBOMEDIA outcomes

Augmented reality functionality in NUBOMEDIA is providing the media server AR capabilities based on marker and planar recognition and tracking, as well as supporting the rendering of 2D / 3D content. This will provide a unique platform for using webRTC enabled implementation and supporting distributed AR application architecture with server side rendering, detection and tracking.

In NUBOMEDIA the rendering will take place on the server side. This opens a potential to new type of applications and business models. The rendered content can be very complex as massive 3D scene descriptions are not transferred between the server and client. On a distributed server architecture, load balancing allows different content rendering for each participant in multipoint video communication. This for example introduces a new way of advertising where the ads can be embedded in video stream and personalized by each person's browser history.

Innovation of AR in NUBOMEDIA is on the usage of the AR technology in new way in multimedia communication, enabling easy to develop AR applications to the very large audiences without the need of installing specific AR applications.

4.3.4 References

[1] Fukayama, A.; Takamiya, S.; Nakagawa, J.; Arakawa, N.; Kanamaru, N.; Uchida, N., "Architecture and prototype of augmented Reality videophone service," in *Intelligence in Next Generation Networks (ICIN), 2011 15th International Conference on* , vol., no., pp.80-85, 4-7 Oct. 2011 doi: 10.1109/ICIN.2011.6081108

[2] Gammeter, S.; Gassmann, A.; Bossard, L.; Quack, T.; Van Gool, L., "Server-side object recognition and client-side object tracking for mobile augmented reality," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on* , vol., no., pp.1-8, 13-18 June 2010

4.4 Interoperability on real-time media infrastructures servers

Interoperability has always been the corner-stone of the Telecom networking ideas, where each carrier or service provider complies with well-structured agreed standards. The web world achieves compatibility by de-facto prevailing mechanisms that can

morph and evolve much faster, but particular services are dominated by few large players, who require all users to join, with no inter-service interworking.

Over the last decade over the top services (OTT), such as instant messaging (IM) and Voice over IP (VoIP), have revolutionized the well-established services and business models of real-time media providers. However, applied communication standards and protocols are mainly used to create closed ecosystems, resulting in a highly fragmented market of mostly isolated communication platforms that restrict free, open and interoperable communication flows. Currently, telecommunication architectures are based either on the Telco federation or on a global "Private Walled garden" Market models:

- Telco Federated distribution model: universal interoperability and a highly regulated market constrain service delivery agility to geographically limited markets. However, this model ensures consumers well defined expectations in terms of reachability among users (independently of its service provider domain), trust in service providers and service quality.
- "Private Walled Garden" distribution model: used by popular players like Google, Skype and WhatsApp (aka Over The Top - OTT) that have much more agility to deliver cost-effective (mostly free) innovative services to borderless markets. However, OTT players are creating silos of users where only intra-domain interoperability is ensured (for example, a WhatsApp user can only send a message to another WhatsApp user, a Skype user can only call to another Skype user or use Skype's breakout service to the PSTN). In this model, service developers and third-party service providers are forced to use proprietary APIs and incur difficulties if they attempt to sell and distribute services to different domains and are limited to the user base of the service provider. In general, services are developed within a specific service domain, constraining service reachability between different service domains. Trust in service providers is more and more questionable and becomes a concern; the service delivery is unregulated and has to rely on best-effort Internet connectivity in all cases

4.4.1.1 Description of current SoTA

4.4.1.2 IMS and VoLTE

The IP Multimedia Subsystem (IMS) is currently the state of the art in terms of standardized Service Architecture for Telecommunication Services. It is an evolution of the Intelligent Network (IN), but based on Internet Protocol (IP) and Information Technology (IT) principles. IMS has been specified by 3GPP in Rel. 5 onwards, as summarized in TS23.228 [3GPP] as the core packet network session control. It supports a layer of communication services that replaces the TDM based IN.

The goal for IMS was to provide Telecom operators a secure, trustworthy session control environment that operators and 3rd party service providers can plug their services into, e.g., Joyn/RCS [GSMA]. However, due to IMS complexity and the traditional restrictive architecture, service development is slow and cumbersome. At the same time, the technological and business-related innovations in the Internet space have generated a wealth of attractive services, with greater agility and low cost base. These services run over the enhanced 4G infrastructure that is now providing high bandwidth (LTE), and this is exploited by OTT (Over The Top) players, locking out the carriers.

Telecommunication service providers are struggling to match the agility and innovation of web based services, even when they replace conventional Telecommunication

services, while the changes in communication habits and user behaviour, e.g. texting and social networking, encourage the shift of users towards web based facilities.
The introduction of VoLTE (IMS over LTE) requires a complete core network replacement based on the 3GPP Evolved Packet Core (EPC) for end-to-end QoS provisioning in LTE networks, with IMS as the session control and service environment for Voice/Video conversational services. This is perceived as a huge effort (manpower, project management, CAPEX, OPEX), while the prospects of new revenues in the increasingly competitive environment of real-time communication services are diminishing.

Figure 37 shows the IMS layered architecture [40], where the layers of session control and service logic have been separated, unlike previous architectures of circuit-based switches and even softswitches. This enables the calling applications to operate independently, while activating the IMS core for session control. Therefore, the current rolling out of VoLTE (IMS over LTE) provides an excellent opportunity for web applications to 'grab' the front-end, while still perform like carriers' IMS systems.
As shown, the media plane is also an independent layer, but it is subject to the session control that determines the media parameters (QoS, priority, security). The media runs directly between the communicating points, while session control signalling traverses the network to the core servers.
Also shown is the range of management functions. Besides CRM, Billing and Provisioning as well as network management, it shows that new facilities for hosting network services are now enabled, due to the modularity of IMS. The implementation of the IMS platform for a large service provider benefits from the ideas of SOA, as described in [COPELAND_SOA_2009]. The SOA principles can help to provide efficient distributed IMS core, built of modular components that can be scaled independently from each other. It follows that what makes IMS so complex is also the reason for its flexibility and openness.

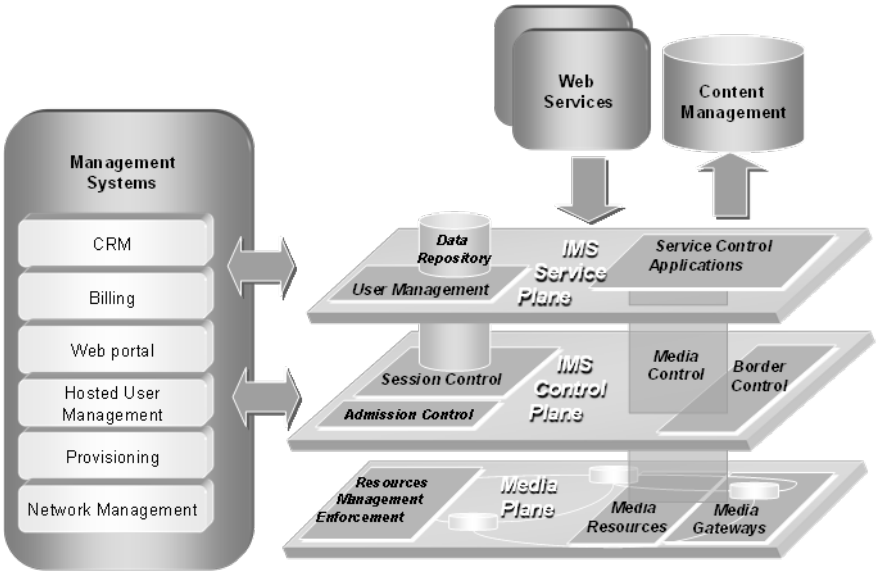


Figure 37 IMS Layered Architecture

The monolithic approach to communication, where devices, access, backhaul, core and applications were all under a single communication service provider is no longer valid. With the advent of LTE and the EPC (Evolving Packet Core), the access network has become an independent layer, hence the IMS core can serve multiple types of access networks. The application layer is also independent, but is still very constraint by the

IMS interfaces. As we are now facing new definition of communications within the Internet space using web technologies, it is important to inherit the achievements of IMS while attempting to surpass it in other respects.

In Figure 38 and Figure 39, [WEBRTCHACKS] attempts to integrate the webRTC client with the P-CSCF, the IMS user agent function, and converge data from the webRTC portal with its authentication system with HSS user data repository.

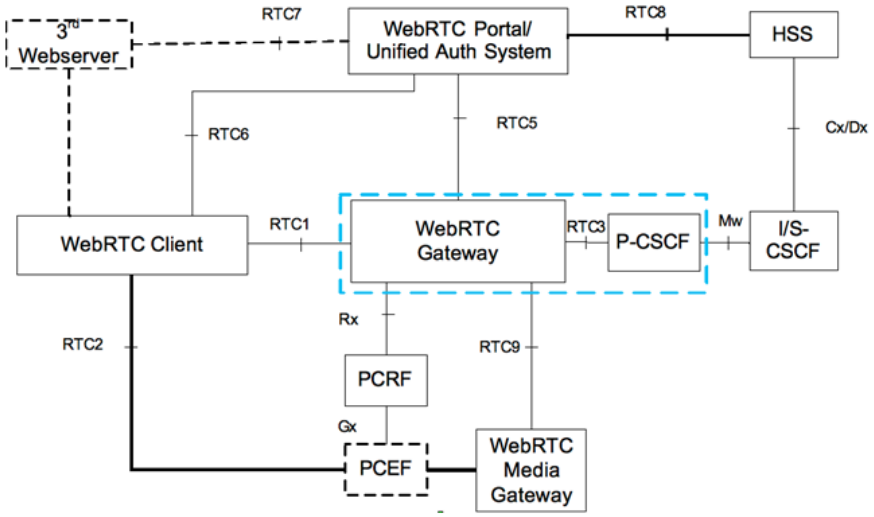


Figure 38 WebRTC integration with IMS user agent and data repository

This approach allows IMS functions to be used in session control for webRTC sessions. While this may not be entirely desirable for those who seek new ways of controlling sessions, the requirement to interconnect with IMS is unavoidable, given the growing numbers of IMS systems being installed.

In Figure 39, the NNI (Network to Network Interface) connects elements of the webRTC architecture to IMS gateways. Note that the IBCF (Interconnection Border Control Function) converts webRTC signalling on the fly, just as it does for H.248 to SIP, and that the Translation Gateway (TrGW) converts media codecs on the fly, for webRTC as well as IMS media.

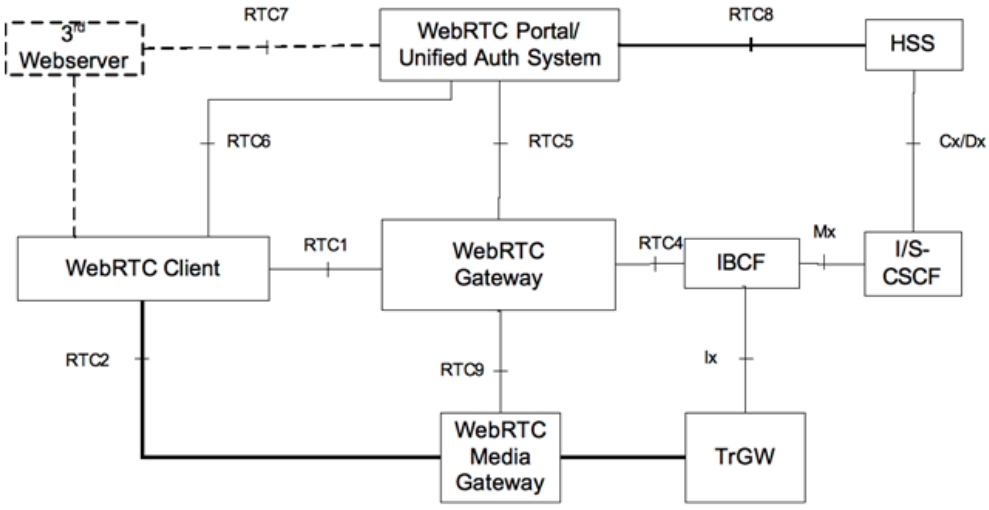


Figure 39 webRTC linking to IMS via NNI

While the IMS core can continue to provide QoS-managed service delivery even when the front-end application is web based, carriers who implement IMS are still constraint by territorial licensing. In addition, the Telecom Industry has failed to provide attractive communication services that match the agility and innovation of the web players. The IMS support of applications is not easy for web developers who find the service interaction via SIP too difficult. SIP, which has been born as a simple way of invoking Voice over Internet, has been extended greatly, and is now perceived as difficult to work with.

The service environment for IMS that integrates HTTP based services, as specified by OMA (Open Mobile Architecture), is given in Figure 11 [COPELAND2009]. It shows the main interface to services – the ISC (IMS Service Control) which is SIP based, connecting to service enablers that deal with compatibility and interworking, before linking to application. Although this architecture is comprehensive, it is clear that it does not encourage rapid service development.

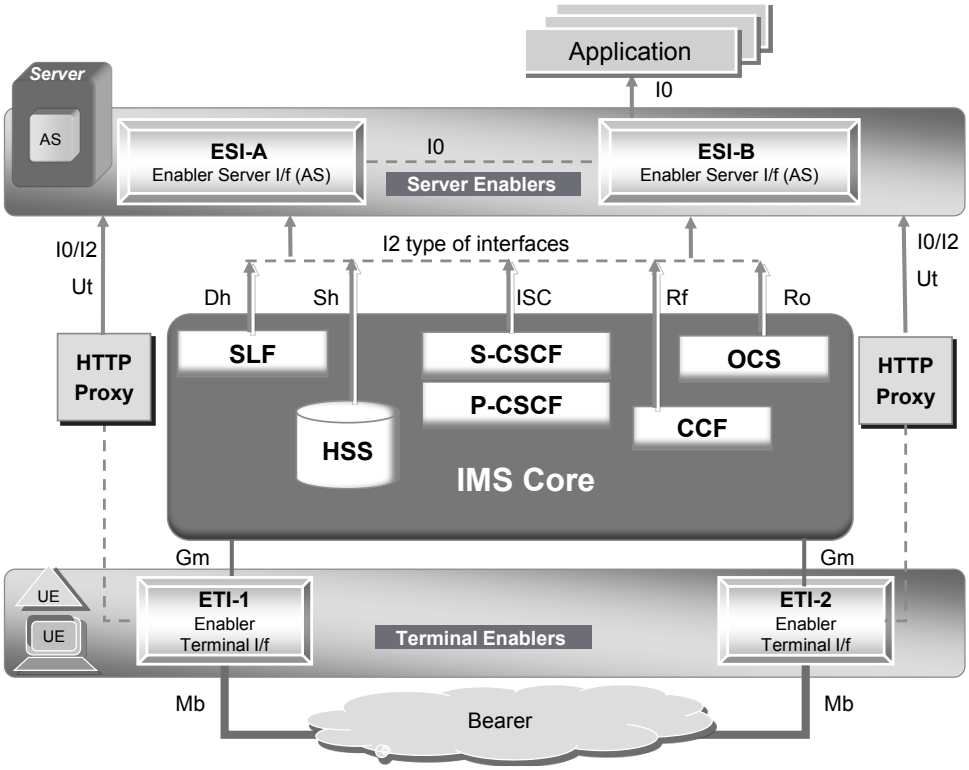


Figure 40 OMA based IMS service architecture

4.4.1.3 WebRTC Media

Taking a web-centric service perspective on IMS, the complete standard and environment may be considered somehow anachronistic. The value of IMS lies clearly on trust, security, accountability and the integration of managed networks but does not address requirements from an open service environment, such as programmability, extensibility and ease-of-use.

With HTML5, and WebRTC features, the Communication Industry have an opportunity to remodel the full Core Communication Services Infrastructure, unifying communication Web and IoT towards a more agile, simple and open environment.

Applying RESTful APIs and light policy driven SOA Governance principles, would lower the barrier, facilitate cloud deployment, and of course the composition with Web applications.

However, WebRTC does not specify how communications are established (e.g. from signalling perspectives). Different initiatives behind the IETF have been adopted to standardize some signalling protocols for WebRTC, but there is no agreement across vendors. SIP over WebSocket, JSON, proprietary APIs and SDKs, standard APIs are options already available by different vendors.

WebRTC services will also require:

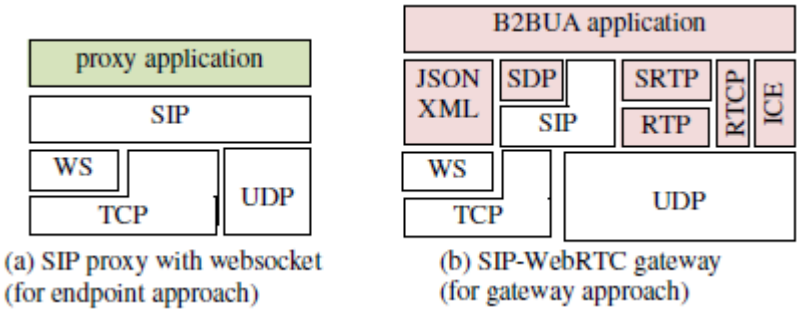
- Opening network functionalities to developers and users. This will be done by standard-based API and SDKs that will help to avoid vendor lock-in and bring a huge quantity of web developers to the world of real time communications. This way, easy-to-use APIs will be available soon.
- Managing quality of service and efficiency with measurement tools. There will be valuable tools to define, implement and test, and some cost-efficient mechanisms to improve the quality of communications.
- Building services that deal with all the security concerns. Different services offered as OTT have suffered security threats and attacks, due to the immaturity of their business model, company background, and issues related to lack of ownership to the layers below applications (multi-layer policy servers or firewalls cannot be used here).
- WebRTC does not mandate the context in which real time communications are used.

4.4.1.4 Signalling protocols and Gateways

Interoperability of WebRTC communications and SIP based systems is an inevitable requirement, given the roll out of VoLTE/IMS in both wireline and wireless networks. In [SINGH2013], a method of 'SIP in JavaScript' is proposed, with two options:

- In an Endpoint Browser approach, the SIP stack runs in JavaScript in browser, using a SIP proxy server to support Web Socket;
- In a network gateway at web server, it is proposed to have a special gateway to enable interworking. This gateway can be hosted by web provider, VOIP provider or an independent third party.

[SINGH2013] argues that signalling over Web Sockets (SIP in JavaScript) and media over WebRTC allows keeping tools separate from the applications, which enhances scalability and flexibility. As the last resort, a network gateway capable of translation is used for transcoding signalling and media.
Figure 41 shows the proposed SIP-webRTC gateway, where highlighted areas indicate the differences.



(a) SIP proxy with websocket (for endpoint approach)

(b) SIP-WebRTC gateway (for gateway approach)

Figure 41 Comparing SIP proxy with SIP-webRTC Gateway

Note, that while SIP is a proxy, the proposed SIP-WebRTC gateway is a B2BUA (Back to Back User Agent), which implies special resident intelligence and decision making, rather than merely represent the user actions to the network.

4.4.2 NUBOMEDIA approach beyond SotA

To fully achieve interoperability on real-time media infrastructures is a big challenge because, these infrastructure apply communication standards and protocols, the effect being creation of mainly closed ecosystems, resulting in a highly fragmented market of mostly isolated communication platforms that restrict free, open and interoperable communication flows.

4.4.2.1 Interoperability Requirements

An alternative method that improves on the integration of web communication applications is required. The NUBOMEDIA project must surpass the IMS alternative in several ways:

- Front-end web communications apps must have easy access to session control, without the constraints of the IMS service architecture interfaces (ISC etc.).
- Web technologies and protocols are assumed to be used by web developers, for example – using REST and resource based architecture instead of extended Diameter for access to service data.
- Different web communication application may be used by the parties in a single session, each supporting different sets of session features (Caller ID, Do-not-Disturb, no-return-calls, bar last caller).
- New applications and modified applications must be able to address users' devices without having to undergo end-to-end testing, i.e. new features may not work in some cases, but will not disrupt the whole network, while a safe mechanism is provided for updating apps clients.
- APIs to network facilities, such as QoS policy, Charging/billing and enhanced authentication, must be standardized, so that web apps do not need to change their service logic to link to different CSPs.
- A network gateway capable of translation is required for transcoding signalling and media

To fulfill all the above requirements within the NUBOMEDIA project is impossible because it requires standard support of creation and integration of gateways binding both the telco and web domains which is out of scope of the project.

Nevertheless, a proof-of-concept implementation demonstrating the interoperability on the signalling plane of a WebRTC based infrastructure and an IMS infrastructure. This approach allows IMS functions to be used in session control for WebRTC sessions. The media path integration on the other hand is still not fully interoperable.

4.4.3 NUBOMEDIA outcomes

Fraunhofer FOKUS has implemented an IMS Connector which provides the possibility to be used as an application server on the IMS network infrastructure or an IMS proxy masking a WebRTC client as a User Agent on the IMS network. The IMS Connector itself provides a grad variation in operation and can operate in both modes at the same time. The software artifact are found on the public fhg-fokus-nubomedia github repository [FOKUS_GITHUB].

4.4.4 References

Referencing websites

[3GPP] 3GPP TS 23.228, IP Multimedia Subsystem (IMS); Stage 2: [Online]. Available: http://www.3gpp.org/DynaReport/23228.htm

[GSMA] GSMA Rich Communications Suite, Specs & Product Docs, [Online]. http://www.gsma.com/network2020/rcs/specs-and-product-docs/

[WEBRTCHACKS] V. Pascual in https://webrtchacks.com/a-hitchhikers-guide-to-webrtc-standardization/

Referencing papers or books

[COPELAND_SOA_2009] R. Copeland. SOA Case Study – the BT IMS OSIP. BT Technology Journal 2009

[COPELAND2009] R. Copeland. Converging NGN Wireline and Mobile 3G Networks with IMS. ISBN 978-0-8493-9250-4, Taylor & Francis 2009

[SINGH2013] Singh, K.; Krishnaswamy, V., "A case for SIP in Javascript," Communications Magazine, IEEE , vol.51, no.4, pp.28,33, April 2013

4.5 Cloud APIs for accessing Media Servers

Analysts such as Marc Andreessen claim that “software is eating the world” stressing the importance of software-centered models into the economy and the transition of traditional business to software-based organizations [ANDREESSEN2011]. This trend is permeating into all areas of IT (Information Technologies) including also multimedia industries. In the last few years, we have witnessed how multimedia technologies have been evolving toward software-centered paradigms embracing cloud concepts through different types of XaaS (Everything as a Service) models [CATHERINE2013].

More recently, another turn of the screw is taking place thanks to the emergence and popularization of APIs (Application Programming Interfaces). This is perfectly summarized by Steven Willmott with his claim “software is eating the world and APIs are eating software” [WILLMOTT2013]. Software developers worldwide are getting used to create their applications as a composition of capabilities exposed through different APIs. These APIs are typically accessible through SDKs (Software Development Kits) and expose in an abstract way all kind of capabilities including device hardware, owned resources and remote third party infrastructures. This model, applied to cloud concepts, is quite convenient for individual developers and small companies, which have now the opportunity of competing with large market

stakeholders without requiring huge effort investments and without needing to acquire hardware infrastructure or software licenses. Thanks to this, in the last few years, we are experiencing an explosion of innovation with thousands of new applications and services both for WWW and smartphone platforms that are being catalyzed by the rich and wide ecosystems of APIs made available to developers.

This trend towards the "APIfication" is also invading the multimedia arena and, very particularly, the RTC (Real-Time multimedia Communications) area. Initiatives such as WebRTC [JOHNSTON2012] are bringing audiovisual RTC in a standard and universal way to WWW users. The main difference between WebRTC and other popular video-conferencing applications is that WebRTC is not a service, but a set of APIs enabling WWW developers to create their customized applications using standard WWW development techniques.

WebRTC belongs to the HTML5 ecosystem and has awakened significant interest among the most important Internet and telecommunication companies. As opposed to other previous proprietary WWW multimedia technologies, it has been conceived to be open in a broad sense, both by being based on open standards and by providing open source software implementations. Currently, a huge standardization effort on WebRTC protocols is taking place at different IETF working groups (WGs), being the RTCWeb WG the most remarkable one [RTCWEB2015]. In turn, WebRTC APIs are being defined and consolidated at the W3C WebRTC WG [WEBRTC2015]. WebRTC standards are still under maturation stage and they might take some time to consolidate. In spite of this, most of the major browsers in the market already support WebRTC and it is currently available in billions of devices providing interoperable multimedia communications.

Hence, WebRTC is an opportunity for the creation of a next generation of disruptive and innovative multimedia services catalyzed worldwide through those emerging APIs. However, to reach this goal, the WebRTC ecosystem needs to evolve further. Basing on WebRTC browser capabilities, services can only provide peer-to-peer communications, which restrict use-cases to simple person-to-person calls involving few users. In order to enhance this model, server side infrastructures need to be involved. This is not new: as it is well known, the traditional WWW architecture is based on a three tier model [FRATERNALI1999] involving an application server layer and a service layer, this latter typically reserved to databases. In the same way, rich media applications also base on an equivalent three tier model where the service layer provides advanced media capabilities. The media component in charge of providing such capabilities is typically called media server in the jargon.

Media servers are a critical ingredient for transforming WebRTC into the next wave of multimedia communications and the availability of mature solutions exposing simple to use yet powerful APIs is a necessary requirement in that area. However, most standardization and implementation efforts are still concentrated at the client side and server side technologies are still quite fragmented. Although there are a relevant number of WebRTC media servers available they do not provide coherent APIs compatible with WWW development models and developing with them typically requires expertise with low level protocols such as SIP [ROSENBERG2002], XMPP [SAINTANDRE2011] or MGCP [ANDREASEN2003], on which average WWW developers do not have any experience. In addition to this, most state-of-the-art media WebRTC media servers just provide the three basic capabilities specified above and are extremely hard to extend with further features. However, nowadays, many RTC services involve person-to-

machine and machine-to-machine communication models and require richer multimedia processing capabilities such as computer vision, augmented reality, speech analysis and synthesis, etc.

4.5.1 Related work

The commoditization of RTC media server technologies brought increasing interest on more flexible mechanisms for media control. Several IETF WG emerged with the objective of democratizing them among common developers. As a result, protocols such as MSCML [VANDYKE2006], MSML [SALEEM2010] emerged providing the ability of controlling media server resources through technologies understandable and familiar to average developers such as XML and [BRAY1998].

Although these protocols are simpler to understand and integrate, developing application on top of them is still a cumbersome, complex and error prone process. Due to this, many stakeholders noticed that the natural tools used by developers are not protocols but APIs and SDKs. Hence, a number of initiatives emerged trying to transform the protocol-based development methodology into an API-based development experience providing seamless media server control through interfaces adapted to programming languages specificities and not to infrastructure characteristics. In particular, the Java platform was one of the first on integrating this philosophy by trying to reproduce the WWW development experience and methodology for the creation of RTC media enabled applications. A relevant activity in this area is JAIN (Java API for Integrated Networks), which issued several APIs for the signaling, control and orchestration of media capabilities. These include the JAIN SIP API [ODOHERTY2003] the JAIN SLEE [FERRY2004] API and the JAIN MEGACO API [BAJAJ2004]; this later being specifically devoted to control media servers through the H.248 protocol. JAIN APIs did not permeated much out of operators, but their ideas inspired more popular developments such as the SIP Servlet APIs [KRISTENSEN2003], for the signaling plane, and the Media Server Control API (aka JSR 309) [ERICSON2009] for the media plane, which have been more widely used for the development of RTC solutions for voice and video.

Among all these APIs, this document is especially interested in the JSR 309. JSR 309 concepts were quite revolutionary at the moment because the API tried to fully abstract the low level media server control protocols and media format details. The objective was to enable developers to concentrate on application logic. For it, JSR 309 defined both a programming model and an object model for media server control through a northbound interface, but independent of media server control protocols and hence, without requiring any specific southbound protocol driver. JSR 309 does not make any kind of assumption in relation to the signaling protocol or to the call flow, which are left to the application logic.

From a developer's perspective, probably the most innovative concept of JSR 309 was the introduction of a mechanism for defining the media processing logic in terms of a topology. This mechanism is based on an interface called `Joinable`. In JSR 309, all objects having the ability to manipulate media (e.g. send, receive, process, archive, etc.) implement such interface, which has a `join` method enabling interconnecting such objects following arbitrary dynamic topologies. Hence, a specific media processing logic can be implemented by developers just joining the appropriate objects. As an example, if you want to create an application mixing two RTP (Real-time Transport Protocol) streams and recording the resulting composite into a file, you just need to join the appropriate objects with the appropriate topology. Taking into consideration that in

JSR 309 the `NetworkConnection` is the class of objects capable of receiving RTP streams, that `MediaMixer` is the class of objects with mixing capability and that `MediaGroup` is the class with be ability of recording; the above mentioned media topology can be achieved just by joining two `NetworkConnection` instances to a `MediaMixer` instance, which in turn, is joined with a recording `MediaGroup`. This approach makes possible for developers to conceive their media processing logic as graphs of "black-box" *joinables*, which is a quite modular and intuitive mechanism for working in abstract terms with the complex concepts involved in RTC multimedia applications.

Another relevant innovation of JSR 309 is the introduction of media events. Thanks to this mechanism, the media processing logic held by a media server can fire events to applications through a publish/subscribe mechanism. This is very convenient for enabling applications to become media-aware meaning that complex processing algorithms at the media server can provide asynchronous information dealing with things happening inside the media, for instance DTMF (Dual-Tone Multi-Frequency) tones being detected, voice activity being present, and so on.

JSR 309 permeated into mainstream developer audiences as a suitable API for media server control following the typical three tier model. However, in the last few years, the emergence of novel technologies and computation paradigms have made JSR 309 to show relevant limitations. For example, nowadays group videoconferencing services are evolving from Media Mixing models, which require relevant media processing, towards SFU (Selective Forwarding Unit) models, which are based on media routing [WESTERLUND2016]. JSR 309 is heavily adapted to Media Mixing and, due to this, most of its APIs assume that participants send/receive only one media stream to/from the media server. As a consequence, SFU models do not fit nicely into JSR 309 APIs. This is particularly a problem when all the streams of a group videoconference are multiplexed into a single RTP session, as happens typically on modern WebRTC SFU media servers supporting bundle RTP [JENNINGS2015] because JSR 309 APIs do not provide any kind of mechanism for demultiplexing streams from a `NetworkConnection`. Moreover, in JSR 309 the API specification explicitly forbids several input `NetworkConnections` to be joined to a single output `NetworkConnection`, as an SFU router would require. Instead, they need to be joined first to a `MediaMixer`, which, in turn, can be joined to the output `NetworkConnection`.

When looking to other modern RTC technologies, we notice again that the JSR 309 design has limitations. For example, if we consider WebRTC W3C APIs, we may understand that they split endpoint capabilities into different functional blocks each of which is exposed through an abstract interface (e.g. `RtpSender`, `RtpReceiver`, `PeerConnection`, etc.) However, if we want to expose WebRTC media server capabilities through JSR 309 we need to accept that endpoints can only be represented through the `NetworkConnection` interface, which is extremely limited to support rich WebRTC capabilities such as `DataChannels` [BECKE2013], Trickle ICE [IVOV2014], simulcast [WESTERLUND2015], etc.

JSR 309 shows also drawbacks in relation to its extensibility. In JSR 309 it is possible to support new media object types using `MediaGroups`, however, configuration of this new types have to be done with Media Server specific descriptions as strings, which cannot be validated by compiler. It is important to note that these new media object

types cannot be `NetworkConnection`, only `MediaGroups`. This is a hard limitation because no other network protocol different than RTP (negotiated through SDP) can be incorporated. The ideal would be to allow supporting the creation of new object types in a similar way than core types, with factory methods in `MediaSession` (e.g. `createNetworkConnection`, `createMediaGroup`, etc.), but this is not possible as `MediaSession` is an interface defined in JSR 309 API and hence it cannot be modified by the API user.

Further limitations about JSR 309 are the following:

- A counter-intuitive asynchronous development model basing on an obscure `joinInitiate` primitive, which is incompatible with modern Java mechanism for managing asynchrony such as futures, continuations or lambdas. This lack of clean asynchronous programming model makes JSR 309 difficult to adapt to reactive programming frameworks and languages that are very demanded by developers today such as Node.js or Scala.
- A complete lack of mechanism for monitoring and gathering quality stats on media sessions. This is an essential ingredient for production systems.
- JSR 309 it is designed specifically for the Java language. It would be desirable a portable API that can be used in as more languages as possible.
- This API is specially designed to control phone Media Servers because it expose concepts like Dialogs (Prompt and record, DTMF, VoiceXML dialog, etc.). For example, it is mandatory for an implementation to provide a player with the capability to detect audio signals in DTMF, but this kind of functionality is not very useful in web applications.

4.5.2 NUBOMEDIA approach beyond SotA

NUBOMEDIA approach for progressing beyond SotA is to create a novel API complying with a number of requirements that, as described above, are not available in current solutions. These progresses can be summarized as follows:

Seamless API extensibility through custom modules

We want developers to be able to plug additional capabilities to the API (e.g. processing algorithms, protocols, etc.) and to enable their consumption as if they were native API capabilities (i.e. without requiring different syntax or language constructs.) The mechanism we require for this is based on modules in the sense that every extension takes the form of a module artifact (e.g. a `.jar` file in the Java language, a `.js` file in JavaScript language, etc.) and that developers may plug the modules they wish at development time without requiring any further modification or configuration. Remark that, for the reasons specified in sections above JSR 309 does not comply with this requirement.

Adaptation to WWW technologies and methodologies.

This requirement has two aspects. The first, and most important, is the need of our API to be adapted to novel RTC WWW technologies and very particularly to WebRTC. The WebRTC architecture, based on heavy use of RTP bundle and RTCP multiplexing mechanisms and requiring complex ICE management techniques such as Trickle ICE makes complex to comply with this requirement. Also as specified in sections above, JSR 309 is not compatible with this as the `NetworkConnection` is based on plain RTP. The second, is the need of the API to adapt to the typical WWW three tier development model. This means that the NUBOMEDIA Media API should be usable for WWW developers with their common development, deployment and debugging techniques and tools. To some extent, this means that the NUBOMEDIA Media API

should be perceived by WWW developers as any other of the APIs consumed in the application logic, such as database APIs or ESB (Enterprise Service Bus) APIs.

Full abstraction of media details (i.e. codecs and protocols)
Media representation and transport technologies are complex and require specialized knowledge that is not typically available for common developers. For maximizing productivity and minimizing development and debugging complexity the NUBOMEDIA Media API should hide all the low level details of such technologies through the appropriate abstractions. In doing so, these abstractions must maintain the appropriate expressiveness enabling the API semantics to provide to developers the ability of performing the required operations onto protocols and formats including payloading, depayloading, decoding, encoding, re-scaling, etc.

Programing language agnostic
In today's Internet, developers use a multiplicity of programming languages for creating their applications. In fact, the majority of applications are called "polyglot" because use different languages. The specific choice depends on factors such as the previous experience, the personal preferences, the tasks to be accomplished, the target platform or the required scalability. In this context, tying developers to a specific programming language may be perceived as inflexible and unfriendly. For this reason, the NUBOMEDIA Media API needs to be language agnostic and adapt to the most common programing languages used nowadays. Of course, the specific syntax of the API calls may differ depending on language specificities. However, this requirement indicates that, somehow, the constructs, basic mechanisms and programming experience needs to be the same across different languages. This means, for example, that a developer having the appropriate expertise for creating applications with the Java NUBOMEDIA Media API implementation should be able of doing so with a JavaScript implementation as long as the subtleties of the two languages are known.

RTC media topology agnostic
One of the main objectives of RTC Media Servers is to provide group communication capabilities to applications. Due to this, any useful NUBOMEDIA media API must consider this as a central aspect of its design by exposing the appropriate constructs for group communications. When looking to how RTC group communications are technical implemented, we can notice that they are based on a set of well-known RTP interconnecting topologies among which the most common ones are Media Mixing Mixers (MMM), Media Switching Mixers (MSM) and Selective Forwarding Units (SFU). In short, MMMs are based on the principle of composing a single output media stream out of N input media streams, so that the final composite stream represents the addition of the N input streams. MMMs require decoding of the N input streams, the generation of the composite (e.g. linear adding in audio or matrix layout for video) and encoding to generate the output stream. Due to the performance cost of these operations MMM do not scale nicely. On the other hand, MSMs and SFUs do not perform any heavyweight processing and they just forward and route N incoming streams to M outgoing streams, reason why they have better scalability properties. Their only difference is that MSMs enable the N to M mapping to change dynamically while on SFUs it is static and the only possible operation is switching on/off forwarding on any of the output M streams.

Understanding the differences and appropriate usage scenarios of these topologies is complex and a source of extra complexity for application developers. Due to this, we include a requirement for our NUBOMEDIA Media API to transparently manage all the

subtleties of this problem so that the most appropriate solution is provided transparently by the API. Remark that JSR 309 also tried to comply with this requirement through the "Joinable" mechanism making possible for developers to establish topologies just by joining sources with sinks. However, as explained above, both JSR 309, and equivalently JSR 79, are only compatible with MMM topologies and cannot manage the, by the way most popular, MSM or SFU models.

Advanced media QoS information gathering
QoS is critical in multimedia services. Some milliseconds of latency or jitter can be the difference between a successful and an unsuccessful application. For this reason, RTC media developers need to have the appropriate instrumentation mechanisms enabling seamless debugging, monitoring and optimization of applications. This requirements guarantees that our NUBOMEDIA Media API developers are able to access advanced QoS metrics of the streams including relevant information such as packet loss, bandwidth, latency or jitter. Remark that none of the above mentioned RTC media server APIs, including the JSR 309, provide this kind of capability.

Compatibility with advanced media processing capabilities
So far, most RTC media technologies and APIs have been concentrated on the problem of transport (i.e. taking media information on one place and moving it to other places.) This happened because the most prevalent use case for RTC is person-to-person communications, where end-users expect from technology to eliminate distance barriers (i.e. to maintain a conversation as if it were face-to-face.) However, during the last decade novel use cases involving person-to-machine and machine-to-machine communications are gaining popularity in different verticals such as video surveillance, smart cities, smart environments, etc. In all these verticals, going beyond plain transport is a relevant requirement. As an example, the number of low latency RTC video applications being used in security scenarios is skyrocketing. In all these applications the ability to integrate Video Content Analysis (VCA) capabilities through different types of computer vision algorithms is an unavoidable requirement. In addition, modern media applications in areas such as gaming or entertainment complement VCA with another trending technology: Augmented Reality (AR), which is also having high demand from users. As a result, we include our NUBOMEDIA Media API to provide full compatibility with these advanced processing techniques enabling their seamless integration and use.

Context awareness
In RTC media services, as in other types of services, context is becoming a relevant ingredient for providing added value to applications. Context is somehow an ambiguous concept for which there is not yet a formal definition. However, most authors accept context as any kind of information that can be used for characterizing the situation of an entity. The OMA (Open Mobile Alliance) has generated a formal definition of context through the NGSI standard [BAUER2010] as a set of attributes that can be associated to an entity. When working with RTC media, the entity is most typically a RTC media session (e.g. a media call).

Considering this context definition, this requirement means that our NUBOMEDIA media API needs to be capable of consuming context for customizing and adapting end-user experience but, most important, need to be capable of extracting context attributes from the media communication itself. In other words, the part of the context dealing with the media itself (i.e. what the media content is and what it represents at any time) needs to be manageable by the proposed API.

Adapted to multisensory multimedia
Traditionally RTC media has referred to simple audiovisual streams comprising typically one video track and one or two (i.e. stereo) audio tracks. However, moderns trends and technologies extend this to a new multisensory notion [PARK2009], where multisensory streams may comprise several audio and video tracks (e.g. Multi-view and 3D video) but may also enable the integration of additional sensor information beyond cameras and microphones (e.g. thermometers, accelerometers, etc.) Hence, we establish a requirement for our NUBOMEDIA Media API to be capable of managing such multisensory multimedia in as seamless and natural way.

Adaptation to cloud environments
Cloud computing is permeating in all IT domains, including multimedia, as the de-facto standard for system deployment and management. This trend is also permeating into the RTC media server arena, reason why we need to consider it in the definition of our API. Adapting the NUBOMEDIA Media API to cloud environments basically means to make it compatible with how a PaaS (Platform as a Service) media server works [VAQUERO2008] In other words, our API needs to be compatible with a new notion of distributed media server, which in opposition with traditional monolithic media servers, is distributed through a cloud environment and can elastically scale to adapt to end-users generated load.

4.5.3 NUBOMEDIA outcomes

The main NUBOMEDIA outcome in this project is the NUBOMEDIA Media API, which is described in NUBOMEDIA Project Deliverable D5.2. This API complies with the above mentioned SotA evolutions due to the following:

Seamless API extensibility through custom modules
The NUBOMEDIA Media API can be extended in a seamless way by using the RTC Media Module mechanism, which provides full flexibility and no restrictions other than extending from the base API classes.

Adaptation to WWW technologies and methodologies
The NUBOMEDIA Media API implementations fully comply with the traditional WWW three tiered development model and enable developers to create applications leveraging novel WWW RTC media technologies such as WebRTC in a seamless and direct way.

Full abstraction of media details (i.e. codecs and protocols)
The NUBOMEDIA Media API makes possible to perform transparent transcoding without requiring to worry about media internal details. This is due to the fact the semantics of the connect primitive mandates the underlying media server capabilities to perform all the appropriate adaptations in a fully transparent way.

Programming language agnostic
In our API, the only requirement for supporting a given programming language is to specify how the API IDL is transformed into it and to implement the appropriate compiler following that specification. In Deliverable 5.2 we provide such specifications and describe their implementations in Java and JavaScript in the context of the Kurento open source software project.

RTC media topology agnostic
The NUBOMEDIA Media API makes possible to interconnect media elements following arbitrary and dynamic topologies thanks to the `connect` primitive. This means that developers do not need to be aware of the low level details of MMM, MSM or SFU technologies: they just need to interconnect their endpoints, filters and hubs accordingly to their needs. The API semantics shall translate these interconnections into the appropriate low level mechanisms using MMMs, MSMs or SFUs in a fully transparent way.

Advanced media QoS information gathering
The API exposes primitives fully compliant with the standard WebRTC "inboutrtp" and "outboundrtp" stats.

Compatibility with advanced media processing capabilities
The concept of NUBOMEDIA API Filter provides this feature in a fully modular way advanced media processing.

Context awareness
The notion of context emerges in quite a seamless through the NUBOMEDIA Media API event mechanism, which makes it possible for media capabilities to publish events to applications. These events may contain semantic information about the media content itself. Hence, creating multimedia context-aware applications is straightforward: the application logic just needs to subscribe to the relevant events and publish them into a context database basing on NGSI or any other equivalent standard.

Adapted to multisensory multimedia
The NUBOMEDIA Media API can manage seamlessly arbitrary sensor data beyond audio and video. This can be achieved through the combination of two features. The first is the support for DataChannels that. The second is the fact that all streams exchanges among MediaElements may have a DATA track. In particular, any information received using DataChannels into a WebRtcEndpoint is published to the rest of the pipeline through the endpoint's source DATA track. In the same way, any information received through the DATA track at a WebRtcEndpoint's sink is send to the network using DataChannels. As the MediaElement interface enables all the information received through the DATA to be use by the element internal logic, this mechanism makes possible, for example, to create Augmented Reality filters that leverage sensor information for customizing the augmentation logic.

Adaptation to cloud media servers
The NUBOMEDIA Media API does not specify how media pipelines are placed into media server instances. The API implementer has full freedom for selecting how newly created media pipelines are scheduled. This flexibility can be leveraged by API implementers to adapt to all kinds of cloud architectures. This has enabled specific extensions that adapt transparently (i.e. without requiring modifying a single line of code) applications created for Kurento Media Server to work into the NUBOMEDIA PaaS.

These results have been useful for generating the following outcomes:

- Submission of two journal papers, one of them to the most important journal in multimedia tools and applications.
- Creation and consolidation of an open source software community around Kurento: a project belonging to the NUBOMEDIA ecosystem.

During the last year of the project, we plan to submit one additional scientific publication related with the NUBOMEDIA media API stack.

4.5.4 References

[ANDREESSEN2011] Andreessen M (2011) Why software is eating the world. Wall Street Journal 20

[ANDREASEN2003] Andreasen F, Arango M, Huitema C, Kumar R, Pickett S, Elliott I, Foster B, Dugan A (2003) Media gateway control protocol (MGCP) version 1.0. Tech. rep., Internet Engineering Task Force, Request For Comments (RFC) 3435

[BAJAJ2004] Bajaj V (2004) JAIN MEGACO API Specification. Tech. rep., Java Community Process, Java Specification Request (JSR) 79

[BAUER2010] Bauer M, Kovacs E, Schülke A, Ito N, Criminisi C, Goix LW, Valla M (2010) The context API in the oma next generation service interface. In: Intelligence in Next Generation Networks (ICIN), 2010 14th International Conference on, IEEE, pp 1-5

[BECKE2013] Becke M, Rathgeb EP, Werner S, Rungeler I, Tuxen M, Stewart R (2013) Data channel considerations for rtcweb. Communications Magazine, IEEE 51(4):34-41

[BLACKWELL2000] Blackwell AF, Green TR (2000) A cognitive dimensions questionnaire optimised for users. In: Proceedings of the Twelfth Annual Meeting of the Psychology of Programming Interest Group, pp 137-152

[BRAY1998] Bray T, Paoli J, Sperberg-McQueen CM, Maler E, Yergeau F (1998) Extensible markup language (XML). World Wide Web Consortium Recommendation REC-xml-19980210 http://www w3 org/TR/1998/REC-xml-19980210 16

[CATHERINE2013] Catherine MR, Edwin EB (2013) A survey on recent trends in cloud computing and its application for multimedia. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) 2(1):304-309

[ERICSON2009] Ericson T, Brandt M (2009) Media Server Control API. Tech. rep., Java Community Process, Java Specification Request (JSR) 309

[FERRY2004] Ferry D, Lim S (2004) JAIN SLEE API Specification. Tech. rep., Java Community Process, Java Specification Request (JSR) 22

[FRATERNALI1999] Fraternali P (1999) Tools and approaches for developing data-intensive web applications: a survey. ACM Computing Surveys (CSUR) 31(3):227-263

[IVOV2014] Ivov E, Marocco E, Holmberg C (2014) A Session Initiation Protocol (SIP) usage for Trickle ICE draft-ietf-mmusic-trickle-ice-sip-03. https://tools.ietf.org/html/draft-ietf-mmusictrickle- ice-sip-03, accessed 12 December 2015

[JOHNSTON2012] Johnston AB, Burnett DC (2012) WebRTC: APIs and RTCWEB protocols of the HTML5 real-time web. Digital Codex LLC

[JENNINGS2015] Jennings C, Holmberg C, Alvestrand HT (2015) Negotiating media multiplexing using the session description protocol (SDP) draft-ietf-mmusic-sdp-bundle-negotiation- 23. https://tools.ietf.org/html/draft-ietf-mmusic-sdp-bundle-negotiation-23, accessed 12 December 2015

[KRISTENSEN2003] Kristensen A (2003) SIP Servlet API. Tech. rep., Java Community Process, Java Specification Request (JSR) 116

[ODOHERTY2003] O'Doherty P, Ranganathan M (2003) JAIN SIP API Specification. Tech. rep., Java Community Process, Java Specification Request (JSR) 32

[PARK2009] Park S, Park NS, Kim JT, Paik EH (2009) Provision of the expressive multisensory adaptation platform for heterogeneous multimedia devices in the ubiquitous home. Consumer Electronics, IEEE Transactions on 55(1):126-131

[ROSENBERG2002] Rosenberg J, Schulzrinne H, Camarillo G, Johnston A, Peterson J, Sparks R, Handley M, Schooler E (2002) SIP: session initiation protocol. Tech. rep., Internet Engineering Task Force, Request For Comments (RFC) 3261

[RTCWEB2015] Internet Engineering Task Force (2015) Real-time communication in web-browsers (rtcweb). https://datatracker.ietf.org/wg/rtcweb, accessed 9 December 2015

[SAINTANDRE2011] Saint-Andre P (2011) Extensible messaging and presence protocol (XMPP): Core. Tech. rep., Internet Engineering Task Force, Request For Comments (RFC) 6120

[SALEEM2010] Saleem A, Xin Y, Sharratt G (2010) Media server markup language (MSML). Tech. rep., Internet Engineering Task Force, Request For Comments (RFC) 5707.

[VAQUERO2008] Vaquero LM, Rodero-Merino L, Caceres J, Lindner M (2008) A break in the clouds: towards a cloud definition. ACM SIGCOMM Computer Communication Review 39(1):50-55

[VANDYKE2006] J. Van Dyke et al. (2006) Media Server Control Markup Language and Protocol. Internet Engineering Task Force, Request For Comments (RFC) 4722.

[WEBRTC2015] World Wide Web Consortium (2011) Web real-time communications working group. http://www.w3.org/2011/04/webrtc/, accessed 11 December 2015

[WESTERLUND2015] Westerlund M, Burman B, Nandakumar S (2015) Using Simulcast in RTP Sessions draft-westerlund-avtcore-rtp-simulcast-04. https://tools.ietf.org/html/draftwesterlund- avtcore-rtp-simulcast-04, accessed 12 December 2015

[WESTERLUND2016] Westerlund M, Wenger S (2016) RTP internet draft topologies draft-ietf-avtcorertp- topologies-update-10. https://tools.ietf.org/html/draft-ietf-avtcore-rtp-topologiesupdate- 10, accessed 7 January 2016

[WILLMOTT2013] Willmott S, Balas G (2013) Winning in the API Economy. Available online: http://www.3scale.net/wp-content/uploads/2013/10/Winning-in-the-API-EconomyeBook-3scale.pdf

4.6 Real-time media APIs in smartphone platforms

4.6.1 Description of current SoTA

One of the objectives of the NUBOMEDIA project is to create smartphone APIs to support developers for easily create mobile applications using NUBOMEDIA capabilities and platform. The support is created for android and iOS mobile platforms and thus the SoTA concentrates on the current implementation possibilities on these platforms.

The area of real-time media APIs for smartphones is heavily populated and research on this field has been widely active for years. For smart phones there is a large number of commercial APIs adopting various technologies and protocols. Some examples of commercial solutions that are available for multiple platforms are introduced below :

- Skype Developer Platform for Web [Skype] - Skype web SDK – is a set of JavaScript components and HTML controls enabling developers to build solutions which integrate a variety of real-time collaboration models. The features include presence, chat, audio and video and can be used on various browsing platforms and device endpoints. In the current version presence and chat services are provided using REST-based web services while the support for audio/video and application sharing is supported via downloadable plugin available only for Windows 7 and 8 PCs and Macs. This meaning that the services available for Android devices are limited to real-time presence and chat data. The aim in the future is to provide support for standards such as SIP, SDP, WebRTC as well as IP and PSTN voice.
- Google Hangouts API [Hangouts API] provides the programming interface to Hangouts video calls. The API is implemented as a JavaScript interface that enables listing of Hangout participants, sharing data between instances of the app, controlling the user interface as well as the microphone, camera and speaker settings.
- OpenTok [OpenTok] is an embedded communications platform supporting WebRTC. The real-time communication features include video, audio, messaging and screen sharing. OpenTok provides solutions for mobile devices supporting iOS and Android SDKs.
- SightCall [SightCall]offers real-time video APIs and mobile SDKs for iOS and Android platforms enabling embedding of video calls and video conferencing to applications. The mobile libraries interact with the target device through three low-level APIs: User Interface API to integrate in the host API and to provide real-time communication, Audio/Video API to access and manage hardware resources and Hardware events API to manage and monitor hardware and device-specific events. The low-level APIs are designed to ensure that the mobile SDKs remain device-agnostic and are easily adaptable. The SightCall platform supports WebRTC and has integrated versions for Salesforce and Zimbra.
- Plivo[Plivo] provides mobile SDKs for iOS and Android to integrate a voice call capability to the mobile apps. The call feature is supported by Plivo data centres

distributed globally. The webRTC SDK can be used to integrate voice communication to browser based services.

- The Apizee ApiRTC[Apizee]is a cloud platform offering live chat, voice and video features to be integrated on websites and mobile applications. The ApiRTC library is implemented to be utilized using HTML and JavaScript. The platform implementation is based on WebRTC technology.

Similarly to commercial solutions there is a large number of FOSS real-time media API's as well. There are numerous communication libraries:

- Gstreamer [Gstreamer] is an open source multimedia framework for constructing graphs of media-handling components. It supports applications from simple audio/video streaming to complex audio mixing and video editing. Gstreamer is released under the LGPL and works on all major operating systems such as Linux, Android, Windows, Max OS X, iOS, as well as most BSDs, commercial Unixes, Solaris, and Symbian. It has been ported to a wide range of operating systems, processors and compilers. It runs on various hardware architectures including x86, ARM, MIPS, SPARC and PowerPC.
- Apache Cordova [Cordova]is an open-source mobile development framework. Apache Cordova enables using of standard web technologies such as HTML5, CSS3, and JavaScript for cross-platform development, avoiding each mobile platforms' native development language. Applications execute within wrappers targeted to each platform, and rely on standards-compliant API bindings to access each device's sensors, data, and network status.
- Mediastreamer2[Mediastreamer2] is a lightweight streaming engine specialized for audio/video communication application. The open source library is managing multimedia streams including audio/video capture, encoding and decoding, and rendering. Liblinphone [Liblinphone] integrates the Mediastreamer2 video call features into a single API.

In this context, NUBOMEDIA concentrates specially into WebRTC technologies. There are existing open source possibilities to utilised WebRTC on mobile platforms where most famous are Google's WebRTC [WebRTC] and Ericsson's openwebRTC [openwebRTC] projects that provide the communication stack for android and iOS as native code.

The support of WebRTC is something that is managed at a browser level. In iOS platforms, mobile web browser Safari does not seem to fully support WebRTC, that leaded to a number of initiatives aiming at solving someway this problem.

A first possible solution which is used to develop WebRTC application is the usage of a plugin for Cordova (previously known as PhoneGap). The idea of Cordova is to let developers build mobile applications using HTML, CSS, and JavaScript. At a very basic level is like building a website, using a simple browser (known as WebView) as application to renders files. However, WebView is limited and it often cannot give developers full access to native hardware like camera, accelerometer, and other native capabilities. Cordova's strength is its ability to let developers leverage native hardware of mobile devices through Cordova Plugins.

Cordova Plugins usually consist of a JavaScript file and all the necessary native code files for each mobile platform (Objective C files for iOS, Java for Android, C# for Windows Phone, etc.) Methods defined in the JavaScript file call the methods defined

in the native code, thus giving developers to access native functionality of mobile devices after installing the Cordova Plugin and adding the provided JavaScript file. The plugin, called OpenTok follows this paradigm: the native code creates a video view on top of the Cordova WebView using native OpenTok SDKs, display the camera's video stream in the view, and stream the video to TokBox's servers.

The Cordova plugin approach has been followed also by *eface2face* solutions, aiming at offering a comprehensive tool for video-conferencing and electronic document signing. Part of the code is released on github where a google RTCApp adapted to Cordova iOS with HTML5 can be found together with a Cordova Plugin mapping the W3C Javascript APIs on the eface2face solution.

A slightly different approach instead is the one followed by OpenWebRTC initiative, aiming at offering "a mobile-first WebRTC client framework building native apps". The Solution, which is free and open source, has been driven by Ericsson Research it is based on GStreamer multimedia framework and it is offered as Cross-platform, cross-browser and fully supporting H264 and VP8 video Codecs.

The ambition of OpenWebRTC is to follow the WebRTC standard closely as it continues to evolve. The bulk of the API layer is therefore implemented in JavaScript, making it super fast to modify and extend with new functionality (one such candidate is ORTC).

The initial version of the OpenWebRTC implementation was developed internally at Ericsson Research over the last few years. OpenWebRTC and Bowser were released publicly as free and Open Source in October of 2014. The well known *Bowser* App, initially released in 2012 both for Android and iOS and the removed from stores till its later release in October 2014 is a mobile App built on top of OpenWebRtc. Bowser uses WebRTC in simple self-view and peer-2-peer audio/video chat and it is free on the app store. Beside this, Bowser offers the possibility to manipulate Videos solving some UI problems typical of the mobile side.

Other not-fully standard solutions are available such as PerchRTC which is open source as well but it does not fully comply to the standard.

Specifically on Android platforms, it is possible to utilise a native WebView or third party Crosswalk functionality for creating native applications using web/Javascript development tools, including browser technology support, such as WebRTC. WebView [WebView] is based on Chromium open source project and is a native component in modern Android devices. Crosswalk [Crosswalk] is a HTML application runtime platform to build android and Cordova apps with. It is focusing on mobile performance and efficient use of device API's.

These frameworks have originated from Android development. Currently the WebView is integrated as a separated app to android operating system version 5.0 (lollipop) on, but the support for WebRTC is only at Android 6.0 (Marshmallow). The great benefit of using these is to be able to create fast prototypes for multiple environments. But on the other hand the performance will never be same that in native applications. Also as the apps are built on top of the framework utilising browser technologies it will have some restrictions and usability might not reach the same levels as with native apps trough out all the platforms.

In the scientific community there is a large number released papers on the topics of mobile multimedia communication, including communication protocols, audio/video compression, networking technologies, distributed computing and multimedia service development. The scientific forums related to mobile multimedia communications include for example conferences such as ACM Multimedia[20], IEEE International Conference on Multimedia and Expo (ICME)[21], The International Conference on Mobile and Ubiquitous Multimedia (MUM)[22], EAI International Conference on Mobile Multimedia Communications[23] and International Conference on Advances in Mobile Computing and Multimedia (MoMM)[24].

In some recent publications authors have proposed extension for using WebRTC to distribute all kinds of sensor data from mobile terminals. In [Azevedo2015] the authors propose an extension to the WebRTC APIs to enable general access to sensors. The approach is based on extending the functionality of JavaScript MediaStream API to allow web applications to access sensor devices both in-device and off-device configured into the browser. WebRTC is leveraged to enable the peer-to-peer transmission of sensor data. Another approach to access the native layer of a smartphone device is presented in [Puder2014]. Instead of defining an API, the services on the smartphone interact with the HTML5 application via a specific protocol that allows more flexible solutions, which can be implemented as a separate service running on the smartphone or packaged with the app itself. The authors have implementations for both Android and Windows Phone.

4.6.2 NUBOMEDIA approach beyond SotA

As it was shown in the previous paragraph, a lot of solutions are available, trying to solve specific problems with different level of compliancy to the standards. NUBOMEDIA aims at offering a comprehensive solution also on the client side, including and extending all the functionalities offered by other solutions.

The smart phone framework is included to the NUBOMEDIA to support extensive use of NUBOMEDIA platform.

It will support efficient client side development by PaaS services provided by the NUBOMEDIA services , give an opportunity to create versatile operations for video streams (e.g. video content manipulation and augmented reality) on the server side and offer a standard compliant WebRTC communication interface.

An SDK, plus related documentation and a sample “how-to-use” application is provided, able to manage the WebRTC Stack, the connection to Kurento media server and session management functionalities.
By enabling the communication with Kurento API and indirectly with Kurento Media server, all the NUBOMEDIA functionalities can be consumed on the iOS and android client.
Offering a Native SDK for iOS and Android will overtake some of the limitations that quasi-Web-based solutions such as Cordova suffer in terms of full access to device functionalities. Also native applications will have smaller memory and resource profile

[20] http://www.acmmm.org/2016/
[21] http://www.icme2016.org/
[22] http://www.mum-conf.org/2015/
[23] http://www.mobimedia.org/2016/show/home
[24] http://www.iiwas.org/conferences/momm2015/home

as they don't need to embed full web capabilities just for media communications. Simply to use API also helps mobile developer to establish connection and parametrize application logic in the NUBOMEDIA services.
Moreover the connection to a continuous integration system will ensure that any contribution won't introduce unwanted issues.

4.6.3 NUBOMEDIA outcomes

NUBOMEDIA will provide native mobile SDK's for iOS and Android.

The first version of the iOS and Android SDK is already provided, as well as the first version of the iOS and Android room and tree API's and they will be maintained and evolved till the end of the project also by means of the integration of a Continuous Integration system.

4.6.4 References

[Apizee] https://apirtc.com/
[Azevedo2015] João Azevedo, Ricardo Lopes Pereira, and Paulo Chainho. 2015. An API proposal for integrating sensor data into web apps and WebRTC. In Proceedings of the 1st Workshop on All-Web Real-Time Systems (AWeS '15). ACM, New York, NY, USA, , Article 8 , 5 pages.[Cordova] https://cordova.apache.org/
[Crosswalk] https://crosswalk-project.org/
[Gstreamer] http://gstreamer.freedesktop.org/
[Hangout API] https://developers.google.com/+/hangouts/api/
[Liblinphone] http://www.linphone.org/technical-corner/liblinphone/overview
[Mediastreamer2] http://www.linphone.org/technical-corner/mediastreamer2/overview
[OpenTok] https://tokbox.com/
[OpenwebRTC] http://www.openwebrtc.org/
[Plivo] https://www.plivo.com/
[Puder2014] Arno Puder, Nikolai Tillmann, and Michał Moskal. 2014. Exposing native device APIs to web apps. In Proceedings of the 1st International Conference on Mobile Software Engineering and Systems (MOBILESoft 2014). ACM, New York, NY, USA, 18-26.
[SightCall] http://www.sightcall.com/
[Skype] https://msdn.microsoft.com/en-us/library/dn962133(v=office.16).aspx[webRTC] http://webrtc.org/
[WebView] http://developer.android.com/reference/android/webkit/WebView.html

4.7 Cloud Videoconferencing APIs

Videoconferencing (VCF) is gaining relevance and its adoption in different environments is increasing due to the economic and social advantages it provides. The adoption of usable APIs (Application Programming Interfaces) -typically accessible through SDKs (Software Development Kits)- is a trend increasingly demanded by developers. The case of VCF is not an exception, and thus nowadays it is emerging more and more initiatives aimed to provide VCFs APIs.

This section provides a brief state-of-the-art review on VCF APIs. Then, the main advantages of the NUBOMEDIA approach for this domain are discussed.

4.7.1 Description of current SotA

The contribution of this section is two-folded. First, it reviews several scientific contributions related to VCF APIs. Second, it presents a summary of some companies and solutions that are investing effort in creating tools and solutions in this area.

We find an interesting scientific reference on VCF APIs on the paper by Rodriguez et al. [RODRIGUEZ2009]. In this piece of research the authors propose a REST API for integrating collaborative Videoconferencing as a Service (VaaS). This approach enables the transformation of a standard client-server service into a Cloud Computing service. The approach presented in this work has been named "Nuve". It is based on Adobe Flex/Flash technology on the client side and Java on the server. The media server used in Nuve is Red5, which allows for voice/video communications to be established among the clients. The scalability is the main drawback of this work. It presents a prototype of Nuve providing videoconference rooms where up only to six participants communicate with each other via audio and video. This figure is quite small, and so Nuve cannot be taken into consideration for a production-ready environment.

On the other hand, Gasparini et al. provides an insight to current trends on VCF, classifying the existing approaches available in the literature into two types [GASPARINI2013]:

1. Hardware-based (HW) VCF (also called dedicated systems) comprises VCF solutions that require specific hardware-other than standard PCs, Macs and smartphones.
2. Software-based (SW) VCF comprises VCF software tools that can be installed and run on standard PCs, Macs and smartphones hardware.

Regarding SW VCFs, this work analyses the following tools:

- Skype [SKYPE] is a freemium VCF tool, owned by Microsoft.
- WebEx [WEBEX] is a freemium VCF tool for group sessions, owned by Cisco.
- Hangouts [HANGOUTS] is a free tool by Google, which can be used to videoconference, to organize group sessions and to broadcast live events.
- Vidyo [VIDYO] and ViTAM [VITAM] require a server infrastructure to be deployed and a client part that can be run on the web, iOS and Android.
- OpenMeetings [OPENMEETINGS] is an open-source project of the Apache, which provides VCF and other tools for holding and managing
- OpenTok [OPENTOK] consists on server-side and client-side libraries for the web, iOS and Android that need to be deployed or invoked to create customized workflows.
- VSee [VSEE] consists of software-based VCF without the need for a server deployment. Security is based on an end-to-end key negotiation and encryption.

This survey analyzes whether or not these tools meets some requirements, namely:

- Recording: some applications need to collect information or evidence.
- Deployment: need for a custom server deployment.
- Privacy and security: user information is subject to the compliance of several national or regional data protection laws.
- Mobility: VCF needs to be available not only in standard PC and Mac devices but also on mobile devices such as smartphones and tablets running Android, iOS, BlackBerry or other operating systems.

- Interfaces: when VCF needs to be used by wide groups of population with different technical backgrounds, interfaces need to be minimalist to increase usability as much as possible.
- Workflow: sequence of actions that need to be carried out before a VCF session can be established and during its progress until it is finished. This can be classified in the following groups:
    - Call: contacts lists are used to see whether users are online or offline. For online users, a call request can be sent, which can be accepted or rejected by the receiving endpoint.
    - Group session: a group of users has a joint meeting in which all participants can interact with the rest.
    - Broadcast: relies on the real-time transmission of a live event to the whole world. Users can freely join and watch the event while interacting with other users.
    - Consultation with waiting room: replicates the typical rendez-vous operation in real life, where patients with an appointment go to the health premises and wait in a waiting room for the corresponding professional.
    - Call center: through VCF, is prescreened by an intermediary prior to being redirected to the most suitable interlocutor.
    - Video surveillance for impaired users: VCF is used for remotely monitoring users with mobility problems or other impairments.
- Integration: Different approaches can be followed to use or create new VCF workflows in existing or new applications:
    - API Integration: an API consists of a series of calls that enable to invoke or start remotely VCF sessions or related functionality.
    - SDK Integration: a SDK is a set of libraries, tools, examples and documentation that can be used by developers to create custom VCF-enabled applications.

	Skype	*WebEx*	*Hangouts*	*Vidyo*	*ViTAM*	*OpenMeetings*	*OpenTok*	*VSee*
Recording		✓	(✓)[1]	✓	✓	✓	✓	(✓)[2]
Own server deployment				✓	✓	✓	✓	-
Privacy & security		(✓)[3]		✓	✓	✓	✓	✓
Own user management					✓		✓	
Mobility								
iOS	✓	✓	✓	✓	✓		✓	(✓)[4]
Android	✓	✓	✓	✓	✓		✓	
PC	✓	✓	✓	✓	✓	✓	✓	✓
Mac	✓	✓	✓	✓	✓	✓	✓	✓
Asymmetric interfaces					✓	✓	(✓)[5]	
Workflow								
Call	✓	✓	✓	✓	✓	✓	✓	✓
Group session	✓	✓	✓	✓	✓	✓	✓	✓
Broadcast			✓		✓			
Consultation w. room					✓			
Call centre					(✓)[4]			
Video surveillance					(✓)[4]			
Customized				✓	(✓)[6]		✓	
Integration								
API	✓	✓	✓	✓	✓	✓	✓	
SDK				✓			✓	

[1] Through YouTube
[2] In one of the endpoints
[3] Personal data management contract needed
[4] Planned
[5] Depends on workflow customization
[6] On demand

Figure 42 Review of VCF tools [GASPARINI2013]

Regarding products and companies working on VCF, we find an important online source on [WEBRTCINDEX.COM]. This website provides a freely available index of WebRTC vendors and services. Focusing on VCF solutions that expose APIs or SDKs, the following table summarizes several significant initiatives:

Product/company	Logo/website
Kandy is a real time communications software development platform, built from. It is a PaaS solution that includes APIs, SDKs and quick starts (pre-built applications like video shopping assistance). Kandy has been developed by **GENBAND**, a company that provides IP-based real time communications software solutions and products for fixed wireline, mobile, and cable service providers as well as large enterprises.	https://www.kandy.io/ https://www.genband.com/
Respoke is a communications platform built for web and mobile developers. It provides simple APIs, libraries and SDKs abstract the complexities of real-time communications, allowing developers to focus on user experience and business logic.	https://www.respoke.io/
Licode is an audio/video conferencing solution. It is based on WebRTC technologies. It allows developer to include VCF rooms on web applications by means of JavaScript APIs. It provides a media server which allows media streaming and recording. **Lynckia** is an open source project that provides conference rooms where participants are able to share their video, audio and data streams.	http://lynckia.com/licode/ http://lynckia.com/
Jitsi is an audio/video Internet phone and instant messenger written in Java. It supports some of the most popular instant messaging and telephony protocols such as SIP, Jabber/XMPP, AIM, ICQ, MSN, Yahoo! Messenger. It is based on the OSGi architecture using the Apache Felix implementation.	https://jitsi.org/
Janus is an open source web conferencing and collaboration platform. It is a WebRTC general purpose gateway, providing capabilities to set up media communication with browsers and relaying on RTP/RTCP between browsers and the server-side application. Janus has been created by **Meteecho**, an academic spin-off of the University of Napoli.	http://www.meetecho.com/ https://janus.conf.meetecho.com/

4.7.2 NUBOMEDIA approach beyond SotA

The cloud videoconferencing APIs proposed in the context of NUBOMEDIA, mainly the Tree and Room APIs, provide powerful tools for developers to build complex VCF applications in an easy way. These APIs can be seen as high-levels services (based on the client-server architecture) implementing common VCF topologies, i.e. the Room (full-duplex real time communication among several participants) and Tree (real time

broadcasting from one presenter to a large number of viewers). Both APIs relies on WebRTC as media transport, which is provided by other low-level NUBOMEDIA API such as the Signaling and Media API. These APIs have the following differential characteristics:

- Server-side and client-side APIs enabling to fully customize the logic of the application both at client and at application server.
- Integration with advanced media processing capabilities.
- Integration with legacy RTP systems

4.7.3 NUBOMEDIA outcomes

The main outcomes of the project are the Room and Tree APIs, that enable respectively to create videoconferencing services based on room and tree topologies. At the time of this writing, these APIs are on their first versions of development and provide basic features.

During the third year of the project, further enhancements are planned so that, depending on the requirements being issued by the industrial partners of the project, the following features may be implemented:

- Screen sharing
- Transparent simulcast
- VP9 support leveraging SVC features.
- Dominant speaker detection
- Recording

These APIs have been integrated as part of the Kurento Open Source Software project, and shall be exploited in the context of the NUBOMEDIA community.

4.7.4 References

[RODRIGUEZ2009] Rodríguez, Pedro, et al. "Vaas: Videoconference as a service." *Collaborative Computing: Networking, Applications and Worksharing, 2009. CollaborateCom 2009. 5th International Conference on*. IEEE, 2009.

[GASPARINI2013] Gasparini, Claudio D., et al. "Videoconferencing in eHealth: Requirements, integration and workflow." *e-Health Networking, Applications & Services (Healthcom), 2013 IEEE 15th International Conference on*. IEEE, 2013.

[SKYPE] http://www.skype.com

[WEBEX] http://www.webex.com

[HANGOUTS] https://hangouts.google.com

[VIDYO] http://www.vidyo.com

[VITAM] http://vitam.udg.edu

[OPENMEETINGS] http://openmeetings.apache.org

[OPENTOK] http://www.tokbox.com/opentok

[VSEE] http://vsee.com

[WEBRTCINDEX.COM] https://webrtcindex.com

4.8 Enhancing real-time media developer efficiency

In the past 2-3 years there has been a growing movement and a proliferation of development environments and frameworks in an attempt to enhance the productivity of developers. The number of new programming languages and data management technologies has exploded and we have passed from development environments lacking in user-friendliness and isolated to a large extent towards more integrated tools that provide almost everything. In fact the *mobile-first* trend has taken development by storm and has created needs of its own.

To reach a coherent analysis we have focused on technologies present in the NUBOMEDIA project, and more specifically on those with support for WebRTC. There are limited resources which provide some kind of compilation of developers involved in this area. We have explored WebRTCHacks [WEBRTCHACKS.COM] (a bit outdated, the last review is from 2013) and WebRTCWorld [WEBRTCWORLD.COM], as they perform regularly a survey of developers active on this topic.

Additionally we have reviewed [DEVELOPERECONOMICS.COM], a well-known report on the state of development tools (platforms, APIs, segments, …). In recent years more focus has shifted towards mobile applications and their particular needs. One feature of this report is that it does not contain a single reference to real-time environments, so it does not contemplate this as an area of interest or even a category worth being mentioned. One conclusion is that they may consider the real-time arena as something marginal, which is in sharp contrast with the investments in and around WebRTC players worldwide.

4.8.1 Description of current SoTA

From the review performed we did find that vendors in the WebRTC arena limit their contribution to developers merely to downloading SDKs or JavaScript libraries, so they can be included in their typical development environments. We include here only those with a specific orientation to real-time media developers. We have left out those offering merely a platform, with no support for developers, or vendors offering for-fee development services.

None of the vendors reviewed provides a “composition” tool to ease the work of the developer. We have summarized the information in an easy to read format, highlighting special features where appropriate.

- **AddLive** (now a SnapChat company): provides developers with an API that allows to add live video and voice to applications.
- **Apizee**: provides the webRTC API to simplify the integration of voice and video chat in web applications.
- **Bistri**: is an open video chat platform, based on WebRTC. It is a PaaS offering and provides a JavaScript API as well as SDKs for Android and iOS.
- **Bit6**: allows developers to integrate real-time communications into mobile and web applications. The platform is offered as software-as-a-service and allows developers to add calling, messaging and many other communication features

with just a few lines of code. There are SDKs for Android, iOS and JavaScript, and a plugin in case of using Cordova.

- **Frozen Mountain Software**: With a history dating back 5 years Frozen has been involved in real-time communications software platforms and SDKs, and now has a full P2P platform, including full support for WebRTC communication. The collection of SDKs available is the largest of the companies reviewed, and they include iOS, Android, several browsers, and several operating systems (Linux, Windows, Mac). It builds on top of the Xamarin platform for development (UI- based IDE).
- **Prologic Technologies**: uses TokBox as a building block. They have created webRTC-based solutions for domains like Health Care/ Tele Medicine, Virtual Interviews, Skill training of local Craftsmen, or Online Recruitment. All development support is delegated to TokBox' OpenTok platform, so it presents the client-side libraries (JavaScript, Android, and iOS), and the server-side SDKs (Java, PHP, Python, Ruby and Node.js).
- **PubNub**: runs a globally distributed "Real-Time Network", a cloud service that developers use to build and scale large real-time apps, without worrying about infrastructure. The WebRTC SDK allows developers to use PubNub for signaling, and enhance their WebRTC apps with features like Presence and Storage & Playback. They have a JavaScript SDK for WebRTC, but there have been no new versions in the last 2 years.
- **Sinch**: provides SDKs for iOS, Android, JavaScript and REST APIs to build applications for voice and video chat. According to their information only the video component uses WebRTC.
- **Tokbox**: The OpenTok platform allows developers to integrate live, face-to-face video directly into their website, iOS, and Android apps using WebRTC. OpenTok provides all the development needs from the infrastructure side (scaling, stream quality optimization, or even analytics).
  NOTE: Since OpenTok has been around for some time and has reached a satisfactory level of stability, it is the platform being used by several vendors (according to the information in [WEBRTCWORLD.COM].
- **Zingaya**: Developers can use VoxImplant as a cloud platform to embed real-time communications into any web or mobile application including already existing one without any additional infrastructure. VoxImplant, developers can build enhanced communication applications and integrate them with existing web or mobile services a very short time, thus reducing development costs. Zengaya's VoxImplant is limited to voice and video communication. It provides SDKs for Android, iOS and JavaScript.

Summary:

- Support for developers in most of the companies reviewed above is limited to providing APIs (mainly JavaScript) and SDKs (iOS, Android, Java, …) and using REST Web Services for integration.
- There is no unified vision on features or deployment or integration with other "legacy" systems.
- Integration of code is done at the text editor level (importing/including libraries).
- Lack of visual GUI type of development.

- Limited applications scope: video/voice conferencing, healthcare, training, one-way broadcasting… (no not taking advantage of other areas of applications: security/surveillance, entertainment – music, games, …)

4.8.2 NUBOMEDIA approach beyond SotA

In recent years there has been an emphasis on the creation of full development platforms encompassing all that is needed to build and operate scalable environments. Integration of different technologies is facilitated by the abundance of API-based development as well as the prevalent REST-mediated communications. Real-time development is getting more traction, since the movement to get data at shorter intervals is moving front center. This is nowhere more true when considering how live / streaming interactions have been growing at an accelerated pace.

As we have seen, development around real-time media, and more precisely WebRTC is fragmented, not very consistent and most times consists of assorted libraries to be integrated into larger collections of code using traditional tools. NUBOMEDIA will move beyond this current state with a visual component GUI tool, that will provide a simplified vision of the elements as exposed by Kurento Media Server: nodes and connections to form a pipeline. Following the philosophy of abstracting the underlying infrastructure components of NUBOMEDIA, the developers creating real-time applications will be freed from the tedious and repetitive tasks associated with infrastructure provisioning for development. The tool will increase developer productivity by generating templates of applications, so they will only need to *draw* which elements are needed and how they need or want to link the nodes in a pipeline.

What will be performed in NUBOMEDIA is in contrast with what we have found (as detailed above). We found that currently there is no consistent view, unlike that proposed by NUBOMEDIA, so this will be a clear step further from the current state of development in the WebRTC arena.

4.8.3 NUBOMEDIA outcomes

Along the years ZED has developed their own tooling on top of established game engines, because they either lacked functionality or were too cumbersome to use.

ZED's main outcome from NUBOMEDIA in this particular topic is a GUI tool that will simplify part of the game creation lifecycle. During the development of a game several different types of tools come into action, and for the specific case of multiplayer games, infrastructure definition takes a lot of time, both in the conceptual phase and the server definition phase. These phases are devoted to low level infrastructure design and setup, before the actual game mechanics can even be started. The tool coming out from NUBOMEDIA, with the visual composition capabilities will boost the overall development team productivity, since that phase could be shortened and then made available for immediate testing. This modification of the development calendar will reduce both time and costs, and consequently also provides ZED with a competitive advantage, by including this tool (with future evolutions and enhancements) into the game creation lifecycle.

4.8.4 References

[DEVELOPERECONOMICS.COM] https://www.developereconomics.com/

[WEBRTCWORLD.COM] http://www.webrtcworld.com/webrtc-list.aspx

[WEBRTCHACKS.COM] https://webrtchacks.com/vendor-directory/

[ADDLIVE] http://www.addlive.com/

[APIZEE] https://apizee.com/

[BISTRI] http://developers.bistri.com/

[BIT6] http://bit6.com/

[FROZEN MOUNTAIN SOFTWARE] http://www.frozenmountain.com/

[PROLOGIC TECHNOLOGIES] http://www.prologic-technologies.com/

[PUBNUB] https://www.pubnub.com/

[SINCH] https://www.sinch.com/

[TOKBOX] https://tokbox.com/developer/

[ZINGAYA] http://voximplant.com/

[XAMARIN.COM] https://xamarin.com/

[TOKBOX.COM] https://tokbox.com/developer/

5 Real-time media in vertical segments

5.1 Real-time media for video surveillance and security

In this section we are going to review the state of the art of real time media for video surveillance and security. We are going to start with a definition of Video Surveillance. According to Wikipedia, Video surveillance implies the use of video cameras to transmit a signal to a specific place, on a limited set of monitors. Though all video cameras fit this definition, the term is most often applied to those used for surveillance in areas that may need monitoring.

The main use of video surveillance cameras has to do with crime prevention. However, video-surveillance systems can be used in different locations to fulfill different goals. An interesting segmentation of the system type is provided by IMS research [IMS] by a segmentation of the video-surveillance market by industry segments until 2014 (see figure below).

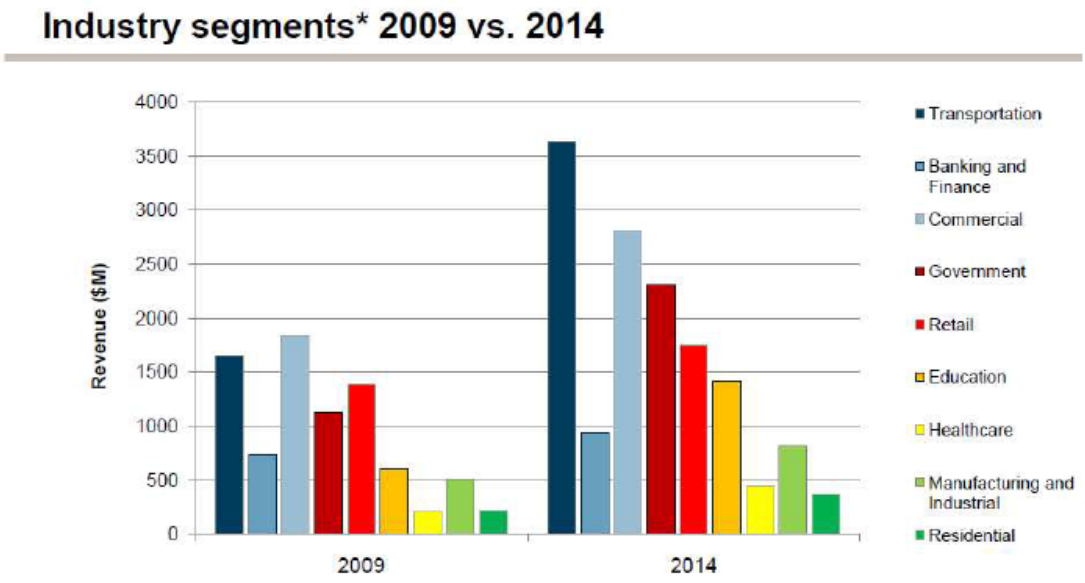


Figure 43: Video Surveillance industry segments

In the previous figure we can see a very important development of the market (between 2009 and 2014, especially in transportation systems, government and commercial. The video-surveillance systems in the IMS-identified industry segments appear to be very different:

- Commercial, Retail, Banking and Finance video-surveillance systems are most of time privately owned and privately operated. Their goals are to protect goods rather than persons. The data produced by these systems may be used to prosecute persons (customers, employees..).

- Government systems are mostly dedicated to the surveillance and protection of goods and persons within critical infrastructures (ministry premises, nuclear facilities..). They are operated most of time by public forces, in deep relation with access control and intrusion detection systems. This segment also embeds urban-security systems dedicated to the surveillance of public spaces in order to ensure citizens and public goods protection (against volunteer actions, natural disasters, accidents and so on)

- Transportation systems, which do represent an important part of the video-surveillance market, are often dual-use systems. The typical infrastructures supervised are metro, main lines (stations and on-board) and airports. On the one hand, the video-surveillance system is used in conjunction with the operation of the system or the infrastructure (surveillance of a train position in a station, of the state of electromechanical devices, of queue length in airports and so on). The operation is for this use performed by private operators belonging most of the time to the organization responsible for the operation of the infrastructure. On the other hand, a police use of the video-surveillance system is often performed for protection of goods and persons. For example, the Paris metro network is equipped with thousands of video-cameras, used both by RATP operators (RATP is the operator of the Paris metro), and French Police, but in 2 separated supervision rooms.

After presenting an overview of the different segments, where video surveillance system are more used, we are going to differentiate the two main systems or architecture for this kind systems. The main differences between these systems are the cameras used to record video. These cameras can be analog or IP.

In the traditional analog video surveillance system, security cameras capture an analog video signal and transfer that signal over coax cable to the Digital Video Recorder (DVR). Each camera may be powered by plugging in the power supply right at the camera or by using RG59 Siamese cable which bundles the video and the power cables. The DVR converts the analog signal to digital, compresses it, and then stores it on a hard drive for later retrieval. Intelligence is built into the DVR to handle such things as scheduling, and digital zoom. Monitors for viewing the video are connected to the DVR, or it can be set up to publish over an internal network for viewing on PCs. The DVR can also be set up to broadcast over the Internet and can add password protection and other features. When broadcasting over the Internet, the video for all of the cameras is transmitted as one stream (one IP address). Therefore, it is very efficient. In the following figure, we can see an example of the architecture for analog video surveillance systems.

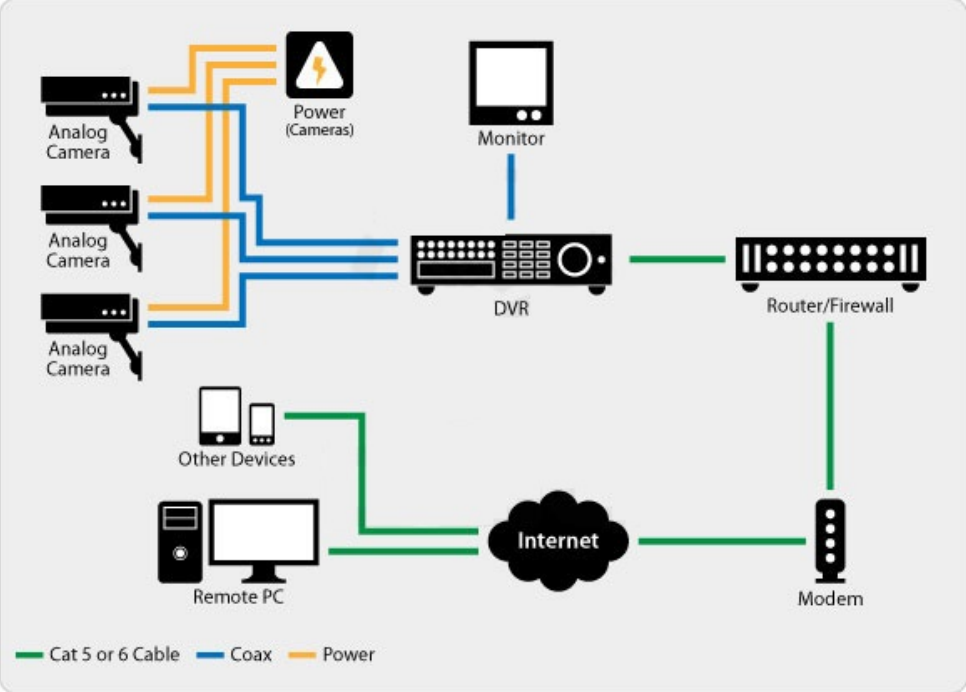


Figure 44: Analog Video Surveillance System Architecture

In the IP world, each network camera captures an analog image but immediately converts it to digital inside the camera. Some digital processing can happen right at the camera, such as compression. The digital video stream is then broadcast over the local area network (LAN) using Ethernet (CAT5 or CAT6) cable. Power is supplied to the cameras through the Ethernet cable via Power-Over-Ethernet (POE) adapters built into the cameras and at the (POE enabled) switch. The Ethernet cable for each camera is plugged into the switch which feeds into the network hub. As with all network devices, some set-up needs to be done for each network camera to set up its IP address and other identifying attributes.

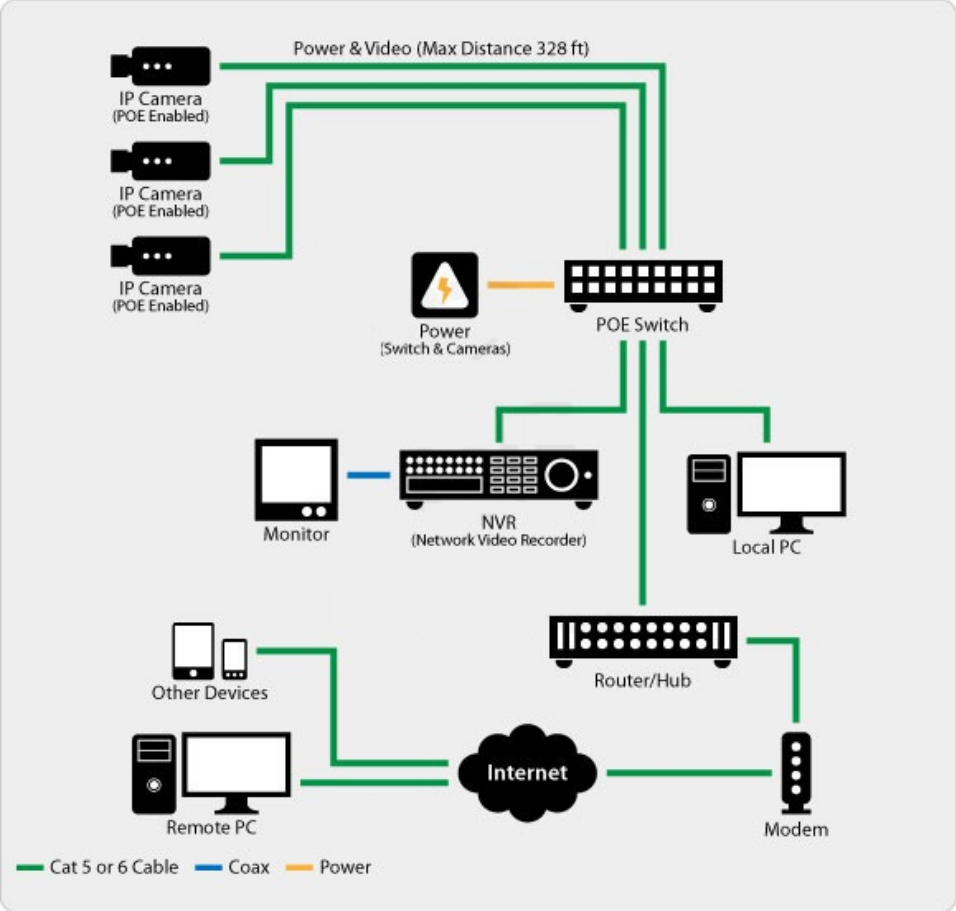


Figure 45: IP Video Surveillance System Architecture

A Network Video Recorder (NVR) performs the same function as its DVR cousin in the analog world. It captures each camera's signal, compresses, and records it. The main difference is that the video feeds are digital (and much higher resolution, you can see the difference produced by the resolution between and analog and IP camera in the following figure) and not analog. Software built into the NVR provides features such as intelligent search and zoom, etc. The NVR combines the video streams from the cameras and handles the broadcast over the LAN and internet for local and remote viewing.

Figure 46: Analog vs IP cameras

The IP systems under the right bandwidth conditions in a LAN will not have latency higher than 500 milliseconds. In the same way, and always with the right bandwidth conditions, if a user attempts to access the system from outside the LAN, the latency would increase approximately 100 ms.

A part from that kind of systems, companies are appearing now doing video recordings, and in some cases video processing with different algorithms, in the cloud, using for this the cloud providers which have been seen on section 5.4.1. Such systems are known as **Video Surveillance as a Service (VSaaS)**. VSaaS is an easy, smarter way to manage IP cameras. It is a hosted platform, meaning that viewing, recording, playback is all done via the web. We could say that the difference among VSaaS, IP and analog systems is the use of a unit to store and process the video, since VSaaS system does this over the cloud. Therefore, we could also consider that the other two systems can make video streaming over internet. On the one hand the analog system, although it has to convert the image from analog to digital introducing greater latency, the remote users could connect the DVR to see the live video or video storage. On the other hand, the NVR offers the same features but without the conversion of the image, and therefore minimize the image delay. Due to the fact that the NUBOMEDIA project is based on video processing and stream in real time over the cloud, we will focus from now on at VSaaS.

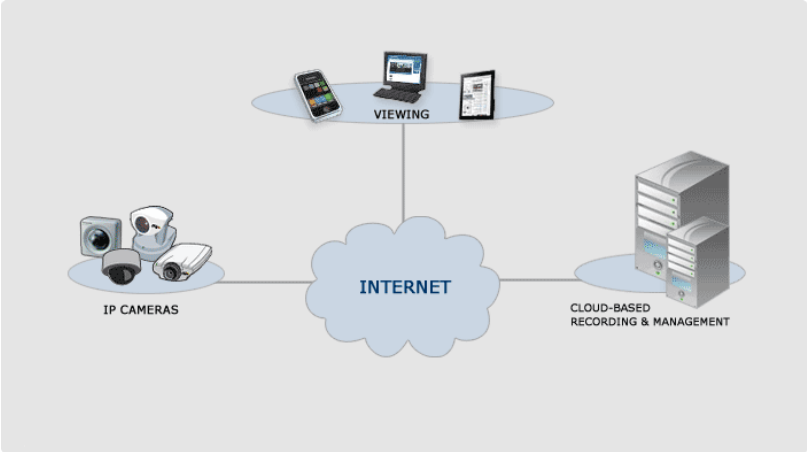


Figure 47. Video Surveillance as a Service (VSaaS)

It is also important to highlight that video surveillance systems usually have some kind of smart algorithms, a part from the typical operations over the video streams such as image resizing, video storage or video compression. Some examples of these smart algorithms are motion and face detection.

5.1.1 Description of current SoTA

Once we have seen the main segments of video surveillance systems and its main architectures, we will see on this section difference commercial and FOSS solutions, as well as the current scientific state of the art.

Commercial solutions

When we talk about commercial solutions in the Video Surveillance area over the cloud, it is mandatory to start with one of the largest transactions in recent years in the

video surveillance market. We are talking about the purchase of the **Dropcam** Company by Google [GOOGLE] in june 2014 by $555 million. After the company was bought, it was renamed to Nest [NEST]. This face reinforces one of the theories that we discussed on section 5.4.1 , by which tech giants bet for computer vision and in this case for the video surveillance market.

Nest has created a camera which is a WiFi-enabled security camera ($149 or $199, depending on video quality) that requires little-to-no-effort to maintain. You plug it in, get it up on your WiFi, and you're set. If you just want to be able to check in on your cameras remotely, that's free through the Nest app. However, if you want Dropcam to keep an archive of recorded footage on their cloud servers, that'll cost you anywhere from $100 per year you get 10-day online video history and for $300 per year that goes up to 30 days of history. The Nest Cam can capture 720p or 1080p video and connects to both 2.4GHz and 5GHz WiFi. Previous the purchase by Google, Dropcam used Amazon web services as its cloud provider. Right now, although we have not found information, nest cam may be using google cloud instead of amazon. The services provided are the following:

- **Video History**: Nest Cam continuously records and stores 10 or 30 days of footage.
- **Alert summary** based on motion and audio detection.
- **Activity zones:** Select the area of the images that you care about and get personalized alerts when something happens there.
- **Video sequences:** Save and downloads video sequences.

Figure 48: Next (left side) and Axis cameras (right side)

Another important company which we are force to comment is Axis [AXIS]. Axis is the company which sells more IP cameras around the world. They have cameras a wide portfolio of solutions for small, mid-size and large scale systems. As for cloud solutions, Axis does not offer a full solution. Their cameras can transmit video over the network. In addition, they can upload the video to the cloud and stored it using a specific service provider. Then, the service provider can manage everything related to the video storage. Another possibility that Axis offers to their clients is the option to record the video with a network attached storage solution (NAS), through which axis needs to do the video streaming in order to record the video in some specific server on the network. Finally, it is important to emphasis that the axis cameras offers the option to integrate algorithms in their cameras. A good example of this is Visual Tools, which is a partner of Axis and the NUBOMEDIA project and has developed several algorithms to be embedded in Axis cameras, such as motion detection and POS (Video management point of sale).

A part from Nest and Axis, which can be the more relevant example about VSaaS, there are a number of companies which also try to take their video surveillance solutions to the cloud. We are going to see some examples below:

- **CamaraManager** [CAMMANAGER] a Panasonic [PANASONIC] company, offers its own video surveillance solutions on the cloud. The main important features of this solutions are:

    - **live video and audio**. In addition they offer the possibility to see multiple camera images, up to 25 different cameras, at the same time.
    - **Storage**: After connecting your camera to the Cameramanager application, the video immediately & automatically start to be saved to the cloud. You can choose to store between 24 hours, 7 days or 30 days and video quality up to Full HD. In addition, this software allows the user to easily schedule when they want to start recording to the cloudm choosing between continuous recording, specific days & hours recording or only record when motion, heat or sound is detected.
    - **Detection:** this application offers the option to record video when motion, heat and audio is detected. In addition, the user can set specific detection areas within the images.
    - **Alarms:** when some detection has been captured by the system, the application can send you a notification to your computer, Smartphone or tablet.

- **Neovsp** [NEOVSP]: The cloud solution is a platform that intelligently stores and manages video content from IP and analog cameras and efficiently delivers content to multiple user devices such as web browsers, mobile phones and media sets. This solution is aimed at Telecommunications companies (telcos), data centers, ISPs, and cable service providers, so that they can use their global infrastructure for another value-add service by offering video surveillance capabilities to their customers on a subscription basis. We highlight the following features and services about this solutions:

    - **Scalability**: the solution allows for system growth parallel to the growth of the client base, by adding servers to the cluster used to manage the video as needed.
    - **Formats**: this solution can broadcast the media in a variety of popular high-quality formats, such as MPEG-4, 3GPP, H.264 Microsoft Windows Media (WMV), Apple QuickTime (MOV) and Adobe Flash. High-Definition (HD) quality is natively supported.
    - **Multiplatform**: this solution is accessible via any web browser, mobile/smart phones (including Android and iPhone), and tablet PCs
    - **Storage**: video data content is stored in a cluster platform. This platform stores video data content from unlimited sources, and allows remote access.
    - **Video streamed content**
    - **Video Pos integrated solutions**

- **Iveda** provides a platform called **Sentir** [SENTIR] for video management with video streaming and storage technology. It offers the video surveillance functionality of traditional security industry DVRs (digital video recorders) and NVRs (network video recorders), all delivered from the cloud as an application.

A part from the video streaming and storage capabilities, this solution does not seem to offer more services related to Video Surveillance. Although they offer the option to back up the video into a slave server. In the same way as the Neovsp, the Sentir platform is aimed at telcos, data centers and ISPs.

- **Smartvue** [SMARTVUE], this company offers different Video Surveillance service on the cloud:

    - **CloudVue:** Fast and secure remote video management and monitoring for one to thousands of locations. CloudVue optimizes video delivery for any bandwidth.
    - **CloudDrive:** On demand cloud video storage service that is secure and redundant
    - **MapVue**: See surveillance locations on a global map and quickly monitor video and system status. Use geographic maps populated with interactive camera icons that show live camera previews when you hover over them. They are color-coded to notify you of server and camera statuses
    - **Other services**: A part of these services, they are working in other ones, such as *CommandValue* through which the users will be able to get reports about the cameras and cloud servers, or *Contribute* which is a social surveillance on demand that provide to the security staff an application that allow them to connect to the platform and send the video through their mobile phones or tablets.

- **Stratocast** [STRATOCAST] is another company with offer a video surveillance cloud solution. They technological partners are Axis which provides them the cameras and Microsoft Azure as a cloud provider. Though its solution, Stratocast offers video streaming and storage. Furthermore, they offer intelligent video management through which the user can monitor multiple cameras, zoom-in on points of interest, playback recorded video to find incidents and see key events. Moreover, they offer an intelligent timeline indicates when activity has occurred, for example based on motion detection, in order to rapidly search through your recordings. Finally, they introduce a permanent storage feature for those video sequences that the users need to maintain saved.

A part from these commercial solutions we can find others which offer solutions with similar features. **Unifore** [UNIFORE], **SecureWerk** [SECUREWERK], **Ynetworks** [YNETWORKS]are other examples of companies which also offer video surveillance solutions on the cloud.

FOSS solutions

When we talk about FOSS solutions in the video surveillance sector over the cloud, we have found a limited number of platforms. Here, we will talk about three solutions. Although, only one of these solutions is deployed on the cloud. The other solutions are open source software in the video surveillance sector, and we have included them on this part of this deliverable, since they carry out video streaming over the network, for example for live video.

Possibly the most interesting platform is iSpy [ISPY]. iSpy is an open source Video surveillance software, which offers the option to deploy its software over the cloud.

iSpy software is free to use a local level, that is why we also consider free, but when we want to use the software on the cloud, we need to pay between $8 and $49 to deploy the product in the specific server over the cloud. The source code is available on github (https://github.com/ispysoftware/iSpy) . In the same line as the commercial solutions iSPy provides to the user the following main features:

- Access live video and audio.
- Record video stream and audio.
- Recording schedule
- Motion detection
- Alerts by email or SMS
- Unlimited number of cameras

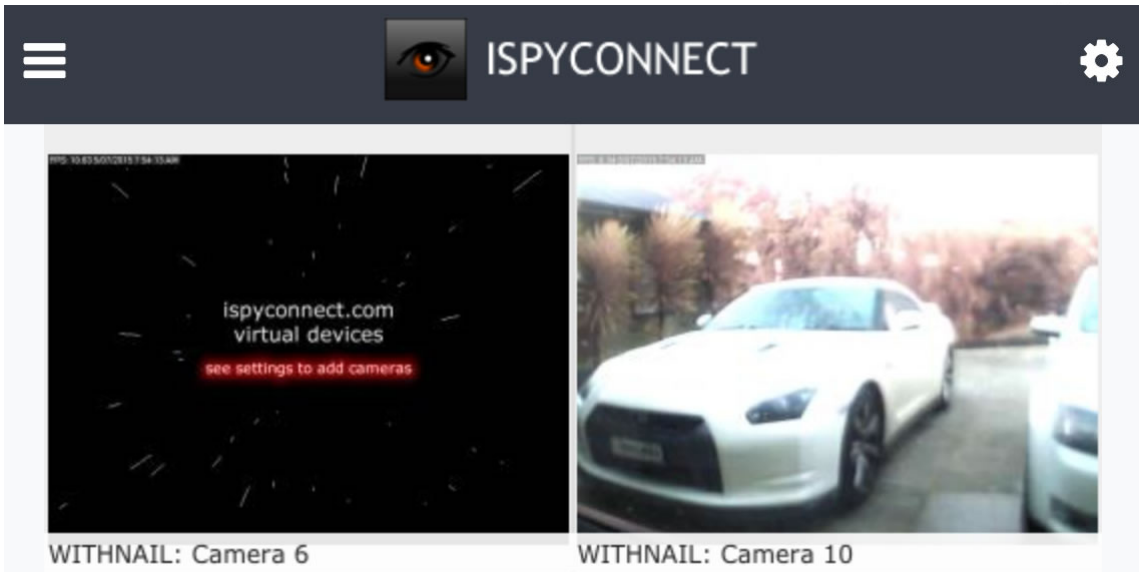


Figure 49: iSpy FOSS solution

The following FOSS solution is **ZoneMinder** [ZONEMINDER]. It is a free and open source Closed-circuit television software application developed for Linux which supports IP, USB and Analog cameras. This solution does not offer the possibility to deploy it on the cloud. However, it does offer the remote access to live video and remote playback of the video storage. Therefore, it performs video streaming over the network and for this reason has been included on this section. Its source code is available on github (https://github.com/ZoneMinder/ZoneMinder). It offers the similar group of pictures than the other solutions previously seen. These features are:

- Video capture
- Video recording,
- Monitoring and live video
- Definition of region of interest within the image
- Motion detection
- Event notification by email or SMS

The last FOSS platform is OpenVSS [OPENVSS](Open Video surveillance System). It is free and open source software, and you can find the source code at google code (https://code.google.com/p/openvss/). The main point to highlight is that as almost all the projects located at google code, OpenVSS has been archived and it is just available in read only mode. Another important point is that it does not seem to offer a cloud

solution. The main features of this platform are video analyzing, recording and indexing, with searching and playback services. It supports multi-cameras platform, multi-analyzers module (OpenCV integrated), and multi-cores architecture. The OpenVSS software consists in three main packages:

- **OpenVSS server**: an online video processing system for analyzing, recording and indexing
  - VsMonitor: Video analyzer management.
  - VsAdmin: remote connection tool for activating the analyzer, recorder and alerter.
  - VsSystem: system configuration tool.
  - VsService: service configuration tool.

- **OpenVSS client:** client application
  - VsLive: video live views.
  - VsPlayback: video searching and playback

- **OpenVSS SDK**: additional plug-ins
  - VsAnalyzerSDK: analyzer plug-in, source code generator integrated with OpenCV 2.0

Scientific publications

Researchers are also interesting on deploy video surveillance system on the cloud. If we loof for the term "Video Surveillance, Cloud Computing" at google scholar, we can obtain 2.400 results in this year, which is a 20 percent of the result obtained for computer vision and cloud computing. However, a big amount of the entries corresponding with this search are only related to cloud computing. Otherwise, if look for the term "VSaaS" we only get 22 result for 2015. Therefore, we can say that video surveillance is an area with little interest in the scientific area. Now, we are going to see some examples.

The paper "The UTOPIA Video Surveillance System Based on Cloud Computing" [PARK2015] explain how the authors of the publication have created a video surveillance system prototype based on cloud designed for their smart city called UTOPIA. In this smart city, there are a lot of networked video cameras to support smart city services and they generate huge data continuously. It is usually required that the smart city should process the huge and continuous video data, real big data in real time. The use of cloud computing is used in the prototype to provide the enough computing resources to process big data successfully. This system collects big video data through scalable video streaming which smoothly processes big data traffic from large number of networked video cameras even with limited bandwidth and process the big data with MapReduce model. The objective of this paper is to present a new mechanism for the optimum cloud resources management based on dynamic scheduling policies and elasticity needs in the video surveillance sector.

Another example is the paper "Framework for a Cloud-Based Multimedia Surveillance System" [ANWAR2014], through which the authors propose a framework so that modern multimedia surveillance systems, which comprise a large number of heterogeneous sensors distributed over multiple places, may overcome their limitations in terms of scalability, resource utilization, ubiquitous access, searching, processing and storage. The paper discusses the different design choices. It further proposes a

general cloud-based surveillance system framework and analyzes it in light of the different design issues. In the following figure we can see the framework proposed. On this framework, we can distinguish the multimedia providers (cameras and other sensors), surveillance users or consumers, the system core and the service stack. In the system core the publish-subscribe broker is one of the most vital components of the system, since it is in charge of publishing and subscribing the media streams to the appropriate clients. The multimedia surveillance service directory makes the system to adopt a service oriented architecture style and hence all its functionalists are exposed as services that are accessible over the internet. The cloud manager module is responsible for managing the cloud based-operations of the proposed framework. It acts as the bridge between the users and the cloud surveillance system components. The Monitoring and metering module is responsible for performance monitoring and usage tracking of cloud virtual machine (VM) resources and provides statistics of usage and billing. Finally, the resource allocation manager manages and allocates several virtual machines (VM) resources for running the surveillance system and associated services. In the service stack the different services of the system are defined. These services include video processing service, storage service, big data analytics service, payment service, composition service, media delivery, and security and privacy services.

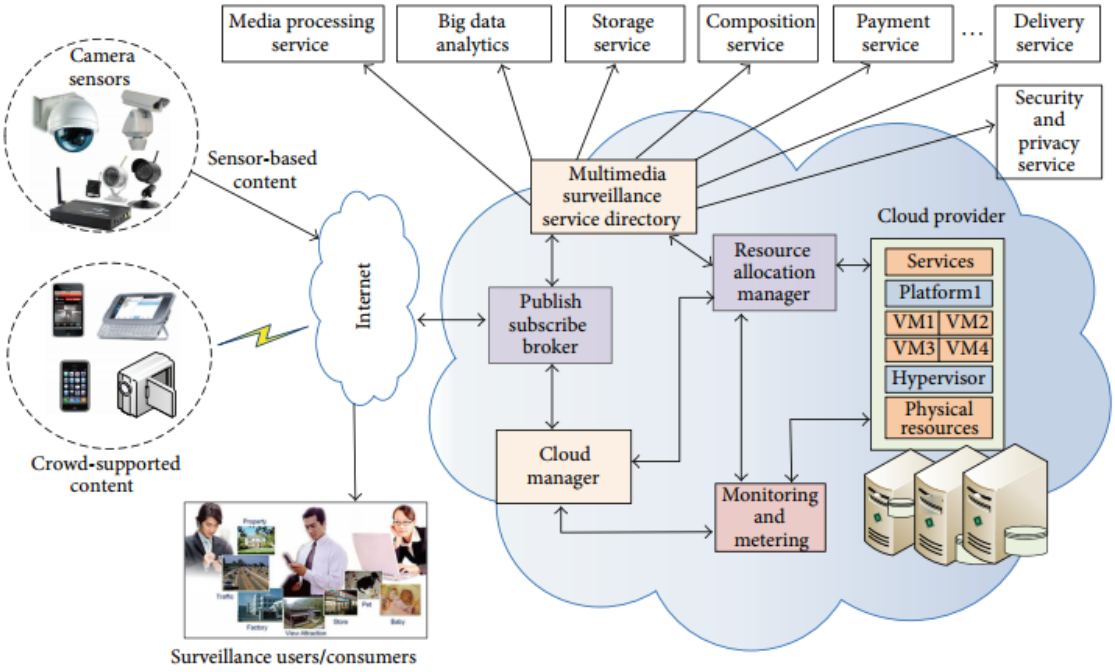


Figure 50: Framework for a Cloud-Based Multimedia Surveillance system

5.1.2 NUBOMEDIA approach beyond SotA

As we saw in the previous section we have seen that the video surveillance systems are a very active field for private companies and their business solutions. A good example which shows that video surveillance is growing sector and has a significant market, it has been the purchase of Dropcam by Google or the important number of companies that we can find in this sector. However, we have found little FOSS solutions about video surveillance system on the cloud. In the same way, scientific publications do not reach a significant volume.

As for the commercial and FOSS solutions, we have seen on the previous section that all of them have similar functionalities. These functionalities are:

- Live video
- Video Storage

- Recording schedule
- Video Content analysis software as motion detection or POS (points of sale)
- Alerts

On this context, the NUBOMEDIA platform will be one of the few FOSS platforms on the market. Futhermore, NUBOMEDIA presents more advantages over the different commercial and FOSS solutions:

- The flexibility to create applications composed by different filters in a short time. This would mean that companies could very easily make more customizable applications. This outcome is particularly relevant, since we have not found any similar capability on the open source platforms and applications found.
- Thanks to NUBOMEDIA the applications in the video surveillance system can be richer, since it will not only provide basic video analysis operations such as motion detection, but it also provides more complex video surveillance operations. In addition, the use of the Augmented Reality capabilities can be a differentiation factor with the rest of solutions.
- The capability to use filters which have not been found on the solutions studied and are able to provide a value added to surveillance systems. These filters, at the time this part of the State of the Art document was written, are perimeter intrusion detection, face detector and tracking.
- Finally, as NUBOMEDIA is an open source community the number of filters, from which the developers can take advantage, may increase considerably generating an important number of filters for this sector.

5.1.3 NUBOMEDIA outcomes

On this section we are going to see the different outcomes of NUBOMEDIA related to Video Surveillance System. At time of writing this part of the deliverable (end of November of 2015), the functionalities developed on NUBOMEDIA that can be useful in the video surveillance sector are:

- Video Streaming
- Video Storage
- Motion detector
- Object Tracker based on motion
- Perimeter intrusion detector
- Augmented reality functionalities, as overlay text on the image

During the next year, a full featured demonstrator leveraging these capabilities for video surveillance application shall be created.

5.1.4 References

[IMS] https://technology.ihs.com/

[NEST] https://nest.com/camera/meet-nest-cam/

[GOOGLE] www.google.com

[AXIS] http://www.axis.com/

[CAMMANAGER] https://www.cameramanager.com/website/

[PANASONIC] http://www.panasonic.com

[NEOVSP] http://www.neovsp.com/

[SENTIR] https://www.iveda.com/sentir/

[SMARTVUE] https://www.smartvue.com

[STRATOCAST] http://www.stratocast.com/

[UNIFORE] http://www.unifore.net/

[SECUREWERK] http://www.securewerk.com/

[YNETWORKS] http://ynetworks.in/

[ISPY] https://www.ispyconnect.com/

[ZONEMINDER] https://www.zoneminder.com/

[OPENVSS] https://code.google.com/p/openvss/

[PARK2015] Jong Won Park, et al. " The UTOPIA Video Surveillance System Based on Cloud Computing " CLOUD COMPUTING 2015: The Sixth International Conference on Cloud Computing, GRIDs, and Virtualization.

[ANWAR2014] M. Anwar Hossain "Framework for a Cloud-Based Multimedia Surveillance System", International Journal of Distributed Sensor Networks Volume 2014, Article ID 135257.

5.2 Real-time media for news reporting

5.2.1 Description of current SoTA

Traditional live news and sport coverage from the field, was done using satellite transmission. Such broadcasting obligated TV station to use complicated, heavy and unique equipment for their live field reports, such equipment required a satellite dish usually installed on top of dedicated vehicles. LiveU have revolutionized the live broadcasting world by using the cellular wide coverage offered today. LiveU “bundle” approach joins a few cellular modems to allow a wideband for live HD video and audio broadcasting. Freeing reporting team from awkward heavy equipment.

Interactive social networks provide powerful and creative way of producing amount of multimedia, often known as user generated content (UGC). UGC is created by regular people who volunteer their data most often on the web. Use of such media has been in rapid growth, one reason for its this is fact that such content is usually offered free of charge when, either the supplier wishes to be recognized or just inform others [Krumm]. Those are the reasons UGC is taking a major part in the new age of media “citizen journalism” that relies heavily on social media [gigaom website]. The increasing growth of social networks, which is the modern method of communications is connecting on

global scale, making of the web the "global village" which was promised since the was popularized.

As described social media is the new way people connect, generating large amounts of media as UGC which only destined to increase, so broadcasters and media suppliers are looking for ways to handle UGC and citizen journalism. The following few paragraphs will describe the available methods for citizen journalism and UGC.

UGC led a growing market of online video platform (OVP), which is a service allowing users to upload, convert and store their video. OVP can come as SaaS or DIY model solution as well. The vast majority of OVP solutions today use industry-standard HTTP streaming or HTTP progressive download protocols. With HTTP streaming, the de facto standard is to use adaptive streaming where multiple files of a video are created at different bit rates, but only one of these is sent to the end-user during playback, depending on available bandwidth or device CPU constraints. Such service providers are:

- Kaltura
- Ustream
- Live youtube
- Livestream on facebook
- Bambuser

OVP might be a great way for users to share their media, but it is a standalone solution and can't provide broadcasters to exchange media and stream it. Making it difficult for broadcaster to connect to their customers.

Another common for publishers to share UGC and media, is by using content delivery networks (CDN). CDN is a globally distributed network of proxy servers deployed in multiple data centers. The goal of a CDN is to serve content to end-users with high availability and high performance. CDNs serve a large fraction of the Internet content today, including web objects (text, graphics and scripts), downloadable objects (media files, software, documents), applications (e-commerce, portals), live streaming media, on-demand streaming media, and social networks.
Poplar solution are:

- Akami
- Limelight Networks

CDN however are far from answering the needs of broadcasters needs for UGC media share, they don't support HD, HQ or live broadcasting. In addition, they are p2p services. So broadcasters find it difficult to provide quality live stream using CDN.

A third option for broadcasters to connect with users (and a very unprofessional one) and exchange media is via general purpose applications. Such application, software and services are spread among users as usually method of communications, making it accessible for users to use their already installed app and use it to connect broadcasters much like they would connect their friends and family. For example, on 1st of January 2016 after a terror attack in Tel Aviv, the leading news company in Israel TV channel 2 news broadcast security videos of the attack by steaming it from what'sup.

Another application common for general users used by broadcasters is Skype, used for interviewing especially when it is not cost effective to send a reporting team.
General purpose commercial channels although widespread come with their limitations. For general use the latency, reliability and quality are not as important when

broadcasting live for millions of people at professional TV stations. Of course such channels provide services for other tens of millions of users sharing resources and services. In addition, news editors are occupied with workstation, when using unprofessional channels which do not interface with the workstation, it makes editing and broadcasting that much harder.

This need encouraged other technological related to the field, designed to take HQ media sharing into the future.

Compeit
Internet-based distribution will transform media broadcasting towards higher levels of interactivity and integration with virtual, mixed and augmented reality. It will be enabled by new web technologies and a proliferation of devices for audio, video and tangible interaction. COMPEIT creates highly interactive, personalized, shared media experiences on the Internet for feeling more present when interacting remotely with other people and enjoying media together.

Cognitus
COGNITUS will deliver innovative ultra-high definition (UHD) broadcasting technologies that allow the jointly creation of UHD media exploiting the knowledge of professional producers, the ubiquity of user generated content, and the power of interactive networked social creativity in a synergetic multimedia production approach. The project will provide a proof of concept to cement the viability of interactive UHD content production and exploitation, through use case demonstrators at large events of converging broadcast and user generated content for interactive UHD services. The envisaged demonstrators will be based on two different use cases drawn from real-life events. These use cases are in turn single examples of the fascinating and potentially unlimited new services that could be unleashed by the un-avoidable confluence of UHD broadcasting technology and smart social mobile UGC brought by COGNITUS. Building on recent technological advances, in UHD broadcasting and mobile social multimedia sharing coupled together with over fifty years of research and development in multimedia systems technology, mean that the time is now ripe for integrating research outputs towards solutions that support high-quality, user sourced, on demand media to enrich the conventional broadcasting experience. COGNITUS vision is to deliver a compelling proof of concept for the validity, effectiveness and innovative power of this integrated approach. As a consequence, over 36 months the project will demonstrate the ability to bring a new range of dedicated media services into the European broadcasting sector, adding critical value to both media and creativity sectors.

5.2.2 NUBOMEDIA approach beyond SotA

As described, current media sharing of users with broadcasters is limited especially in a UGC driven environment trying to enrich itself using citizen journalism. NUBOMEDIA will offer the infrastructure for a platform which will connect between users and broadcasters, both for reporting and UGC media. WebRTC technology will allow to promise quality streaming in HQ, HD and live. while still allowing easy communication methods using poplar wide spread technologies and keep the channel flexible allowing p2mp streaming.

Using NUBOMEDIA technology the following service could answer broadcasters needs by implanting a distribution chain:

- A user both reporters and private could connect using a WebRTC protocol to "broadcaster connector".
- The "broadcaster connector" could be controlled with an "external 3rd application".
- From the broadcaster connector the media could be disturbed to other users and the relevant broadcaster named "TV media distribution network".
- The "TV media distribution network" will be controlled using the "broadcaster application".
- From the "TV media distribution network" the broadcaster could broadcast the media via "Broadcaster streaming".

The designed service using NUBOMEDIA technology should revolutions the way broadcasters retrieve media and broadcast it to their customers. The mentioned services could replace the current methods of communication to broadcasters and fill the gap of current solutions.

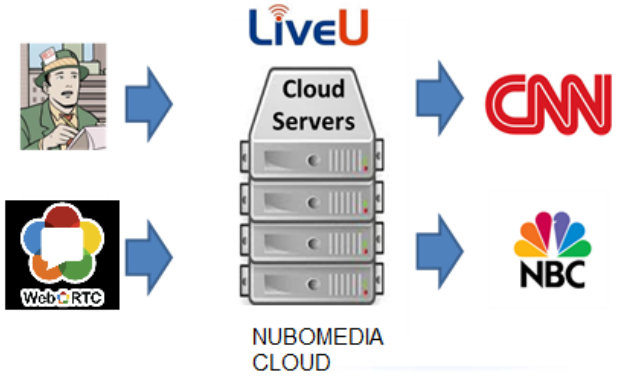
Compared to OVP, NUBOMEDIA will allow easy sharing and distribution using the platform flexibility for programmers and p2mp capability.

Compared to CDN, which can't allow live HQ streaming, there for is not suitable for broadcasters needs. NUBOMEDIA WebRTC support p2mp, live, HD and HQ streaming.

Compared to general purpose application, NUBOMEDIA using WebRTC input streams and point to multi point (P2MP) technology, answers those issue by providing end users with purpose custom software application to connect broadcasters and the editors with their media suppliers, guaranteeing:

- Quality
- Reliability
- Latency
- Suited interfacing

With a platform assigned for that with resources and services to answer their needs.
In addition, allowing to distribute the solution via news apps, making customers and professional a like a part of global "news room".



5.2.3 NUBOMEDIA outcomes

NUBOMEDIA outcomes, is the base for technology, using D2.4.2 IPTV connector such a service of connecting users and broadcasting news groups with an infrastructure for the mentioned "News Room". This virtual room will provide direct means of connecting broadcasters and media suppliers. Allowing for both professional and amateur reporters to provide quality live feeds of broadcast from every point covered with cell reception. This services are a part of the designated "BeFirst" solution will encourage citizen journalism [Goode, L., 2009,Bruns, A. (2010)], which is characteristic way of media consumption in the information era. The increasing consumption of citizen journalism will drive this NUBOMEDIA depended service to come together in the near future.

5.2.4 References

[compeit.eu] http://www.compeit.eu/

[spl.edu.gr/index.php/projects/active-projects/cognitus/] http://www.spl.edu.gr/index.php/projects/active-projects/cognitus/

[gigaom.com/2015/12/28/this-was-the-year-social-networks-turned-into-news-organizations/] https://gigaom.com/2015/12/28/this-was-the-year-social-networks-turned-into-news-organizations/

[Krumm2008] Krumm, John, Nigel Davies, and Chandra Narayanaswami. "User-generated content." *IEEE Pervasive Computing* 4 (2008): 10-11.

5.3 Real-time media on e-health environments

The healthcare industry has been attempting to find new models of care delivery as a response to the escalating cost of care, shortage of medical professionals, increased number of patients with chronic diseases and inefficiencies in the current system. To that aim, e-health (or telehealth) is emerging as one of the innovations in healthcare industry to address the challenges of access, experience and efficiencies. E-health services are delivered online through the Internet such as via e-mail, web applications and videoconferencing. Moreover, currently novel augmented reality (AR) approaches are more and more adopted in e-health by practitioners. All in all, the current description of the state-of-the-art on e-health environments is divided into three parts:

1. **Videoconferencing** used in e- health environments.
2. **Instant messaging** communications for e-health.
3. **Augmented reality** technologies on e-health communications.

The following section presents a summary of the most significant current research efforts in these areas. In addition, in order to complete this study, relevant companies, products, and technological solutions are also presented.

5.3.1 Description of current SoTA

Videoconferencing used in e-health environments

Video based telehealth is emerging as an important technology for effective and efficient collaboration between providers and patients. It facilitates face to face real time interactions with the specialists exchanging patient data resulting in prompt feedback. The feedback is crucial for overcoming time and spatial limitation for interactive tele-learning and quick knowledge sharing in order to provide better local treatment to patients [NILSEN2012].

In the case of an on-demand appointment model, the selection of the provider can be initiated by the patient or by the system. For example, a patient can click on a button on a mobile app, make a payment and the system can route the call to an available provider. While these models provide convenience, the fact that the patient has not interacted with the provider can result in a trust issue [LIU2008].

[VARGHEESE2014] proposes a model takes the validation further and provides an enhanced trust model leveraging in-session real time video real time analytics. The key benefits of that approach are improved patient satisfaction, increased trust on behalf of the patients and enhanced compliance.

To ensure patient confidentiality, national regulations exist to address control of access and transmission of medical data with which telemedicine systems may have to comply. In terms of encryption of the transmitted data, the most usual solution is to use secure socket layer (SSL) or transport layer security (TLS) with a hypertext transfer protocol secure (HTTPS) web address [SCHREIER2008]. An alternative is to use a virtual private network (VPN) which effectively brings a remote user into the secure private clinic intranet over an insecure Internet link by a secure connection [FENSLI2005]. For mobile devices, an additional preliminary safeguard before making such a secure connection, or controlling access to authorized users only beyond login details (e.g., username, password), is to verify the device used to access the data. The knowledge a mobile network operator has regarding its subscriber is used to verify the subscriber identity module (SIM) card to the clinic for data access [LIU2008b]. Once a connection is established and before data is transmitted, some additional mutual authentication between people at the remote ends is another safeguard proposed [YAN2010]. Here biometric data, such as from face recognition, was proposed to verify a clinician's identity to remote users.

Research for applications in low-bandwidth connectivity was identified [STEELE2013]. This work considers the use of new forms of health consumer and public health communication via health social networks, the use of advanced and automated health emergency notification systems including smartphone-based systems, the use of smartphone-enabled sensor technologies for home-based capturing of physiological data for at-risk patients, the use of personal health records by chronically ill and emerging mobile device–based videoconferencing capabilities.

[CLARKE2008] conducted a systematic review of the literature to critically analyze the evaluation and assessment frameworks that have been applied to telemedicine systems. It presents a framework that has been used to evaluate videoconferencing systems telemedicine applications. The conclusion of this work is that that there has been insufficient rigorous objective technical evaluation and assessment of telemedicine systems.

[KIAH2014] proposes a secure framework for video conferencing systems, using JMF (Java Media Framework) to develop secure framework and the RTP over UDP protocol

to transmit the audio and video streams. This work uses the RSA and AES algorithms to provide the required security services.

Real-time tele-echocardiography is widely used to remotely diagnose or exclude congenital heart defects. Cost effective technical implementation is realized using low-bandwidth transmission systems and lossy compression (videoconferencing) schemes. [MOORE2008] uses DICOM (Digital Imaging and Communication in Medicine) video sequences were converted to common multimedia formats, which were then, compressed using three lossy compression algorithms. We then applied a digital (multimedia) video quality metric (VQM) to determine objectively a value for degradation due to compression.

Regarding telerehabilitation, services fall into two categories: clinical assessment (the patient's functional abilities in his/ her environment) and clinical therapy. To provide both types of services remotely while interacting with the patient, the rehabilitation professionals rely on establishing a telepresence through bidirectional video and audio from videoconferencing equipment connected through a high-speed Internet connection. [HAMEL2008] proposes a telerehabilitation platform consisting of two H264 videoconferencing codecs (Tandberg 500 MXP) with integrated wide-angle view cameras and remotely controlled pan tilt zoom (PTZ) functions, local and remote computers with dedicated modular software interfaces for user-friendly control of videoconferencing connections, PTZ camera function, and external devices (i.e., tablet PC and sensors).

In [HERNANDEZ2001], the authors proposed a tele-homecare multimedia platform for videoconferencing based on standard H.320 and H.323, and a standard TV set based on integrated services digital network (ISDN) and Internet protocol (IP) to let the patients upload their physiological information to a healthcare center and to provide tele-homecare service such as consultancy over phone.

The following table summarizes some of the main companies and their products in the domain of e-Health videoconferencing:

Lifesize - https://www.lifesize.com/

Lifesize, a division of Logitech, provides high definition videoconferencing endpoints and accessories, and infrastructure services. Some of the main products of Lifesize are the following:

- Lifesize Cloud, which is a desktop/mobile video conferencing application
  - https://www.lifesize.com/en/solutions/cloud
- Lifesize Icon Series, which is video conference application
  - https://www.lifesize.com/en/products/video-conferencing-systems/icon
- Lifesize Phone HD, ut is a smartphone for your conference room—everything that you need to make web, audio and video conference calls.
  - https://www.lifesize.com/en/products/video-conferencing-accessories/phones/phone-hd

Digital Resources Inc. - http://www.digitalresources.com/

DRI is an equipment provider and systems integrator. They provide services from visualization to tele-presence, from acquisition to post production, from streaming to IPTV to broadcast. Regarding telemedicine, provides services based on three kinds of videoconferencing systems (http://www.digitalresources.com/medical.aspx):

- Standard Videoconferencing systems, for use in the doctor's office or for medical education,
- Purpose-built telemedicine carts equipped with videoconferencing capabilities, and

- Unmanned robotic systems or any number of medical devices such as a stethoscope or blood pressure monitor.
Vidyo Inc. - http://www.vidyo.com/ Vidyo provides both software-based technology and product-based visual communication solutions. The company's VidyoConferencing solutions take advantage of the H.264 standard for video compression. The portfolio of Vidyo includes the following products (http://www.vidyo.com/products/): - VidyoRouter: Media routing and video optimization - VidyoPortal: Management and call control - VidyoGateway: Interface for H.323 and SIP - VidyoReplay: Conference recording and webcasting - Vidyo Server for Microsoft Lync: Native integration with Microsoft Lync - Vidyo Server for WebRTC: Native integration with WebRTC
IVèS - http://www.ives.fr/ IVèS stands for Interactivity Video and Systems. This company develops, integrates and deploys innovative services in the fields of e-health, videoconferencing and relay centers for the deaf and hard of hearing people. Their products includes solutions for (http://www.ives.fr/index.php/en/ives/ives-the-company/11-content-uk): - Videoconference: Meeting rooms equipped with dedicated terminals with flat screens and motorized cameras can connect multiple sites together. These high-end solutions provide HD video and high quality of sound with adapted microphones and speakers. - Webconference: allows with equipment such as PC, Mac, mobile, tablet, videophone or phone to do a conference over the Internet.
Polycom - http://www.polycom.com/ Polycom is a multinational corporation that develops video, voice and content collaboration and communication technology. The company licenses H.264 video codecs, Siren codecs, session initiation protocol, native 1080p high-definition cameras and displays, native 720p and 1080p high-definition encoding/decoding, low-latency architecture and low bandwidth utilization, wideband advanced audio coding with low delay (AAC-LD), multichannel spatial audio with echo cancellation and interference filters to eliminate feedback from mobile devices, and inter-operation with legacy video conferencing. Regarding telemedicine and telehealth, the main products of Polycom are listed as follows (http://www.polycom.com/solutions/solutions-by-industry/healthcare/telemedicine-telehealth.html): - Polycom RealPresence Platform: collaborative infrastructure that enables healthcare professionals and patients to communicate. - Polycom Practitioner Cart: brings high-definition video, audio and image-sharing to medical professionals and patients no matter where they are located. - Polycom HDX 6000-View Media Center: compact solution for a patient or small group of patients to collaborate with a far end practitioner or to participate in a video call with other groups of patients at a distance. - Polycom CMA Desktop and HDX 4500: Telemedicine video call solution. The practitioner can place a video call and control the far end camera to get a good look at their remote patient. - Video Content Management: Software for secure video capture (recording and playback), as well as content management, administration, and delivery of video content. - RealPresence Mobile Polycom: video client application delivers HD-quality, audio, and content sharing to tablet users.

Table 10 Companies and products related with e-Health videoconferencing

Instant messaging communications for e-health

Smartphones support several means of communication including voice calling, video calling, text messaging, email messaging, multimedia (text, image, and video) messaging, and conferencing through the cellular phone service provider [MOSA2012].

Besides standard communications, clinical communication applications are designed to simplify communication among clinicians within a hospital.

[GAMBLE2009] describes mVisum, a communication application for cardiologist's that receive patient data on smartphone. It receives monitor data, alarms, lab results, echocardiograms, discharge notes and other reports. This app is available for iPhone, Android, Windows Mobile, and Blackberry.

Other mobile applications used for instant messages are listed in the following table (see http://www.ehealth.acrrm.org.au/technology-directory):

App name	Description
Global Meet (mobile)	Price: Free App Operating system compatibility: iOS, Blackberry Features: Call recording
LifeSize Clear Sea	Price: Free App Operating system compatibility: iOS, Android Frames per second: 30fps Features: Call recording Compatibility: QLD Health
Polycom RealPresence Mobile	Price: Free trial Hardware standards: H.239, H.263+, H.264, H.323 H.460 (Firewall traversal) Operating system compatibility: iOS, Android
Scopia Mobile	Price: Free App Operating system compatibility: iOS
Skype (mobile)	Price: Free App, free calls, free video calls Operating system compatibility: iOS, Android Features: Encryption

Table 11 Mobile applications that can be used for e-Health instant messages

Augmented reality technologies on e-health communications

Augmented Reality (AR) refers to a live view of physical real world environment whose elements are merged with augmented computer-generated images creating a mixed reality [BEHRINGER2007]. There are several domains in which AR can be used for e-health.

AR technology provides the medical field with possibilities for improvements in diagnosis and treatment of patients. This technology offers patients and health professionals with potential benefits in therapy, rehabilitation, diagnosis, and explanation. [GARCIA2014] proposed an immersive AR system to create multiple interactive virtual environments that can be used in Parkinson Disease rehabilitation programs. The main objective of this work is to develop a wearable tangible AR environment focused on providing the sense of presence required to effectively immerse patients so that they are able to perform different tasks in context-specific scenarios. The findings of this work help to evaluate the viability of using AR as an auxiliary tool for Parkinson Disease rehabilitation programs.

The ability of augmented reality to combine three-dimensional scan data with real-time patient images has made it invaluable in the medical industry. It has enormous potential in radiation oncology [NICOLAU2005].

Another hot topic related with AR is the healthcare education. A possible approach to provide learning opportunities in this area is the use AR where virtual learning experiences can be embedded in a real physical context. [ZHU2014] provides a complete integrative review of this domain.

SCOPIS - http://www.scopis.com/ SCOPIS is an example of augmenting the endoscopic video data with 3D planning information, registered with the 3D intraoperative anatomy of the patient.
Brainlab - https://www.brainlab.com/de/chirurgie-produkte/ubersicht-uber-neurochirurgie-produkte/mikroskopintegration/ Brainlab have started integrating AR supported navigation systems into optical see-through devices. In order to take full advantage of augmented microscopic views a video see-through device would be a much better platform for augmented reality based navigation software (e.g. registration, synchronization, image composition).
Arri Medical - http://www.arrimedical.com/ The company ARRI has released the first video see-through operating microscope. The digitalization of the real world with high quality stereo cameras and optics in combination with surgical applications being characterized by complex anatomical structures, availability of 3D imaging data, preoperative planning procedures can be a perfect match to develop Augmented Reality software solutions that bring a real benefit for patient treatment.
Google Glass - http://www.google.com/glass/start/ Google Glass is a headset, or optical head-mounted display, that is worn like a pair of eyeglasses. Its first version is maybe not the very best hardware solution to make the difference and convince non-believers about the potentials of AR, receiving massive criticism and legislative action due to privacy and safety concerns.
Microsoft Kinet - https://dev.windows.com/en-us/kinect Microsoft came up with the Kinect sensor, which allowed to combine tracking information with video data to create AR scenes. Among other ideas how to use this sensor, people started developing AR mirror interfaces for multiple applications, e.g. an anatomy mirror.
Oculus Rift - https://www.oculus.com/ On March 2014 Facebook bought Oculus Rift, a Virtual Reality interface. The Rift is a virtual reality head-mounted display developed by Oculus VR. Zuckerberg said: "At this point we feel we're in a position where we can start focusing on what platforms will come next to enable even more useful, entertaining and personal experiences."

Table 12 Companies and products related with e-Health AR

Finally, several research efforts have been done in the medical imaging and image-guided surgery. The work presented in [NAVRATIL2011] describes a prototype of real-time stereoscopic transmissions of robotic surgeries using a system for low latency streaming over packet networks. Medical AR systems can merge virtual images into real surgical scenes. Examples include AR microscope, projection AR system, head mounted display and image overlay [LIAO2011].

AR is still considered as a novelty in the literature. Most of the studies reported early prototypes. For that reason, there are not yet many professional solutions available on the market. Nevertheless, there are some companies that are working on create products and solutions in this domain. A good reference that summarizes these efforts is

available online on http://medicalaugmentedreality.com/. The following table summarizes several significant companies and products:

5.3.2 NUBOMEDIA approach beyond SotA

The environment of ICT applied to e-health is a multi-billion market with very relevant stakeholders in hard competition. In this context, the NUBOMEDIA approach beyond SotA is pragmatic and aims at defeating some barriers and limitations shared by most of the above mentioned solutions. In particular, these include:

- Most solutions suitable for e-health communications (i.e. providing the appropriate reliability and security guarantees) are extremely expensive and not affordable for many organizations even in developed countries.
- None of the described solutions combines together interoperable real-time media, instant messaging and augmented reality capabilities.
- None of the described solutions provides, off-the-shelf, support for multisensory data that can be combines with the audio visual streams for enriching the media information with biometric indicators.

In this context, the proposed approach is to leverage the rich media features of NUBOMEDIA for creating an open source application capable of defeating these limitations.

5.3.3 NUBOMEDIA outcomes

As specified in Task 6.2 of the, during the last year of the project we shall create a demonstrator providing professional-grade communications for e-health and incorporating augmented reality supported multisensory information. This application shall be released as open source with the aim of democratizing e-health communications. Eventually, we also plan to present the application into commercial or engineering conferences.

5.3.4 References

[NILSEN2012] Nilsen, Line Lundvoll. "Collaboration between Professionals: The Use of Videoconferencing for Delivering E-Health." *Future Internet* 4.2 (2012): 362-371.

[LUI2008] Liu, Qian, et al. "Securing telehealth applications in a Web-based e-health portal." *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*. IEEE, 2008.

[VARGHEESE2014] Vargheese, Rajesh, and Prashant Prabhudesai. "A real time provider identity verification service for a trusted telehealth video collaboration." *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2014 International Conference on*. IEEE, 2014.

[TAN2002] Tan, Joseph, Winnie Cheng, and William J. Rogers. "From telemedicine to e-health: Uncovering new frontiers of biomedical research, clinical applications & public health services delivery." *The Journal of Computer Information Systems* 42.5 (2002): 7.

[YAN2010] Yan, Hairong, et al. "Wireless sensor network based E-health system?? implementation and experimental results." *Consumer Electronics, IEEE Transactions on* 56.4 (2010): 2288-2295.

[SCHREIER2008] Schreier, Günter, et al. "A mobile-phone based teledermatology system to support self-management of patients suffering from psoriasis." *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. IEEE, 2008.

[FENSLI2005] Fensli, Rune, Einar Gunnarson, and Torstein Gundersen. "A wearable ECG-recording system for continuous arrhythmia monitoring in a wireless tele-home-care situation." *Computer-Based Medical Systems, 2005. Proceedings. 18th IEEE Symposium on*. IEEE, 2005.

[LIU2008b] Liu, Qian, et al. "Securing telehealth applications in a Web-based e-health portal." *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*. IEEE, 2008.

[STEELE2013] Steele, Robert, and Amanda Lo. "Telehealth and ubiquitous computing for bandwidth-constrained rural and remote areas." *Personal and ubiquitous computing* 17.3 (2013): 533-543.

[CLARKE2008] Clarke, Malcolm, and Chinnaya A. Thiyagarajan. "A systematic review of technical evaluation in telemedicine systems." *Telemedicine and e-health*14.2 (2008): 170-183.

[KIAH2014] Kiah, ML Mat, et al. "Design and develop a video conferencing framework for real-time telemedicine applications using secure group-based communication architecture." *Journal of medical systems* 38.10 (2014): 1-11.

[MOORE2008] Moore, Peter Thomas, et al. "Objective video quality measure for application to tele-echocardiography." *Medical & biological engineering & computing*46.8 (2008): 807-813.

[HAMEL2008] Hamel, Mathieu, Rejean Fontaine, and Patrick Boissy. "In-home telerehabilitation for geriatric patients." *Engineering in Medicine and Biology Magazine, IEEE* 27.4 (2008): 29-37.

[HERNANDEZ2001] Hernández, Alfredo, et al. "Real-time ECG transmission via Internet for nonclinical applications." *Information Technology in Biomedicine, IEEE Transactions on* 5.3 (2001): 253-257.

[MOSA2012] Mosa, Abu Saleh M., Illhoi Yoo, and Lincoln Sheets. "A systematic review of healthcare applications for smartphones." *BMC medical informatics and decision making* 12.1 (2012): 67.

[GAMBLE2009] Gamble, Kate Huvane. "Beyond phones." *Healthcare Informatics* 26 (2009): 23-5.

[BEHRINGER2007] Behringer, Reinhold, et al. *Some usability issues of augmented and mixed reality for e-health applications in the medical domain*. Springer Berlin Heidelberg, 2007.

[GARCIA2014] Garcia, A., et al. "Immersive Augmented Reality for Parkinson Disease Rehabilitation." *Virtual, Augmented Reality and Serious Games for Healthcare 1*. Springer Berlin Heidelberg, 2014. 445-469.

[NICOLAU2005] Nicolau, S. A., et al. "A complete augmented reality guidance system for liver punctures: First clinical evaluation." *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2005*. Springer Berlin Heidelberg, 2005. 539-547.

[ZHU2014] Zhu, Egui, et al. "Augmented reality in healthcare education: an integrative review." *PeerJ* 2 (2014): e469.

[NAVRATIL2011] Navratil, Jiri, et al. "Real-time stereoscopic streaming of robotic surgeries."*e-Health Networking Applications and Services (Healthcom), 2011 13th IEEE International Conference on*. IEEE, 2011.

[LIAO2011] Liao, Hongen. "3D Medical Imaging and Augmented Reality for Image-Guided Surgery." *Handbook of Augmented Reality*. Springer New York, 2011. 589-602.

5.4 Real-time media on games

Real-time gaming have some features that set it apart from the needs of other real-time communications solutions. And first of all it needs an infrastructure to provide a satisfactory gaming experience, which starts with dedicated servers, how the game design and objectives are limited by the infrastructure (bitrate and network performance among other things), when to use virtualized servers rather than bare-metal servers, how matchmaking works (the task of pairing players), or how to allocate additional capacity when demand rises.

The information collected covers two main areas: companies providing real-time media capabilities (although the term *streaming* is now prevalent) and those providing specifically streaming capabilities for gaming entertainment, since they offer some characteristics that set them apart. We use the term streaming in the gaming context in the sense of bidirectional or conversational (stateful), since in multiplayer games at least two players are matched against each other. That is, in most of the official information checked the term live streaming is understood as real-time bidirectional to sustain multiplayer interaction, which involves two channels for receiving and sending data streams.

Most if not all of this companies have a very short history, and have grown mainly by acquiring other companies of lesser size, be it because of the technology provided or because the amount of registered users actually consuming the services provided. Thanks to the increased technical capabilities of network bandwidth in several countries and devices (game consoles, as well as mobile devices and last generation TV sets), the emphasis is clearly moving towards real-time interaction.

The information collected is that which is publicly available, mainly from the companies' web sites, since we could not find any scholarly article covering this area of interest, and since changes of all kinds present a higher frequency than in other industries, with a constant reorganization of active players (ie. Companies acquiring other companies). The market is expanding and changing at a rapid pace.

Right now this sector is already well populated by many companies trying to make a profit from entertainment, a large part of which consists of real-time online gaming.

5.4.1 Description of current SoTA

Real-time multiplayer games services connect multiple players together in a single game session and transfer data messages between connected players. The typical real-time games show the following characteristics:

- Network connections are managed to create and maintain a real-time multiplayer *room*. This enables network communication between multiple players in the same game session and lets players send data directly to one another (matchmaking).
- A player uses a user interface (UI) to invite players to join a room, look for random players for auto-matching, or a combination of both.
- Participant and room state information is stored about games services servers during the lifecycle of the real-time multiplayer game (server stickiness).
- Room invitations and updates are sent to players. Notifications appear on all devices on which the player is logged in (these could be game consoles or game-enabled mobile devices).

The following concepts relate to the typical lifecycle of a real-time multiplayer game:

Room initialization

Internally, the room sets up a peer-to-peer network between participants where clients can communicate directly with each other.

Room configuration

The number of players is specified (the max number per match allowed in the room). Depending on different versions of the game, variants of the game can be used to ensure that only players who are on compatible versions are auto-matched.

Participants

When players initiate a multiplayer game, they can choose to invite specific people or have the games services automatically select other participants randomly via *auto-matching*. In real-time multiplayer games, auto-matched participants will appear as anonymous players to each other.

Connected set

As players join or leave the room, the games services actively attempt to create a network of peer-to-peer connections between all participants. This forms a *connected set* of participants in the room, and every player in the connected set is fully connected to the other players in the set.

In-game networking

The real-time multiplayer engine (most times exposed as an API) is flexible enough to allow for creating an in-game network for participants over the underlying peer-to-peer network. One of the participants acts as a 'host' to establish the authoritative game data first, and then streams this data to the other connected participants through data streaming.

Invitations

A mobile device user who is sent an invitation will see a notification on devices where they are logged in. From the invitation object the game can retrieve additional details such as the invitation creation timestamp, the invitation ID, and the player who sent the invitation.

Gameplay

Once the required number of participants for a room have been connected, the room is considered to be 'filled' and gameplay can begin.

Event notifications

As the status of the room, its participants, or connection status of the participants change, Google Play games services will send notifications to your game.

Game data interchange

The games services are used to broadcast data to participants in a room, or allow participants to exchange messages with each other while playing. This is the same principle used in games with voice activation.

Sending messages

Messages can only be sent to participants who are connected to the room.

Room closure

The game controls are responsible for *leaving* the room (that is, disconnecting the room from the servers) when a participant logs out of the game or exits the real-time portion of the game. Your game should also handle the scenario where all participants except the local player have left the room. When this happens, your game should disconnect the local player from the room immediately.

The room is considered 'closed' when all its participants have left the room. At this point, your game should shut down any game currently in progress, and make sure to save game data appropriately.

The Game Engine (GE) is the tool for the creation, development and deployment of the games. Real-time multiplayer online games have requirements beyond those needed for single-player games. In this case, among other things, the GE is responsible for the networking, streaming, threading, or memory allocation, in relation with the infrastructure available.

Real-time games require an infrastructure capable of supporting the specific loads of bidirectional streaming typical of online multiplayer gaming. We will review a number of companies offering from basic connectivity to those specialized in game environments so we could point out how NUBOMEDIA's platform will go beyond the current state of technology.

5.4.1.1 General streaming technologies / companies

There are a number of companies that are already offering media streaming capable for setting up a multiplayer infrastructure. According to public information most are still using "first-generation" technologies (some still serving content only through Flash-enabled players), and NUBOMEDIA's collection of technologies, most specifically WebRTC support is only available in two of them. The main reason for commenting on these companies here is that some have already expressed an interest in moving to

"next-generation" technologies (interactivity through WebRTC), and consequently we need to observe the current and immediate future evolution.

The companies with explicit support of WebRTC are:

- **Flashphoner**, with their product Flashphoner Web Call Server 4.0, which is a web-streaming or web-SIP solution using WebRTC audio and video streams.
- **Genband** offers real time communications solutions to connect people to each other and address the demands of users for real time contextual communications. Among Genband's products there are WebRTC capabilities in embedded communications. Genband's WebRTC solutions extend traditional communications to any device (including game consoles), on any IP Network, in any media.

The rest of these companies offer traditional communications:

- **Wowza**, with the Wowza Streaming Engine, which provides live streaming from any encoder, delivering live video and audio streams to any player, any device, over any protocol.
- **Imgtec**, with the Imagination – PowerVR Video Encoder, which includes multi-standard SD and HD video encoders. The cores encode video from raw image data, producing a compliant bit stream in one of several supported formats.
- **Bitcodin** encodes videos from a variety of input locations (e.g. HTTP or FTP servers, cloud storages, etc.) into single or multiple output qualities (representations/renditions), which are used for adaptive streaming formats such as MPEG-DASH or HLS.
- **Clearleap** is providing direct-to-consumer multiscreen streaming across multiple device and video format families, with full reporting and management tools. It has been acquired by IBM in December, 2015.
- **MTG (Modern Times Group)** is a holding of companies centered around content distribution on any device, with a recent focus on eSports, one of the largest and fastest growing online video entertainment categories, with two dedicated companies for eSports: **ESL**, an organizer of online leagues and tournaments; **Dreamhack**, runs global eSports leagues, tournaments and championships (Europe and the US).
- **Unified Streaming** has several products: a CDN, a process supporting both video on-demand (VoD) and live content; a Broadcast service, supporting many devices, from tablets, smartphones to set top Boxes, or game consoles (Xbox).
- **Envivio** has several products: Gateway Multiplexer handles live channels from any compressed broadcast source using variable bit-rate (VBR) or constant bit rate (CBR); Video Delivery interfaces with multiple content delivery networks (CDNs). It has been acquired by Ericsson in October, 2015.
- **Microsoft Azure Media Services** covers live and on-demand streaming. From 2014 their offering includes interactive game streaming platforms: Xbox Media Services (for multiplayer games) and Microsoft Azure for game development.

5.4.1.2 Game streaming platforms / companies

The following companies do offer a platform for interactive games, or are in the process of providing one during 2016. These are the ones more closely comparable to NUBOMEDIA (in the sense of real-time communications platform), but as has been mentioned before, they still use "first-generation" technologies.

- **Twitch** offers social video for gamers, with more than 55 million unique users per month to broadcast, watch and talk about video games, and the video game ecosystem: game developers, publishers, media outlets, events, user generated content, and the entire eSports scene. Twitch was acquired by Amazon on august, 2014.

  Real-time communication is currently limited to a chat channel that gamers use to communicate among themselves. It is a distributed system written in Go. In terms of scalability, it delivers hundreds of billions of messages per day to users who are watching video games via Twitch's proprietary protocols, as well as supporting IRC as a communication protocol.

  Similar to NUBOMEDIA they offer client applications for iOS, Android, Web, and in addition game consoles: Xbox ONE, Xbox 360 and PlayStation 4.

- **Valve Corporation** is an American video game developer and digital distribution company. They are behind popular titles such as Dota 2. It also developed and maintains Source on which most of its games run, and the software distribution platform Steam, which has led to the Steam Machine, a line of pre-built gaming computers running SteamOS.

  Steam is the world's largest online gaming platform. Steam provides instant access to more than 1,800 game titles and connects its 35 million active users to each other to play, share, and modify communities around Valve products. In terms of scalability, the Steam platform reached a peak of 12.5M+ users on Jan, 4th, 2016. That means users playing, trading and even buying games online concurrently.

  Valve has their own game engine, *Source*, for character animation, advanced AI, real-world physics, shader-based rendering, but no explicit capabilities as to the level of real-time support. The SDK for the engine contains many of the tools used by Valve to develop assets for their games. It comes with several command-line programs designed for special functions within the asset pipeline, as well as a few GUI-based programs designed for handling more complex functions

- **Amazon** is entering new areas of technology, and online real-time gaming is one of the most recent (from 2014 on). What this means is that virtual game consoles (Xbox from Microsoft, and others in the future) can be offered through a virtual emulator in multiscreens (PC, mobile, tablet) at the user convenience, with an HTML5-capable web browser, and using an original console controller in the same way as if using the console to play. They currently offer an Xbox cloud game console (ORBX), Red5 Media Server for video conferences, and multi-user gaming, and NVIDIA GRID GPU for multiplayer DirectX games.

- **Microsoft Media Services for Game Development** is a very recent offering from Azure (Microsoft's Cloud branch). The concept underlying the infrastructure is similar to NUBOMEDIA.

  It supports online, mobile and social games, with users playing the same game on multiple platforms and devices, players expecting to receive instant notifications when the status of their time-based game has changed, and trying to reach audiences in multiple locations around the world.

  With this objective multiplayer game servers are provided as IaaS virtual machines. Multiplayer game servers are usually based on open source or licensed frameworks, acting as the source of events for the clients (players)

connected to them and providing information about other players who have joined the same server. These frameworks require stateful interactions with clients, as well as access to local storage, provided by the infrastructure.

A final comment needs to be made about a recent trend that is gaining traction and will extend to cover multiplayer gaming in the near future, which is eSports broadcasting, and that NUBOMEDIA does not need to cover. **Azubu** and **Dailymotion Games** are providing gaming, viewing, and interactive experiences. They offer video gamers a way to live stream their game play so they can review games in front of a live audience and interact with the public. Although interesting to see how it develops, however, this is an area that lies outside of NUBOMEDIA's objectives.

5.4.2 NUBOMEDIA approach beyond SotA

Although the common basic concepts of real-time multiplayer gaming will be kept, we plan to move beyond the current SotA by making use of the integrated tech stack provided by NUBOMEDIA, that, as far as this review of the current state of this industry is concerned, is not currently offered by the companies reviewed here.

As we have seen the technologies and business of online multiplayer gaming (or cloud gaming, that is, using IaaS + PaaS) is mature with an increased number of users year after year. But the technologies in use are still "first-generation", and in contrast NUBOMEDIA will allow to create more complex games (for instance, using Augmented Reality techniques on top of real-time interactions) to enrich the player experience, increasing the engagement of players (that is, the willingness to play time and again). Engagement is the most critical factor in long-term success for games, and the unique user experiences made possible with the technologies in NUBOMEDIA will certainly move it at least one step further than what is currently available.

In this same respect, WebRTC is not being widely used or even widely promoted, so NUBOMEDIA technologies will represent a differentiating factor and will have an advantage over the current state of the industry. That said, WebRTC is still a growing field and a lot of improvement will come in future years, but the technology is viable even in the present scenario, even if more effort should be put on performance and scalability.

Another differentiating factor, not to be neglected, is the amount of physical infrastructure currently needed. For instance, Twitch reports several worldwide-deployed PoPs (Points of Presence), making operating costs a critical factor in profit making. NUBOMEDIA's streamlined and focused platform will decrease physical needs. Although it is difficult to quantify at this point in time, a reduction of 10-20% of comparable costs with established companies will mean a significant step forward beyond the current state.

None of companies reviewed follows a FOSS strategy. They may offer for-free SDKs to ease integration, but, unlike NUBOMEDIA, overall FOSS is out of the picture. This is pretty clear when comparing it to Microsoft Azure for Game Development, which of all the companies or offerings reviewed is the one that more closely resembles NUBOMEDIA's platform, and no WebRTC official support yet.

5.4.3 NUBOMEDIA outcomes

ZED's main outcomes from NUBOMEDIA revolve around testing and validating new ways of creating multiplayer games. Games with more "layers" than are commonplace in the current typical games. Successful games nowadays are necessarily multiplayer in concept and practice. They are more engaging and consequently generate more revenue than single-user games. ZED will create a social game as a demonstrator of the stack of technologies composing NUBOMEDIA. The philosophy underlying this demonstrator is centered on real-time interaction and will be a testbed for experimentation of some game concepts. It will be too ambitious to say that this could create the *killer app* that WebRTC needs to become the de facto standard.

ZED plans to include NUBOMEDIA's resulting technologies (either all of them or just part) into a future generation of games, even exploring new categories where traditionally ZED had no previous products. For instance, highly interactive game-based educational material will be made possible in a much easier way using the technologies of the project. ZED has agreements with a number of game publishers and distributors and will propose creating these new games using NUBOMEDIA's outcomes, thereby simplifying the full game development and deployment lifecycle, where infrastructure and platform and tooling traditionally take a lot of time and effort. This will result in savings in time, testing and overall quality.

5.4.4 References

[FLASHPHONER.COM] http://flashphoner.com/

[GENBAND.COM] https://www.genband.com/

[WOWZA.COM] http://www.wowza.com/

[IMGTEC.COM] https://imgtec.com/

[BITCODIN.COM] https://www.bitcodin.com/

[CLEARLEAP.COM] http://clearleap.com/solutions/

[MTG.COM] http://www.mtg.com/our-world/what-we-do/our-digital-products/

[UNIFIEDSREAMING.COM] http://www.unified-streaming.com/

[ENVIVIO.COM] (http://www.envivio.com/

[AZUREMICROSOFT.COM] https://azure.microsoft.com/en-us/services/media-services/live-on-demand/

[TWITCH.TV] http://www.twitch.tv/

http://engineering.twitch.tv/

[VALVESOFTWARE.COM] http://www.valvesoftware.com/

Steam http://store.steampowered.com/

Source https://developer.valvesoftware.com/wiki/SDK_Docs

[AMAZON.COM] https://aws.amazon.com/

[MICROSOFT.COM] https://customers.microsoft.com

Azure Xbox https://customers.microsoft.com/Pages/CustomerStory.aspx?recid=13468

MS Game Development https://msdn.microsoft.com/en-us/magazine/dn532200.aspx

[AZUBU.TV] http://www.azubu.tv

[DAILYMOTION.COM] http://games.dailymotion.com/

5.5 Real-time media for social TV

At its very beginning, TV was natively "social": when a TV screen was too expensive to be considered "everyone's technology", it was quite normal to have shared TV screens in public places or even to meet all together with the neighborhood to watch TV in the evening.

During The 90s, we saw a deep growth in telecommunications. First of all the number of mobile phones suddenly grew with the advent of GSM, while being this the "brand new communication way", the fall in terms of prices and appeal of fixed communications led to the first "fixed flat rates" offers. In that context we first saw the some cross-interactions like voting via SMS or Phone call, or having SMS texts shown on top of the screen, aiming at saying hello to relatives or friends.

In general, "Social TV" is the term used to describe the current integration of social media interaction with television programming. Social television has sought to recapture those early days of television, when families gathered in their homes to share the experience of watching television together. Over the past several years, online social media communities such as message boards, Twitter, and Facebook have become the new virtual water cooler for today's tech-savvy television viewers. With the proliferation of social media applications and Smartphone technology, social interaction around television programming can now be shared amongst millions of viewers simultaneously. Twitter and other 2 social media platforms have "become an integral outlet for TV viewers who look to express themselves while watching broadcasts of their favorite television programs." This "backchannel" of communication during TV shows has also led to the resurgence of people's interest in watching live shows.

It has been reported that people watch more live TV to both avoid spoilers and to communicate with other viewers. This is in contrast to the trend only a few years ago when people started using DVR machines to watch shows at their own pace. While the amount of data generated by users in the context of TV is enormous and ripe for data mining and business analytics, the problem is that the raw data are a noisy stream of consciousness. Such a limitation is probably one of the reasons why current TV applications does not actually exploit the possibilities given by Social Data at their top. As it was mentioned above, many users use Twitter in order to express their opinion about what they are seeing live on TV, most of the TV shows, particularly talk shows and talent shows, tend to use twitter in the same way they used SMS during the 90s, by simply superimposing tweets on top of the screen, this is of course much more simple than with SMS.

On the opposite side, mobile and tablet applications related to the TV world use social network mostly for user authentication and as a way to review TV programs, leading to applications called "Social EPG". Some TV shows also used specific social network applications for voting, but it's not a widely diffused approach.

5.5.1 NUBOMEDIA approach beyond SotA

NUBOMEDIA allows to build brand new services related to the Social World. As it was mentioned above, the Social world somehow re-discovered the pleasure of sharing the TV experience, but as the current applications just allow discussion among live viewers, NUBOMEDIA could even do more. It could allow to mix video-conferencing with live TV events, allowing final users to keep in contact while watching TV shows or Live events such as soccer games, or even movies, taking care both to put in communication the different peers and to keep the synchronization so that viewers see things happen at the same time.

Besides this, the possibility to superimpose information in a simple way allows to simply show or monitor social information directly on a specific video stream, given that we are able to have a parallel service monitoring social networks and extracting information in a smart and simple way.

The solution provided by NUBOMEDIA will allow to monitor some statistical and mood information about a specific video stream: such an application could be useful for talk show, speeches and other TV shows in general needing to monitor and show the "social mood" and "social stats" related to a specific broadcasting event.

5.5.2 NUBOMEDIA outcomes

As specified in NUBOMEDIA DoW, during the last year of the project we shall create a social TV demonstrator that will combine media content with a social monitoring tool called SDA (Social Data Aggregator).

5.5.3 References

[BENTON2013] Benton, Adrian and Hill, Shawndra. "The Spoiler Effect: Designing Social TV Content That Promotes Ongoing WOM".