D7.5

Version	1.0
Author	URJC
Dissemination	PU
Date	25/01/2017
Status	Final

D7.5: NUBOMEDIA WWW community sites

Project acronym:	NUBOMEDIA
Project title:	NUBOMEDIA: an elastic Platform as a Service (PaaS) cloud for interactive social multimedia
Project duration:	2014-02-01 to 2017-01-31
Project type:	STREP
Project reference:	610576
Project web page:	http://www.nubomedia.eu
Work package	WP7
WP leader	Claudio Venezia
Deliverable nature:	Report
Lead editor:	Luis Lopez
Planned delivery date	01/2017
Actual delivery date	25/01/2017
Keywords	NUBOMEDIA, FOSS, Community

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

Contributors:

Luis López (URJC)
Boni García (URJC)

Internal Reviewer(s):

Teofilo Redondo (ZED)

Version History

Version	Date	Authors	Comments
0.1	05/10/2016	Luis Lopez	Initial version
0.2	11/11/2016	Boni García	Update sections 4 and 5.2. Include section 7
0.3	12/12/2016	Boni García	Update community website section
1.0	04/12/2017	Luis Lopez	Include community feedback information

Table of contents

List of Figures:

Acronyms and abbreviations:

API	Application Programming Interface
AR	Augmented Reality
IMS	IP Multimedia Subsystem
IoT	Internet of Things
KMS	Kurento Media Server
RTC	Real-Time Communications
RTP	Real-time Transport Protocol
SCTP	Stream Control Transmission Protocol
VCA	Video Content Analysis
WebRTC	Web Real Time Communications

1 Executive summary

This document contains a description on the NUBOMEDIA community website. The NUBOMEDIA Community is an open source software community and an associated ecosystem that is built with the aim of maintaining, evolving and disseminating the results of the NUBOMEDIA research project. This document also presents the dissemination strategic guidelines of the community as well as a brief description of the software repositories associated to it.

2 Introduction

Open Source Software strategies can be useful for multiplying the impact of software products and for disseminating them more effectively and rapidly among the different stakeholders. There are many examples of very successful projects that have succeeded in the generation of profitable business models and for stimulating the emergence of novel industries, standards and services. Among them, we can cite the Apache ecosystem, the Asterisk VoIP stack, the MySQL DDBB manager, the Linux kernel, the Mozilla Foundation browsers, etc. Our objective is making NUBOMEDIA one of such successful projects by becoming the FOSS reference on the area of interactive multimedia communications.

This objective is not simple given the relevant hype that WebRTC solutions has experienced during the last few years, which has made relevant stakeholders and thousands of developers to enter the play. In addition, due to its own nature, NUBOMEDIA is a heterogeneous collection of technology which provides a bunch of capabilities that can be useful both as a whole, and as separated solutions for specific problems.

Anyhow, gaining the attention of wide developer audiences will strengthen significantly the dissemination capabilities of the project and the business impact it may generate. For this, two essential ingredients are a must:

- The existence of a flexible FOSS license guaranteeing flexibility, freedom, fairness and different types of business models around the software.
- The release of useful software artifacts under that license suitable for helping developers
- The management of the community activities through the appropriate governance rules and the communication of its activities through a website and several social channels and policies.

This deliverable is concentrated on the last item. For this, we define governance mechanisms for the community taking into consideration the different sensibilities and needs of the involved organizations as well as the possibility of the different involved projects to evolve with independence. We also define the main communication mechanisms for the community that are designed around a community website and several social channels. These shall be leveraged for giving cohesion to NUBOMEDIA as a whole. The following sections are devoted to introducing all of them.

3 The NUBOMEDIA Community governance

3.1 The NUBOMEDIA project and the NUBOMEDIA community

Our objective in this deliverable is to specify how the activities carried out into the NUBOMEDIA Project (a research project funded by the European Commission) can be leveraged for creating the NUBOMEDIA Community (an open source software community). This community should contribute to enhance the exploitation and dissemination of results, so that the project gives the investment back to the society, and to provide a mechanism for the generated technologies to survive beyond the project.

Achieving this objective requires finding a balance between two different forces: on one side, the needs of the project for which the community is instrumental; on the other, the needs of the community for which attracting developers is critical.

Starting with the former, the needs of the project in relation to the community are clear and can be enumerated in the following way:

- The creation and consolidation of the community must take place in the context of duration of the project and through the resources that the project makes available to it, being both limited and bounded. This decreases the flexibility for organizing the community as an independent entity.
- The community initial software assets shall be the ones associated with the project partners' activities. In the short and medium term, this means that it is the partners who need to invest the appropriate resources for maintaining such assets and for communicating them to potentially interested developers. Due to this, the community governance model needs to be compatible with their different interests, sensibilities, roadmaps and expectations. This shall generate some degree of fragmentation which is unavoidable.
- Due to contractual reasons, the community's initial objectives need to be aligned with the ones of the project. The objectives of the project were designed several years ago and may not be fully aligned with current state-of-the-art in relation to market status and market needs. This decreases the flexibility of the community to take the most appropriate shape for adapting to current trends and market status.

On the other hand, for being successful the community requires a number of ingredients, which include the following:

- To have very high flexibility for adapting in an agile way to developer needs and market trends, even if this means dropping support for some technological pieces for enhancing others.
- To offer a unified and coherent vision, discourse and organizational structure susceptible of being understood by developers in a seamless and simple way.
- To be expansive trying to maximize impact by incorporating all kinds of organizations, individuals, business models and support mechanisms susceptible of helping in its mid-term survival, even if they are not fully aligned or compatible with all partners' or project's objectives.

Following this discussion, and trying to achieve a balance among these opposite forces, we propose to create the NUBOMEDIA community basing in a governance model built upon a number of design decisions.

The first of them deals with the community as an entity. Many communities out there are articulated around some kind of legal entity that sometimes takes the form of a company but most frequently takes the form of a foundation. The community governance model is then driven by the specific regulations of such legal entity. There are also many other relevant communities that are not managed through any kind of entity. They consist just of an informal association among different stakeholders which may include organizations and individual developers, who cooperate pursuing their own objectives but who share a common interest in the specific artifacts settled under the community umbrella. These communities tend to be more organic and they self-organize following a number of implicit or explicit rules that are accepted by all the members and that are enforced by the community as a whole through one governing rule: meritocracy [RAYMOND1999]. In the context of FOSS software, meritocracy is the direct consequence of the value of contribution in a FOSS project. Anything else in FOSS, such as democracy, ultimately lowers contributions to the benefit of uninvolved people and to the law of numbers, masses and manipulation. That's why FOSS has never been about democracy but about meritocracy, even if it practices limited democracy among its contributors. But that notion belongs to community processes [STDF2011].

In relation to this specific topic, we chose the second option: the organic and self-organized community. We select this model for a number of reasons:

- The legal entity model is not compatible with the financial and time constrains we have in the NUBOMEDIA project. Growing such legal entity would require relevant investments for which no specific funding has been reserved. In addition, coming to an agreement on formal governance rules to be accepted by the NUBOMEDIA Project legal representatives would require huge time and effort investments that might generate relevant impediments for objectives of both the community and the project.
- There are no clear evidences on the advantages of supporting the community through a legal entity. Moreover, many authors defend that they are counterproductive because they increase the bureaucratic effort, which tends to discourage developers from participating [OMAHONY2005]. Although having the support of a legal entity may enforce the emergence of clear driving authorities, many relevant FOSS communities just self-organize and generate a notion of authority through very simple democratic mechanisms [OMAHONY2007]. Hence, the effort of articulating the NUBOMEDIA community through a legal entity is not clearly justified.

The fact that the community is self-organized does not mean that it is anarchic. For guaranteeing its balance between flexibility and efficacy, and for enforcing its alignment with the objectives of the project, we shall define the democratic rules driving to conflict resolution and decision making in the community as part of the community governance model.

The second decision to be taken is in relation of the community structure. Communities exhibiting a simple and unified structure tend to be more attractive for developers and, very particularly, to newcomers. This is due to their ability to communicate a simpler and direct message in relation to the capabilities of the distributed software. However, such uniformity is not possible in projects involving heterogeneous technologies where the involved stakeholders pursue divergent objectives through independent roadmaps. Due to this, many communities organize as an association or aggregation of projects, which are managed independently, but which share a common vision and governance model [MOCKUS2002]. These communities might result more obscure and difficult to understand for newcomers, but they have a very relevant advantage from a practical perspective: flexibility. Letting projects to organize independently sharing just a common vision makes possible for them to maintain custom roadmaps and policies which enhance competition enabling more successful initiatives to shine and filtering unsuccessful ones through natural-selection mechanisms [SCACCHI2006].

Following this, in NUBOMEDIA we took the decision of organizing following the structure of an informal project association. The following arguments support it:

- Due to contractual reasons, the community must integrate all the partners of the NUBOMEDIA project. However, not all the partners have the same degree of commitment towards the NUBOMEDIA open source strategy. For some partners the NUBOMEDIA generated software is a critical asset that needs to be maintained and evolved in an agile and rigorous way, while for others the community just represent an experiment. Due to this, designing the community as a single and uniform project fully sharing governance and decision making would be a guarantee of failure. More involved partners do not require the agreement of less involved ones.
- Due to contractual reasons, the copyright holders for each of the artifacts generated on the NUBOMEDIA project remain under the ownership of their creators. Due to this, organizing the community as a single project with a single repository policy would require all partners to have access to commit changes on others' code. This might generate relevant legal problems in relation to copyright ownership. In addition, this would make extremely complex to maintain control of code evolution, forcing developers to invest relevant resources on coordination problems instead of on developing. Again, this situation would automatically drive towards a community failure.
- The worldwide open source software ecosystem is extremely competitive and, as other technological markets [SCHILLING2002], exhibits a tendency towards a winner-take-all situation. This means that creating successful open source projects may require in many cases destroying competing alternatives. Evidently, this requires a number of ingredients such as high commitment with the project from its stakeholders, agility, innovation capability, technological excellence, etc. The NUBOMEDIA project is a very heterogeneous software stack where some of its components may satisfy those ingredients but others may not. Due to this, linking the future of components having more possibilities with the ones with less is clearly the less appropriate strategy.

As a result of these discussions, we can conclude that the proposed strategy lays down onto a NUBOMEDIA open source software community organized organically through a meritocratic and flexible democratic process. This community as the structure of an association among projects where the NUBOMEDIA community itself is an umbrella providing coherence towards the objective of maintaining and evolving the NUBOMEDIA platform. This coherence is basically achieved through a number of communication channels, which include the community website and the community social channels.

3.2 Structure and governance of open source software communities

For performing the appropriate design of the structure and governance of the NUBOMEDIA Community we have inspired on how some successful open source communities are organized. There is a relevant variability on this aspect, but most authors accept that there are two main organizational paradigms that are commonly used: the Cathedral and the Bazaar [RAYMOND1999]. Communities basing on a Cathedral paradigm are typically governed complying with a set of explicit rules that define roles, positions and bodies. These assume responsibilities and coordinate among each other. Typically, Cathedral-based communities are compared to companies as both comply with well-defined organizational schemes. O the other hand, Bazaar-based communities tend to organize through implicit rules and through discussions that take place in an open way through public means without the mediation of any kind of body.

Just for illustration, Figure 1 depicts the organization structure of two Bazaar-like communities: GStreamer and PostgreSQL. In both of them, a number of individuals (that we identify as “Members”) hold control of the evolution of the technology by determining, through consensus mechanism based on public discussions, how the source code evolve and who has the right of pushing commits. These types of communities might identify a number of informal representatives that hold some kind of role (e.g. Regional Contact in the case of PostgreSQL)

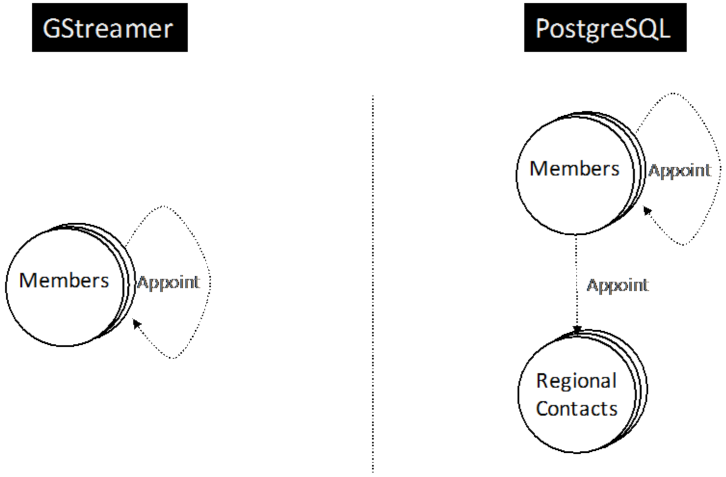


Figure 1. Schema depicting how unstructured open source communities such as GStreamer (left) and PostgreSQL (right) are organized. In the case of GStreamer, a set of individuals (that we call Members) hold write rights to the source code repository and control through informal public consensus mechanisms how the code evolves and what new members are accepted. In the case of PostgreSQL this basic structure is reinforced introducing Regional Contacts that perform communication and evangelization of the community into specific geographic and cultural areas.

Also for illustration, Figure 2, Figure 3 and Figure 4 show some other more structured communities that define explicit governance mechanisms and management bodies for enforcing them. Perhaps the most popular of such communities is the one driven by the Apache Software Foundation that, as depicted in Figure 2, is based on a company-line simplified structure where members hold the right of electing the main organizational bodies which, in turn, appoint the appropriate resources suitable for driving the community in a day-to-day basis. The Apache Software Foundation governance is typically considered a good example of a simple organizational structure suitable for providing the required mechanisms enabling the appropriate management and evolution of a complex software ecosystem.

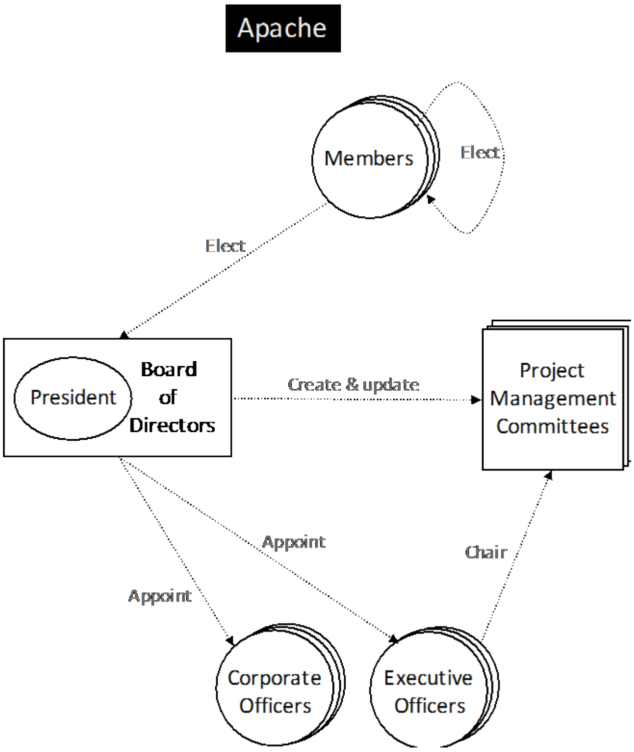


Figure 2. Community structure and governance used by the Apache Software Foundation. This community is organized as a company where the individual Members act as stockholders. These Members have control to elect new members and to elect the Board of Directors that manages the community. The Board of Directors appoints officers that assume the day-to-day executive responsibilities. The Board of Directors also determines which projects software integrate into the community, where each of such projects must be supported by a Project Management Committee. All in all, the Apache Software Foundation have a nice equilibrium between complexity, organizational rigor and agility.

Another popular community is the one managed by the OpenStack Foundation, whose structure is sketched in Figure 3. As it can be observed, the complexity of the structure is increased by differentiating among multiple types of members that may have different obligations and rights of vote. This community also introduces explicit bodies assuming specific responsibilities such as the Users Committee, in charge of representing users' voice; or the Legal Affairs Committee, that advices the Board of Directors in all legal aspects including intellectual property ones.

To conclude, Figure 4, we depict the governance structure of the FIWARE community, that is a relevant example given that, as in the case of NUBOMEDIA, this is a community emerging out of a research project. As it can be observed, the FIWARE community is strongly inspired in OpenStack but introduces novel roles and bodies adapted to FIWARE specificities. In particular, FIWARE technologies are organized in terms of Generic Enablers (GE) that are independent projects providing one or several capabilities. GEs, in turn, organize in Chapters that group interrelated GEs. FIWARE considers GEs and Chapters at the time of specifying the management roles and bodies of the community. In addition, further bodies are included with the objective of providing voice to further community actors such as the Ecosystem Support Committee or the External Associations. All in all, the FIWARE community is an example of a complex governance associated to a highly-regulated community

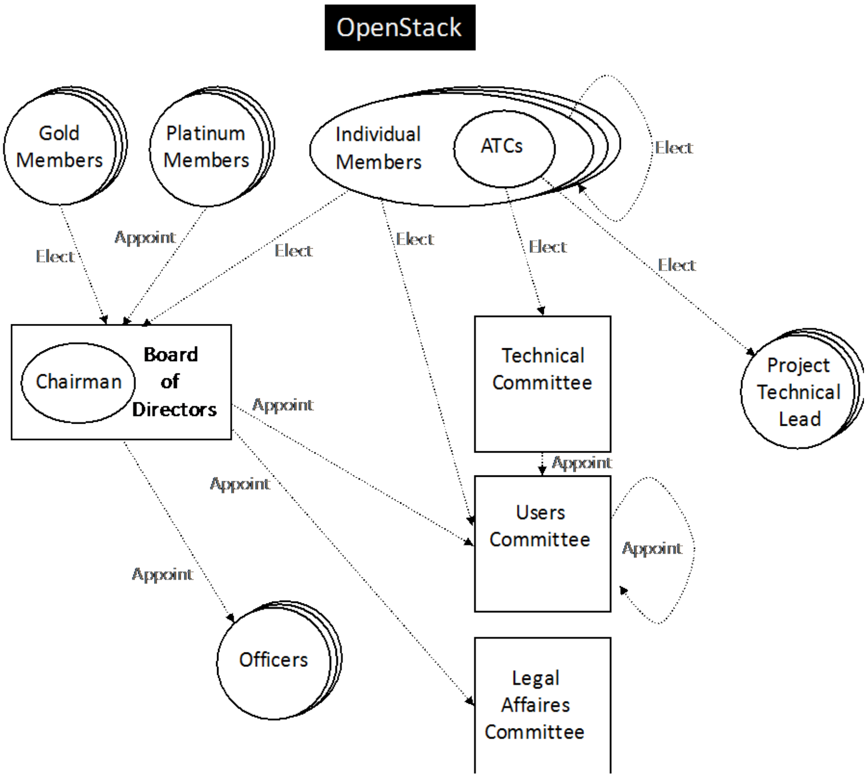


Figure 3. Community structure and governance used by the OpenStack Foundation. This community is organized basing on corporate governance principles. The community distinguishes between Individual Members (i.e. developers) and institutional Members (i.e. Gold and Platinum) providing different voting rights for them. The community also identifies Active Technical Contributors (ATC) as Individual Members that commit code to the project in a regular basis. All in all, members elect the Board of Directors, that manages non-technical aspects of the project, and the Technical Committee, that manages the technical ones. A number of additional bodies are also set up in place with different objectives (e.g. Legal Affairs Committee, Users Committee, etc.) The day-to-day technical responsibilities are assumed by Project Technical Leads, that hold latest responsibility of conde evolution into the different projects comprising the community. The OpenStack Foundation has higher complexity than the Apache software foundation approaching significantly the structure preferred by software companies but maintaining simplicity.

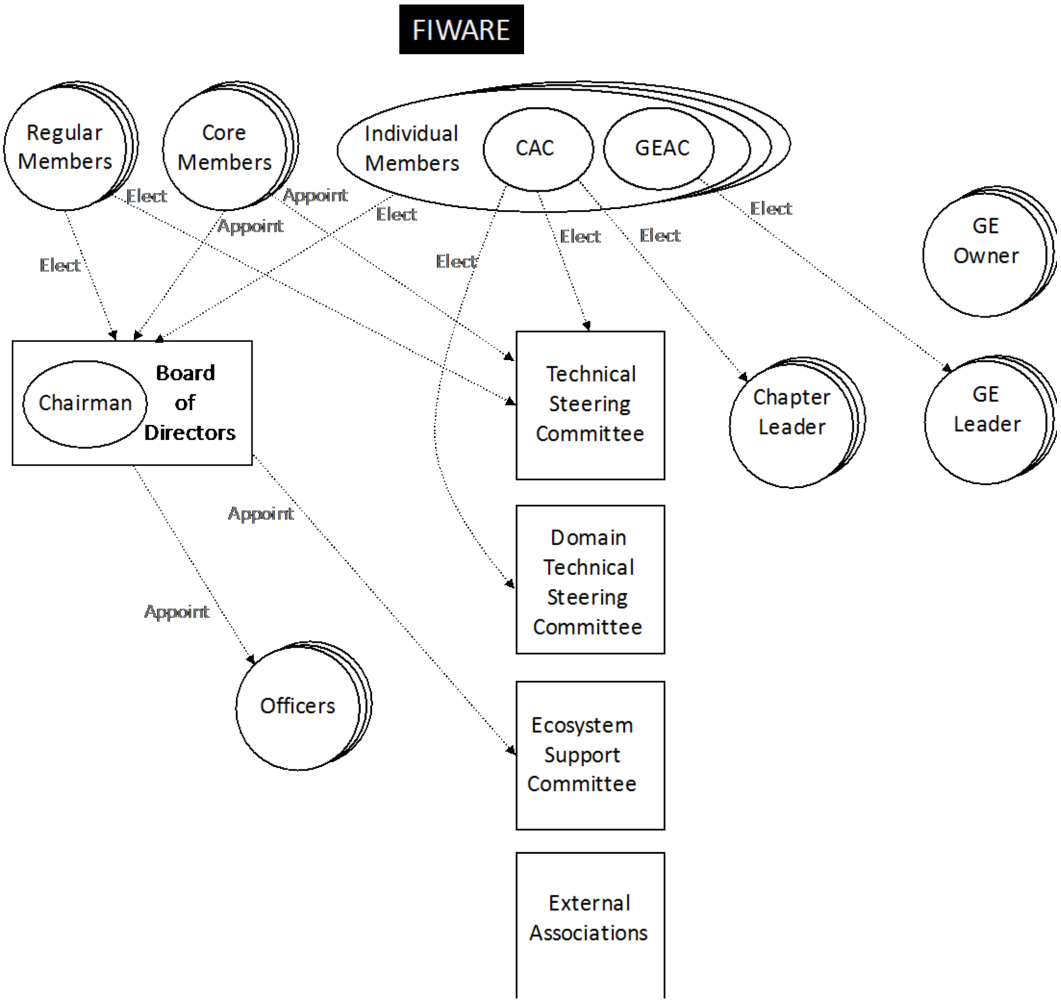


Figure 4. Community structure and governance of the FIWARE community. The FIWARE community is a relevant example as it shares with NUBOMEDIA the fact of being an open source software community emerging out of a research project. This community is strongly inspired in the OpenStack Foundation governance model inhering from it the existence of different types of members (e.g. individual, institutional, etc.) that hold different types of voting rights. The community defines a number of support bodies that include the Technical Steering Committee, a number of Domain Technical Steering Committees, an Ecosystem Support Committee, etc. From a technological perspective, the community is organized in Chapters that group independent software projects called Generic Enablers each. For each Chapter and Generic Enabler, the Active Contributors elect the respective leaders. The FIWARE community is an example of a heavily regulated community as it has a quite complex governance structure.

3.3 The NUBOMEDIA Community governance structure

One of the main objectives of this document is to define what is the structure of the NUBOMEDIA Community and how it is governed. Based on the discussions above, we first need to choose whether our community is to be based on a Bazaar or a Cathedral philosophy. In this case, the decision is towards a Cathedral-like model due to a number of reasons:

- The NUBOMEDIA Community has not grown organically but out of a well-organized and highly regulated research project. Hence, maintaining a tight organization is the natural choice.
- The NUBOMEDIA Community comprises heterogeneous software components that require tight coordination. Letting such coordination to happen basing on organic implicit organizations basing on public consensus is probably not the most efficient solution.

- The NUBOMEDIA Community involves software artifacts of great value for their respective owners and that are used extensively beyond the project. Due to this, the owner partners prefer to base on well-established rules for managing their evolution.

In spite of this decision, the NUBOMEDIA partners believe that for an open source software community to be successful its governance must provide the appropriate flexibility and openness and must be simple enough for being understandable and transparent for newcomers. Due to this, and due also to the moderate volume of software artifacts that need to be managed, we decide to avoid very complex organizational structures as the one of FIWARE. On the other hand, we rather prefer a simpler model closer to Apache Software Foundation's one. Basing on this, we have created a NUBOMEDIA Community structure that is sketched in Figure 5, that is based on the following roles and bodies:

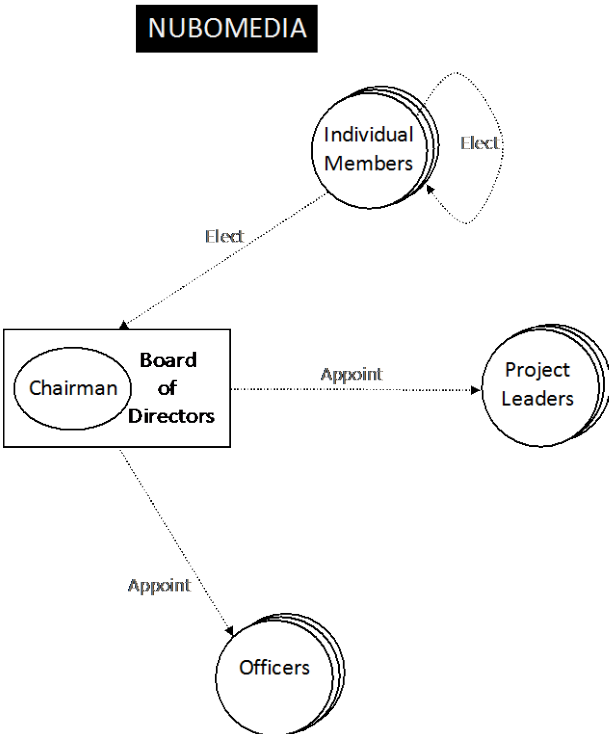


Figure 5. The NUBOMEDIA Community Governance Structure. This structure is inspired in the one of the Apache Software Foundation but it is further simplified by replacing the Project Management Committees with Project Leads that, as in OpenStack, manage independently the day-to-day technical aspects of every of the comprising projects.

- Individual Members: as in the case of Apache, they are individuals that hold the voting rights of the community. They elect the Board of Directors and also manage membership through a democratic process.
- Board of Directors: it holds executive responsibilities of the community. It can appoint a number of Officers for supporting it. It also appoints the Project Leads that manage the day-to-day technical activities of the software projects.
- Project Leaders: Inspiring in how OpenStack works, these are experts that hold the responsibility of maintaining and evolving a specific software project belonging to the community.

The next section develops this governance structure into a formal set of rules.

3.4 The NUBOMEDIA Community governance

This section specifies the NUBOMEDIA Community governance model that has been approved by the NUBOMEDIA project consortium and which shall be used for launching the community in an ordered way.

3.4.1 Definitions

- The NUBOMEDIA Community (or Community in the following) refers to a collection of Assets and a group of Individual Members who act complying with the NUBOMEDIA Governance Model.
- The NUBOMEDIA Governance Model refers to the rules specified in this document.
- The NUBOMEDIA Platform (or Platform in the following) refers to the collection of software artifacts providing the functionalities described in the NUBOMEDIA Project.
- The NUBOMEDIA Project refers to a research project funded by the European Commission under Framework Programme 7 with reference number GA 610576.
- The NUBOMEDIA website, also called NUBOMEDIA Community website or Community website, refers to the website reachable at http://www.nubomedia.eu

3.4.2 The NUBOMEDIA Community objectives

The NUBOMEDIA Community is a Free Open Source Software (FOSS) community having the objective of promoting, enhancing, maintaining and evolving, in the long term, the software generated as a result of the NUBOMEDIA Project. We call the NUBOMEDIA Platform to this software and to the rest of enhancements and extensions that may be created after the above-mentioned project ends.

The NUBOMEDIA Community shall be based on a membership mechanism through which members shall commit to the following:

- To maintain and evolve the software assets of the Community
- To comply with the Community governance model and best practices
- To promote the Community and work constructively for defending the NUBOMEDIA good name.

3.4.3 The NUBOMEDIA Community: Associated Projects and Members

For attaining its objectives, the NUBOMEDIA community is organized around two concepts: Projects and Individual Members.

Projects are structured sets of software artifacts providing demonstrable value to the NUBOMEDIA Platform. Projects typically consist of open source software projects which have been designed or are suitable for providing a capability to the NUBOMEDIA Platform. Hence, from a software perspective, the NUBOMEDIA Community is a collection of Projects whose composition is determined through the NUBOMEDIA Decision Procedures. These Projects maintain some independence to evolve and to define their own governance mechanism and roadmaps as long as the NUBOMEDIA Governance rules specified here are satisfied. The community website shall maintain an inventory of Projects publicly accessible.

Individual Members are individuals who support the NUBOMEDIA Community and who govern it following the NUBOMEDIA Decision Procedures.

3.4.3.1 NUBOMEDIA Community Individual Members

Governance of the NUBOMEDIA Community is managed through Individual Members, who have a right of vote in agreement with the Community Decision Procedures. Membership is free of charge, although it requires the satisfaction of a number of requirements:

- An Individual Members is a person who participates on her own.
- Any Individual Member must demonstrate its commitment towards the Community by providing relevant contributions to it in terms of monetary donations, infrastructure donations, effort or code.
- Any Individual Member must commit to comply with the Community governance model and practices
- Any Individual Member must commit to promote the Community and work constructively for defending the NUBOMEDIA good name.
- Anyone can become an individual member as long as the Community Decision Procedures for the integration of new Individual Members of the Community, as specified below, are satisfied.

Each of the Individual Members shall have one vote on the Community Decision Procedures

3.4.4 Community Governance Structure

The Community shall be governed by the following bodies and positions:

Board of Directors

The Board of Directors is elected by the Individual Members following the Community Decision Procedures specified below. The Board of Directors comprises 5 individuals, that might be Individual Members or not. Once constituted, the Board of Directors shall appoint a Chairman that shall preside the Board of Directors and shall have be appointed to representing it in all fora. The Board of Directors may appoint a number of Executive Officers that shall assume, when appropriate, specific responsibilities on the community such as marketing, communication, operations, release planning, etc.

Project Lead

Any Projects integrated as part of the Community shall have a Project Lead, who shall be supervising the technical activities on the specific Project. The Project Lead shall be appointed by the Board of Directors and shall report to it.

3.4.5 Community Decision Procedures

The following decision procedures shall be used for the Community governance

Decision procedure for the integration of new Individual Members into the community

The integration of a new Individual Member into the community shall be performed through vote following this procedure:

- Candidatures to become an Individual Member shall be endorsed by at least one actual Individual Member.

- Candidatures to become an Individual Member shall provide the information justifying the opportunity of integrating that member into the community. That information should contain any relevant data helping the actual members to evaluate the suitability of the new member and their contributions to the Community. The information shall be made available for evaluation to the rest of members, at least, during 1 month previous to the vote.
- All actual Individual Members shall have right to vote through any available mechanism providing reasonable reliability guarantees, including e-mail or WWW voting services.
- The new Individual Member shall be accepted whenever the number of positive votes is strictly over half of the number of actual Individual Members at the time of vote .
- Upon acceptance, the new Individual Member shall become an actual member to all effects.

Decision procedure for electing the Board of Directors

The Board of Directors shall be elected among all actual Individual Members in a vote taking place once per year:

- Candidatures to become part of the Board of Directors shall be presented providing any relevant information about candidates to the Individual Members. That information shall be available during at least 1 month previous to the vote.
- The Individual Members shall vote among the available candidates through any available mechanism providing reasonable reliability guarantees, including e-mail or WWW voting services.
- The Board of Directors shall be constituted by the 5 candidates with more votes.

Decision procedure for modifying the Community Governance Model

The Board of Directors may propose modifications to the Community Governance Model specified in this document. Their approval shall require the following procedure to be executed:

- Suggested modifications shall be presented together with any additional information considered relevant for their evaluation. That information shall be available during at least 1 month previous to the vote.
- The Individual Members shall vote through any available mechanism providing reasonable reliability guarantees, including e-mail or WWW voting services.
- The proposed modifications shall be approved whenever the number of favorable votes is equal or bigger than two thirds of the number of actual Individual Members at the time of vote.

Decision procedure for the removal of Community Members

Any Individual Member may explicitly refuse to its membership, in which case, it shall be cancelled automatically. In addition, any Individual Member may propose the removal of other members or representatives from the list of actual Individual Members. For this, the following procedure shall be applied:

- The identity of the specific member to be removed, as well as the justification for removal, shall be provided to the Indiidual Members. The specific appointed member may answer to that information with further data defending its membership. That information shall be available during at least 1 month previous to the vote.

- The Individual Members shall vote through any available mechanism providing reasonable reliability guarantees, including e-mail or WWW voting services.
- The member or representative shall be removed whenever the number of votes supporting the removal is equal or bigger than two thirds of the number of actual Individual Members

Decision procedure for the integration of new Projects into the community

Any member may propose the integration of a new Project into the Community. For this, the following procedure shall be applied:

- The proposer shall present all the relevant information demonstrating the interest of the new Project for the Community as well as the identity of the initial Project Lead for it. That information shall be available during, at least, 1 month previous to the vote.
- The Individual Members shall vote through any available mechanism providing reasonable reliability guarantees, including e-mail or WWW voting services.
- The project shall be accepted as an new Community Project whenever the number of positive votes is strictly over half of the number of the actual Individual Members.
- In case of acceptance, the Project shall have 1 month of time for adapting to the NUBOMEDIA Community Rules in Relation to the Release of Software Artifacts. After that period, the Project shall become part of the community with full rights.

Decision procedure for the removal of Projects from the community

Any member may propose the removal of an Project from the Community. For this, the following procedure shall be applied:

- The proposer shall present all the relevant information justifying the need for removing the Project. That information shall be available during, at least, 1 month previous to the vote.
- The Individual Members shall vote through any available mechanism providing reasonable reliability guarantees, including e-mail or WWW voting services.
- The project shall be removed as a Community Project whenever the number of votes supporting the removal is strictly over half of the number of actual Individual Members.

3.4.6 Community Governance Principles

This section contains a set of informal ethical and behavioral principles that should govern Community decisions in order to maintain an open vision and attitude towards users, committers and members.

Meritocracy

The basic principle through which authority emerges is meritocracy (literally government by merit). Individual Members demonstrating higher knowledge and commitment with the project should have priority at the time of assigning responsibilities.

Communication

The main communication channels among community members is e-mail. E-mails shall be organized through subjects as “virtual discussion rooms” where conversations shall take place asynchronously. This shall simplify the collaboration in geographically distributed groups. Some projects may also use more synchronous mechanisms such as instant messaging or teleconferences.

Documentation

Each project is responsible for its own project communication mechanisms. This includes the project website, which is optional, and the project documentation, which is mandatory.

Philosophy

While there is no official philosophical doctrine, the following principles are considered as core beliefs of the Community:

- Collaboration: this is the way to work and to solve problems.
- Commercial-friendly standard license: LPGL and Apache-like licenses are preferred.
- Consistently high quality software: test, test and test.
- Respectful, honest, technical-based interactions: leave your ego at home.
- Faithful implementation of standards: standards are not mandatory, but are preferred.

Operation

All Projects are composed of volunteers and none of their members are paid directly by the NUBOMEDIA Community. Committers and Individual Members may be paid to work on the projects by any interested individual or organization, but never by the NUBOMEDIA Community itself. The NUBOMEDIA Community, however, may contract out other different services including infrastructure administration, accounting or communication.

Balancing confidentiality and public discussion

We endeavor to conduct as much discussion in public as possible. This encourages openness, provides a public record and stimulates the broader community. However, sometimes internal communication is necessary. In this case, members should use their common sense for maintaining reasonable etiquette and do not disclose sensitive information without the permission of the affected people or entities.

3.4.7 Community Rules in Relation to the Release of Software Artifacts

This section contains a set of rules and recommendations for the release of software artifacts in the context of the NUBOMEDIA Community. All Associated Projects MUST comply with the rules and SHOULD take into consideration the recommendations and comply with them whenever it is possible.

3.4.7.1 Community Rules

Distribution of source code

- With independence on the existence of multiple distributions mechanisms, all associated project MUST make available the source code in GitHub (http://www.github.com)
- Each GitHub repository README.md file MUST include the following:
  - The following sentence: “This project is part of NUBOMEDIA”
  - A link to www.nubomedia.eu
  - Licensing and contributions policies
  - A short description of the repository structure

- The released artifacts must comply with the following:
    - They MUST contain the source code of any NUBOMEDIA-related feature.
    - They SHOULD contain source code of examples and tutorials
    - They MUST contain up-to-date documentation including:
        - Installation guide
        - Developers guide
        - Reference documentation
    - Documentation sources MUST be under version control in GitHub
    - Up-to-date versions of the documentation MUST be published in https://readthedocs.org/

Distribution of open binaries

- All Java API releases MUST be distributed through maven central
- All JavaScript API releases MUST be distributed through bower (browser) and npm (node)
- All Debian or Ubuntu artifacts MUST be distributed through a public debian repository

Community support

- All Associated Projects MUST maintain a public issue tracker where bugs and problems can be reported
- All Associated Projects MUST maintain a public mailing list where questions from potential end users shall be answered
- All Associated Projects MUST nominate a responsible person that could be contacted in case of problems.

Continuous integration rules

- All Associated Projects SHOULD create and maintain a public CI system validating
    - That all artifacts can be built and deployed accordingly to the installation guide
    - That artifacts' functionalities are validated with, at least, one integration test each.

3.4.7.2 Community launch

As the Community is the result of the NUBOMEDIA Project, the initial community structure shall be created from the project decision making organisms. The official launch of the community to the public with full independence of the project shall also be provided by such organisms and shall never be after the last day of the project (i.e. Jan, 31st 2017)

4 The NUBOMEDIA Community structure

As specified above, the initial community structure, including members and associated projects, shall emerge from the project. In relation to this, the project has decided to host the NUBOMEDIA source code in GitHub, under the organization **nubomedia** (https://github.com/nubomedia/). This organization has a hierarchical structure that comprises 3 levels of repositories. The first level (https://github.com/nubomedia/nubomedia) is the top-level repository containing the whole structure of the NUBOMEDIA projects. The second level of repositories is depicted in the following table:

Name	URL	Description
nubomedia-controller	https://github.com/nubomedia/nubomedia-controller	Code providing the cloud management features
nubomedia-admin	https://github.com/nubomedia/nubomedia-admin	Code related to administration and management
nubomedia-demonstrators	https://github.com/nubomedia/nubomedia-demonstrators	Code related to the NUBOMEDIA FOSS demonstrators
nubomedia-documentation	https://github.com/nubomedia/nubomedia-documentation	Artifacts related to high-level documentation
nubomedia-media-server	https://github.com/nubomedia/nubomedia-media-server	Artifacts related to Kurento Media Server and its modules
nubomedia-sdks	https://github.com/nubomedia/nubomedia-sdks	Artifacts related to NUBOMEDIA SDKs, including Web, Android and IOS
nubomedia-connectors	https://github.com/nubomedia/nubomedia-connectors	Artifacts related to NUBOMEDIA FOSS connectors

Each of these second-level repositories contains individual repositories with the different parts of NUBOMEDIA. All these third-level repositories are described in the next sections.

4.1 nubomedia-controller

The nubomedia-controller contains all the three-level repositories related with the cloud management features of NUBOMEDIA. These repositories are: NFVO, generic-vnfm, marketplace, nubomedia-msvnfm, and nubomedia-paas. The following tables summarizes the main information for each individual repository.

Name	NFVO	**Owner**	TUB	**License**	Apache 2.0
URL	https://github.com/openbaton/NFVO				
Description	Open Baton NFVO is an open source project providing a reference implementation of the NFVO based on the ETSI NFV MANO specification				

Name	generic-vnfm	**Owner**	TUB	**License**	Apache 2.0
URL	https://github.com/openbaton/generic-vnfm				
Description	The Generic VNFManager is an implementation of a VNF Manager following the ETSI MANO specifications				

Name	marketplace	**Owner**	TUB	**License**	Apache 2.0
URL	https://github.com/nubomedia/marketplace				
Description	The NUBOMEDIA Marketplace provides an easy way for managing and downloading NUBOMEDIA applications from a central point				

Name	nubomedia-msvnfm	**Owner**	TUB	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-msvnfm				
Description	Media Server VNF Manager (EMM) implemented in Java using the spring.io framework				

Name	nubomedia-paas	**Owner**	TUB/USV	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-paas				
Description	Manager for PaaS Platform that exposes REST API to allow NUBOMEDIA users to build and deploy applications on the NUBOMEDIA Platform				

4.2 nubomedia-admin

The nubomedia-controller comprises all the three-level repositories related to NUBOMEDIA administration and management. These repositories are: kms-monitoring-java, nubomedia-autonomous-installer, nubomedia-nova-docker, nubomedia-benchmark and nubomedia-network-benchmark. The following tables summarizes the main information for each individual repository.

Name	kms-monitoring-java	**Owner**		**License**	LGPL 2.1
URL	https://github.com/nubomedia/kms-monitoring-java				
Description	Examples showing how to use KMS v6.0 monitoring API				

Name	nubomedia-autonomous-installer	**Owner**		**License**	LGPL 2.1
URL	https://github.com/nubomedia/nubomedia-autonomous-installer				
Description	The NUBOMEDIA Autonomous Installer is able to install the NUBOMEDIA platform into an IaaS environment which has already installed a PaaS system				

Name	nubomedia-nova-docker	**Owner**		**License**	LGPL 2.1
URL	https://github.com/nubomedia/nubomedia-nova-docker				
Description	NUBOMEDIA Nova-docker plugin for OpenStack improvements				

Name	nubomedia-benchmark	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-benchmark				
Description	Benchmark application aimed to assess the performance of NUBOMEDIA				

Name	nubomedia-network-benchmark	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-network-benchmark				
Description	Benchmark application aimed to assess the network performance of NUBOMEDIA				

4.3 nubomedia-demonstrators

The nubomedia-demonstrators comprises all the three-level repositories related with the demonstrators developed within the NUBOMEDIA project. These repositories are: arfilterdemopaas, msdatademopaasgraph, nubomedia-art, nubomedia-design, nubomedia-magic-mirror, nubomedia-ouatservice, nubomedia-ouatservice-dev, and nubomedia-ouatservice-pre. The following tables summarizes the main information for each individual repository.

Name	arfilterdemopaas	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/arfilterdemopaas				
Description	Demonstrator using the ARModule filter (augmented reality library which contains marker detector filter utilizing ALVAR)				

Name	msdatademopaasgraph	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/msdatademopaasgraph				
Description	Demonstrator using the MultisensoryData filter (library that implements multi-domain augmented reality filter providing 2D graphics visualization service)				

Name	nubomedia-art	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-art				
Description	Art of Tales, an interactive comic where the user interacts with the story through AR, VCA, and so on.				

Name	nubomedia-design	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-design				
Description	Design of Tales, the Zed's demonstrator				

Name	nubomedia-magic-mirror	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-magic-mirror				
Description	NUBOMEDIA Tutorial: WebRTC in loopback with FaceOverlay filter				

Name	nubomedia-ouatservice	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-ouatservice				
Description	ouatservice demonstrator				

Name	nubomedia-ouatservice-dev	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-ouatservice-dev				
Description	ouatservice demonstrator (development environment)				

Name	nubomedia-ouatservice-pre	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-ouatservice-pre				
Description	ouatservice demonstrator (production environment)				

4.4 nubomedia-documentation

The nubomedia-documentation comprises all the three-level repositories for documentation about NUBOMEDIA. These repositories are: developer-guidelines, doc-kurento-room, doc-kurento-tree, doc-kurento-repository, doc-kurento-jsonrpc, nubomedia-repository-tutorial, nubomedia-room-tutorial, nubomedia-cdn-tutorial, Doc-Nubomedia-MultisensoryDataModule, Doc-Nubomedia-VTT-AR-Demo, Doc-

Nubomedia-Armodule, doc-kurento-room-client-android, doc-kurento-tree-client-android, doc-webrtcpeer-android, nubomedia-graphedit-doc, and kurento-ios-documentation. The following tables summarizes the main information for each individual repository.

Name	developer-guidelines	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/developer-guidelines				
Description	Documentation project with the development guidelines for NUBOMEDIA				

Name	doc-kurento-room	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-kurento-room				
Description	Kurento Room framework documentation				

Name	doc-kurento-tree	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-kurento-tree				
Description	Kurento Tree documentation				

Name	doc-kurento-repository	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-kurento-repository				
Description	Kurento Repository documentation				

Name	doc-kurento-jsonrpc	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-kurento-jsonrpc				
Description	Kurento JSON-RPC documentation				

Name	nubomedia-repository-tutorial	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-repository-tutorial				
Description	NUBOMEDIA Tutorial: WebRTC with recording capabilities to repository				

Name	nubomedia-room-tutorial	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-room-tutorial				
Description	NUBOMEDIA Tutorial: N to N video communication (room)				

Name	nubomedia-cdn-tutorial	**Owner**	FRAUNHOFER	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-cdn-tutorial				
Description	NUBOMEDIA Tutorial: example application of the usage of the CDN Connector				

Name	Doc-Nubomedia-MultisensoryDataModule	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/Doc-Nubomedia-MultisensoryDataModule				
Description	Documentation of the MultisensoryData filter				

Name	Doc-Nubomedia-VTT-AR-Demo	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/Doc-Nubomedia-Armodule				
Description	Documentation of the demo which uses the ARModule filter				

Name	Doc-Nubomedia-Armodule	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/Doc-Nubomedia-Armodule				
Description	Documentation of the ARModule filter				

Name	doc-kurento-room-client-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-kurento-room-client-android/				
Description	Documentation for Kurento Room API for Android				

Name	doc-kurento-tree-client-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-kurento-tree-client-android/				
Description	Documentation for Kurento Tree API for Android				

Name	doc-webrtcpeer-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/doc-webrtcpeer-android/				
Description	Documentation for WebRTCPeer API for Android				

Name	nubomedia-graphedit-doc	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-graphedit-doc/				
Description	Nubomedia graphical modelling and code generation tools documentation				

Name	kurento-ios-documentation	**Owner**	TI	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-ios-documentation/				
Description	Documentation for Kurento Toolbox for iOS				

4.5 nubomedia-media-server

The nubomedia-media-server comprises all the three-level repositories related with Kurento Media Server and its modules. These repositories are: armodule, gst-libav, gst-plugins-bad, gst-plugins-base, gst-plugins-good, gst-plugins-ugly, gstreamer, jsoncpp, kms-chroma, kms-cmake-utils, kms-core, kms-crowddetector, kms-datachannelexample, kms-elements, kms-filters, kms-jsonrpc, kms-opencv-plugin-sample, kms-platedetector, kms-plugin-sample, kms-pointerdetector, kurento-media-server, kurento-module-creator, libnice, libs3, msdata, openwebrtc-gst-plugins, and usrsctp. The following tables summarizes the main information for each individual repository.

Name	armodule	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/armodule				

Description	ARModule contains marker detector filter utilizing ALVAR augmented reality library ie ALVAR markers can be detected from the video image

Name	gst-libav	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/gst-libav				
Description	This module contains a GStreamer plugin for FFmpeg, as well as an internal copy of the FFmpeg source code				

Name	gst-plugins-bad	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/gst-plugins-bad				
Description	GStreamer 1.7.x development series (bad)				

Name	gst-plugins-base	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/gst-plugins-base				
Description	GStreamer 1.7.x development series (base)				

Name	gst-plugins-good	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/gst-plugins-good				
Description	GStreamer 1.7.x development series (good)				

Name	gst-plugins-ugly	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/gst-plugins-ugly				
Description	GStreamer 1.7.x development series (ugly)				

Name	gstreamer	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/gstreamer				
Description	GStreamer 1.7.x development series				

Name	jsoncpp	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/jsoncpp				
Description	JsonCpp is a C++ library that allows manipulating JSON values, including serialization and deserialization to and from strings				

Name	kms-chroma	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-chroma				
Description	The kms-chroma project contains a chroma filter element for the Kurento Media Server				

Name	kms-cmake-utils	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-cmake-utils				
Description	Common CMake utilities for Kurento projects				

Name	kms-core	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-core				
Description	The kms-core project contains core elements needed for the Kurento Media Server				

Name	kms-crowddetector	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-crowddetector				

Description	The kms-crowddetector project contains a crowd detector filter element for the Kurento Media Server

Name	kms-datachannelexample	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-datachannelexample				
Description	This project contains an example module implementing WebRTC datachannels for Kurento Media Server				

Name	kms-elements	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-elements				
Description	The kms-elements project contains elements needed for the Kurento Media Server				

Name	kms-filters	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-filters				
Description	The kms-filters project contains filter elements for the Kurento Media Server				

Name	kms-jsonrpc	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-jsonrpc				
Description	JsonRPC protocol implementation in C++				

Name	kms-opencv-plugin-sample	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-opencv-plugin-sample				
Description	Sample project for KMS plugins based on OpenCV				

Name	kms-platedetector	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-platedetector				
Description	Plate detector filter element for Kurento Media Server				

Name	kms-plugin-sample	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-plugin-sample				
Description	Sample Module for Kurento Media Server				

Name	kms-pointerdetector	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kms-pointerdetector				
Description	Pointer detector filter element for Kurento Media Server				

Name	kurento-media-server	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-media-server				
Description	Kurento Media Server is the Kurento's core element. It is responsible for media transmission, processing, loading and recording				

Name	kurento-module-creator	**Owner**	NAEVATEC	**License**	Apache 2.0

URL	https://github.com/nubomedia/kurento-module-creator
Description	Processor for the Remote Object Model of the Kurento media server

Name	libnice	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/libnice				
Description	GLib ICE library				

Name	libs3	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/libs3				
Description	C Library API for Amazon S3				

Name	msdata	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/msdata				
Description	Implementation for utilizing MultisensoryDataFilterModule ie AR data can be received from other modules and visualized				

Name	openwebrtc-gst-plugins	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/openwebrtc-gst-plugins				
Description	OpenWebRTC specific GStreamer plugins				

Name	usrsctp	**Owner**	NAEVATEC	**License**	Apache 2.0
URL	https://github.com/nubomedia/usrsctp				
Description	userland SCTP stack supporting FreeBSD, Linux, Mac OS X and Windows				

4.6 nubomedia-sdks

The nubomedia-sdks comprises all the three-level repositories related with NUBOMEDIA SDKs, including Web, Android and IOS SDKs. These repositories are: Kurento-iOS, jsonrpc-ws-android, kurento-java, kurento-maven-plugin, kurento-room, kurento-room-client-android, kurento-tree, kurento-tree-client-android, kurento-tutorial-java, kurento-tutorial-js, kurento-tutorial-node, kurento-utils-js, nubo-test, nubo-test-tree, nubomedia-graphedit, nubomedia-media-client, nubomedia-repository-client, promisecallback, utilities-android, and webrtcpeer-android. The following tables summarizes the main information for each individual repository.

Name	Kurento-iOS	**Owner**	TI	**License**	Apache 2.0
URL	https://github.com/nubomedia/Kurento-iOS				
Description	It provides a set of components for the native development of WebRTC applications with Kurento				

Name	jsonrpc-ws-android	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/jsonrpc-ws-android				
Description	Android library for sending JSON-RPC messages over a WebSocket connection				

Name	kurento-java	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-java				
Description	Project that include all Java code from Kurento				

Name	kurento-maven-plugin	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-maven-plugin				
Description	Maven Plugin for Kurento				

Name	kurento-room	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-room				
Description	This project introduces the Room API, designed for the development of conferencing applications with Kurento				

Name	kurento-room-client-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-room-client-android				
Description	Kurento Room API for Android				

Name	kurento-tree	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-tree				
Description	Kurento Tree implements a broadcasting service using WebRTC technology				

Name	kurento-tree-client-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-tree-client-android				
Description	Kurento Tree API for Android				

Name	kurento-tutorial-java	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-tutorial-java				
Description	Examples on usage of the Kurento Java Client				

Name	kurento-tutorial-js	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-tutorial-js				
Description	Kurento Client JavaScript demos				

Name	kurento-tutorial-node	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-tutorial-node				
Description	Examples on usage of the Kurento Node.js Client				

Name	kurento-utils-js	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/kurento-utils-js				
Description	Set of reusable components that have been found useful during the development of the WebRTC applications with Kurento				

Name	nubo-test	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubo-test				
Description	Test software for Android Room Client and WebRTC client.				

Name	nubo-test-tree	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubo-test-tree				

Description	Test application demonstrating use of Kurento Tree API in Android

Name	nubomedia-graphedit	**Owner**	ZED	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-graphedit				
Description	NUBOMEDIA graph editor				

Name	nubomedia-media-client	**Owner**	TUB	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-media-client				
Description	Library needed to deploy Java application on the NUBOMEDIA Cloud Platform				

Name	nubomedia-repository-client	**Owner**	TUB	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-repository-client				
Description	Library needed to use the Kurento repository within the NUBOMEDIA Cloud Platform				

Name	promisecallback	**Owner**	URJC	**License**	Apache 2.0
URL	https://github.com/nubomedia/promisecallback				
Description	Set a callback to be called by a promise				

Name	utilities-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/utilities-android				
Description	Android utility library meant to be used in other VTT's NUBOMEDIA related Android projects				

Name	webrtcpeer-android	**Owner**	VTT	**License**	Apache 2.0
URL	https://github.com/nubomedia/webrtcpeer-android				
Description	Android library for creating WebRTC connections				

4.7 nubomedia-connectors

The nubomedia-connectors comprises all the three-level repositories related with NUBOMEDIA Connectors. These repositories are: nubomedia-cdn-connector and nubomedia-ims-connector. The following tables summarizes the main information for each individual repository

Name	nubomedia-cdn-connector	**Owner**	FRAUNHOFER	**License**	Apache 2.0
URL	https://github.com/nubomedia/nubomedia-cdn-connector				
Description	NUBOMEDIA Connectors enable interoperability of the NUBOMEDIA platfrom with external media service providers				

Name	nubomedia-ims-connector	**Owner**	FRAUNHOFER	**License**	LGPL 2.1
URL	https://github.com/nubomedia/nubomedia-ims-connector				
Description	The IMS Connector is a light weight library which can be used on an IMS User Agent or as a protocol stack on an Application Server				

5 The NUBOMEDIA Community website

As specified above, the website is the main communication channel of the NUBOMEDIA Community. Following the strategic guidelines described above, the community website and the project website shall be created in convergence, so that at the beginning, the project-related contents have more relevance but slowly, the community related-content gains importance on it until it absorbs it all.

To this aim, we have created a community site which consists of a brief description and a number of hyperlinks to the associated projects. As specified in NUBOMEDIA Project Task 7.2, these contents shall be evolved until a full-featured and mature community page is in place. This community has been integrated as part of the NUBOMEDIA project website in a specific section devoted to "Community":

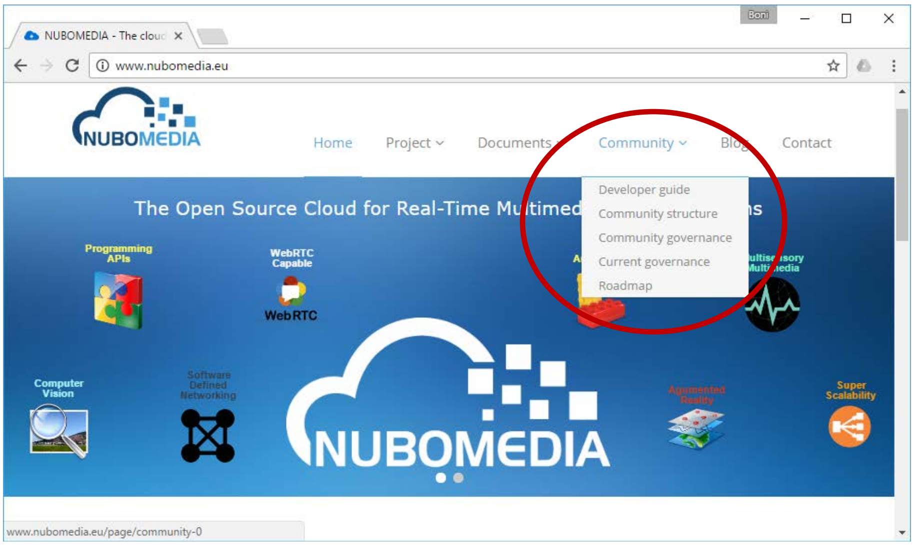


Figure 6 NUBOMEDIA community web

As shown in Figure 6, the NUBOMEDIA Community site contains five sections:

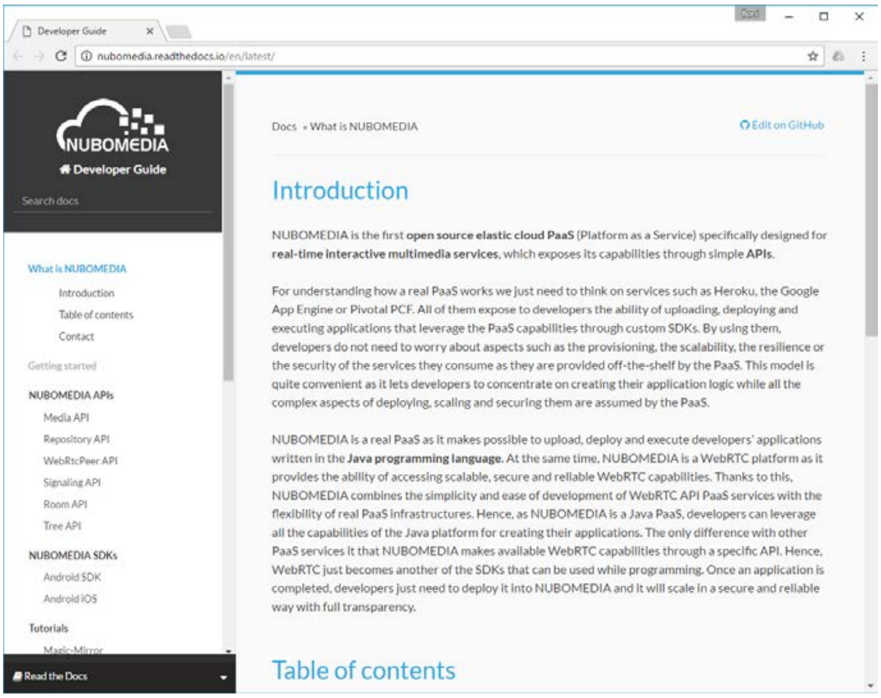
- Developer guide. This entry is a link to the general NUBOMEDIA developer guide documentation.
- Community structure. This section contains a description of the NUBOMEDIA ecosystem and source code structure.
- Community governance. This section contains the NUBOMEDIA Community governance model that has been approved by the project consortium.
- Current governance.This section contains the list of Community members, Board of directors (BoD), and Project leads of the NUBOMEDIA project.
- Roadmap. This section contains a list of new features to be implemented in the future.

The following sub-sections provides more details about these sections.

5.1 Developer guide

The developer guide documentation contains the general information needed for developers to create and deploy applications using NUBOMEDIA. This documentation has been created using Mkdocs and it is hosted on readthedocs.org in the following link (see Figure 7):

- http://nubomedia.readthedocs.io/



Introduction

NUBOMEDIA is the first **open source elastic cloud PaaS** (Platform as a Service) specifically designed for **real-time interactive multimedia services**, which exposes its capabilities through simple **APIs**.

For understanding how a real PaaS works we just need to think on services such as Heroku, the Google App Engine or Pivotal PCF. All of them expose to developers the ability of uploading, deploying and executing applications that leverage the PaaS capabilities through custom SDKs. By using them, developers do not need to worry about aspects such as the provisioning, the scalability, the resilience or the security of the services they consume as they are provided off-the-shelf by the PaaS. This model is quite convenient as it lets developers to concentrate on creating their application logic while all the complex aspects of deploying, scaling and securing them are assumed by the PaaS.

NUBOMEDIA is a real PaaS as it makes possible to upload, deploy and execute developers' applications written in the **Java programming language.** At the same time, NUBOMEDIA is a WebRTC platform as it provides the ability of accessing scalable, secure and reliable WebRTC capabilities. Thanks to this, NUBOMEDIA combines the simplicity and ease of development of WebRTC API PaaS services with the flexibility of real PaaS infrastructures. Hence, as NUBOMEDIA is a Java PaaS, developers can leverage all the capabilities of the Java platform for creating their applications. The only difference with other PaaS services it that NUBOMEDIA makes available WebRTC capabilities through a specific API. Hence, WebRTC just becomes another of the SDKs that can be used while programming. Once an application is completed, developers just need to deploy it into NUBOMEDIA and it will scale in a secure and reliable way with full transparency.

Table of contents

Figure 7 NUBOMEDIA developer guide

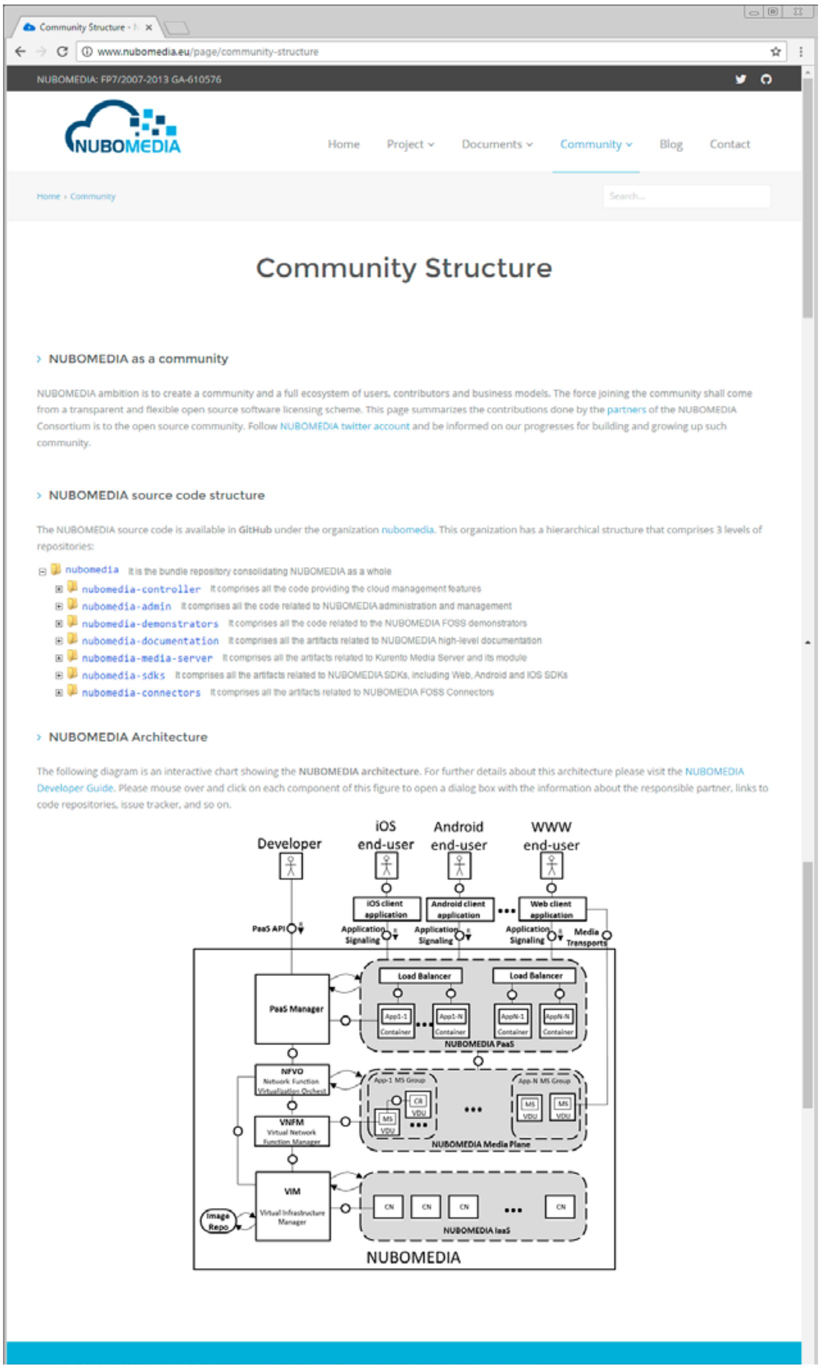
The NUBOMEDIA developer guide is composed by the following sections:

- What is NUBOMEDIA. Brief introduction, table of contents, and contact information about NUBOMEDIA.
- Getting started
- NUBOMEDIA APIs. High-level documentation of the NUBOMEDIA APIs, namely: Media API, Repository API, WebRtcPeer API, Signaling API, Room API, and Tree API.
- NUBOMEDIA SDKs. High-level documentation of the NUBOMEDIA SDKs, namely: Android SDK and Android iOS.
- Tutorials. This section contains example application showing how to use the NUBOMEDIA APIs to build different types of applications with advanced media capabilities, such as the “Magic-Mirror” (example of applications that uses the Media API), “Repository tutorial” (which uses the Repository API), “Room tutorial” (which uses the Room API), and “CDN tutorial” (which uses the CDN connector).
- PaaS Manager. This section explains the steps necessary for building your application and deploying it on the NUBOMEDIA PaaS, both the GUI and the PaaS GUI, paying special attention on the autoscaling mechanisms.
- Media Processing. This section contains links to the Video Content Analysis (VCA) and Augmented Reality (AR) parts developed within the NUBOMEDIA project.
- Framework tools. This section is devoted to contains some tools created in the NUBOMEDIA project aimed to simplify the development lifecycle of applications using NUBOMEDIA, such as the Visual Development Tool, Monitoring Tools, and the Autonomous Installer.
- Advanced. This is a miscellaneous section containing advanced information about NUOMEDIA, namely: NUBOMEDIA Architecture, Media Server Discovery Process, Content Distribution Network, and IMS Connector.

5.2 Community structure

As depicted in Figure 8, the Community structure page has three different parts:

- NUBOMEDIA as a community. This part contains a brief introduction about the NUBOMEDIA ecosystem.
- NUBOMEDIA source code structure. This section contains a dynamic tree structure with links to the three-tier architecture of the NUBOMEDIA repositories, as described in section 4.
- NUBOMEDIA architecture. This part of the page shows an architectural diagram of NUBOMEDIA. This diagram is interactive: users can mouse over it and click on the component of their interest. Upon click, a dialog box is opened with the information about the responsible partner, links to code repositories, issue tracker, and so on.



Community Structure

› NUBOMEDIA as a community

NUBOMEDIA ambition is to create a community and a full ecosystem of users, contributors and business models. The force joining the community shall come from a transparent and flexible open source software licensing scheme. This page summarizes the contributions done by the partners of the NUBOMEDIA Consortium is to the open source community. Follow NUBOMEDIA twitter account and be informed on our progresses for building and growing up such community.

› NUBOMEDIA source code structure

The NUBOMEDIA source code is available in **GitHub** under the organization nubomedia. This organization has a hierarchical structure that comprises 3 levels of repositories:

- nubomedia It is the bundle repository consolidating NUBOMEDIA as a whole
  - nubomedia-controller It comprises all the code providing the cloud management features
  - nubomedia-admin It comprises all the code related to NUBOMEDIA administration and management
  - nubomedia-demonstrators It comprises all the code related to the NUBOMEDIA FOSS demonstrators
  - nubomedia-documentation It comprises all the artifacts related to NUBOMEDIA high-level documentation
  - nubomedia-media-server It comprises all the artifacts related to Kurento Media Server and its module
  - nubomedia-sdks It comprises all the artifacts related to NUBOMEDIA SDKs, including Web, Android and IOS SDKs
  - nubomedia-connectors It comprises all the artifacts related to NUBOMEDIA FOSS Connectors

› NUBOMEDIA Architecture

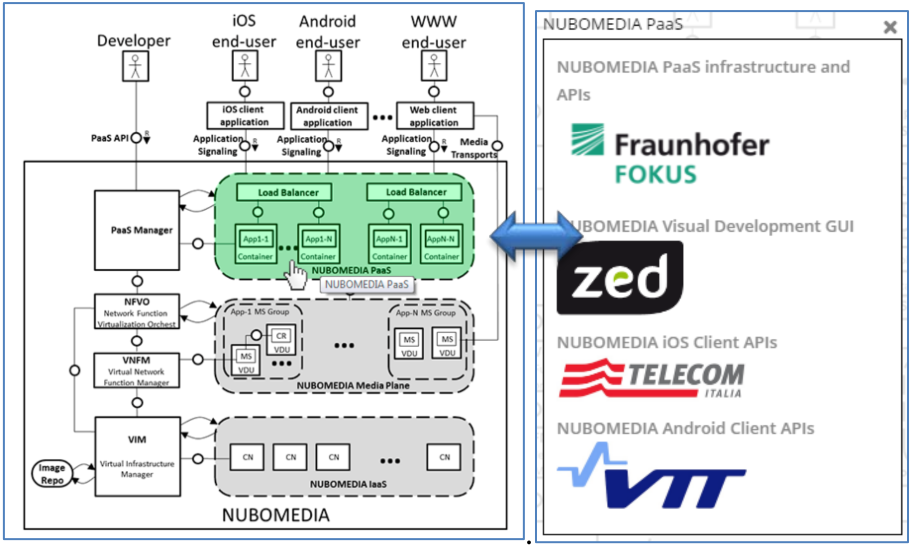
The following diagram is an interactive chart showing the **NUBOMEDIA architecture**. For further details about this architecture please visit the NUBOMEDIA Developer Guide. Please mouse over and click on each component of this figure to open a dialog box with the information about the responsible partner, links to code repositories, issue tracker, and so on.

Figure 8 NUBOMEDIA community web page screenshot

Regarding the NUBOMEDIA architecture chart, at the top of the picture we can see the different types of users of NUBOMEDIA, namely developers and end-users. There are different kinds of end-users, for example iOS, Android, and WWW users.

The first component within the NUBOMEDIA architecture is the PaaS (Platform as a Service), which we associate with the development capabilities exposed by NUBOMEDIA. As the other components, this part of the picture is highlighted when the mouse is over it (see left part of Figure 9). When clicking in this area, a dialog box opens showing the partners involved in the development of such capabilities (see right part of Figure 9). This component involves three main associated projects:

- NUBOMEDIA PaaS infrastructure and APIs (developed by Fraunhofer Gesellschaft zur Förderung der angewandten Forschung e.V.)
- NUBOMEDIA Visual Development GUI (developed by Zed Worldwide)
- NUBOMEDIA iOS Client APIs (developed by Telecom Italia).
- NUBOMEDIA Android Client APIs (developed by VTT)



NUBOMEDIA PaaS ×

NUBOMEDIA PaaS infrastructure and APIs

Fraunhofer FOKUS

NUBOMEDIA Visual Development GUI

zed

NUBOMEDIA iOS Client APIs

TELECOM ITALIA

NUBOMEDIA Android Client APIs

VTT

Figure 9 NUBOMEDIA PaaS

Following the same model, in the NUBOMEDIA Media Plane component (see Figure 10) there are a number of associated projects:

- NUBOMEDIA Media Server and Media APIs (developed by Universidad Rey Juan Carlos and Naevatec)
- NUBOMEDIA Video Content Analysis capabilities (by Visual Tools)
- NUBOMEDIA Augmented Reality Capabilities (created by VTT Technical Research Centre of Finland).

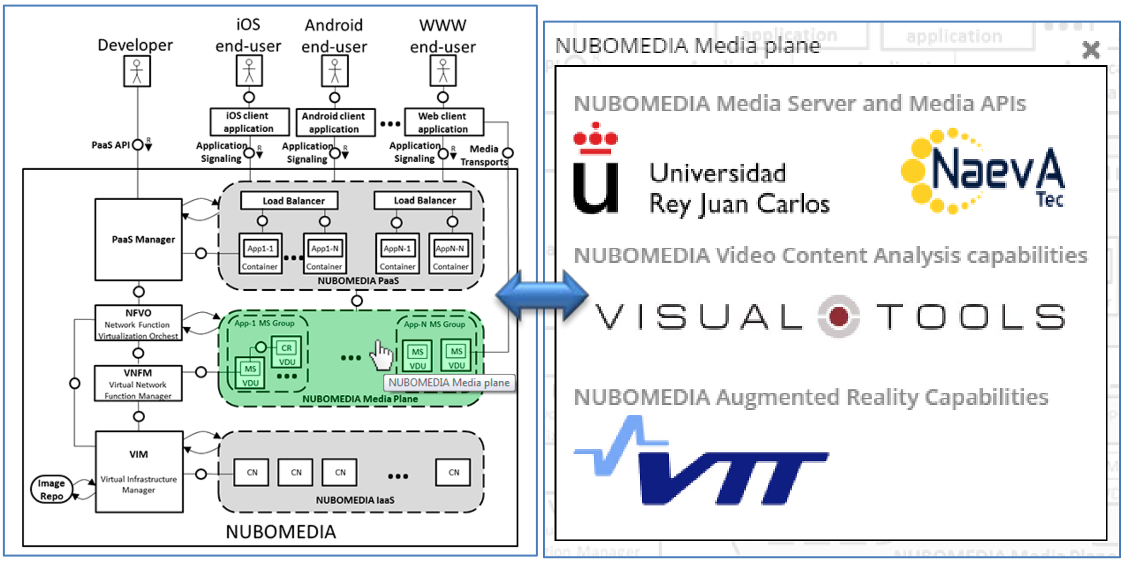


Figure 10 NUBOMEDIA Media Plane

The partner responsible of the NUBOMEDIA NFV Cloud Components (i.e. Network Function Virtualization Orchestrator, NFVO; and Virtual Network Function Manager, VNFM) is Technische Universität Berlin (see Figure 11).

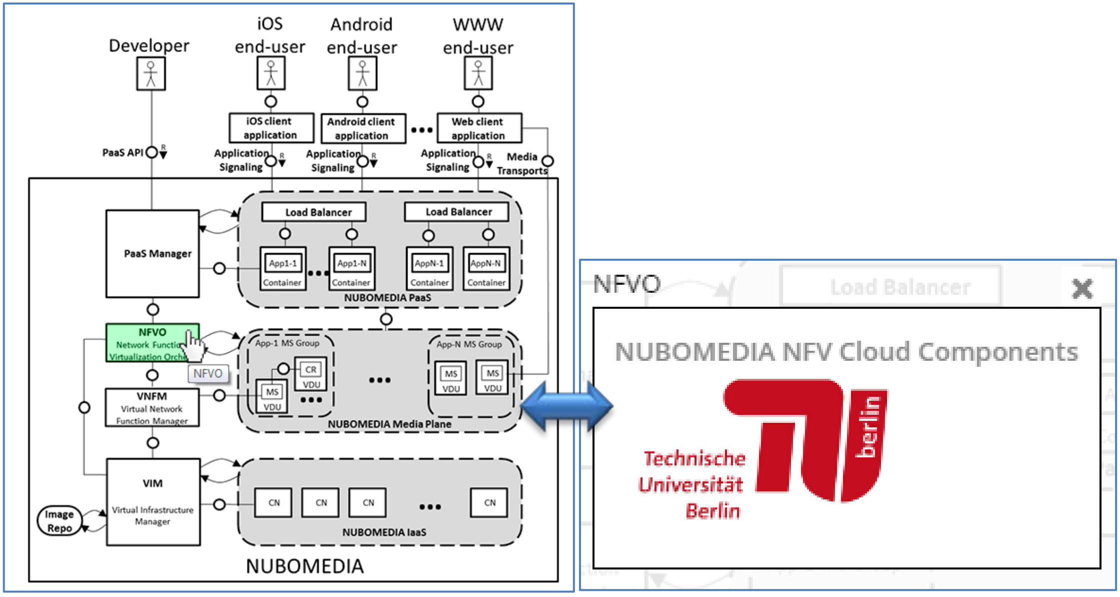


Figure 11 NUBOMEDIA Virtual Network

Finally, the NUBOMEDIA Virtual Infrastructure Management (Virtual Infrastructure Manager, VIM; and NUBOMEDIA Infrastructure as a Service, IaaS) has been developed by Universitatea Stefan cel Mare Suceava (Figure 12).

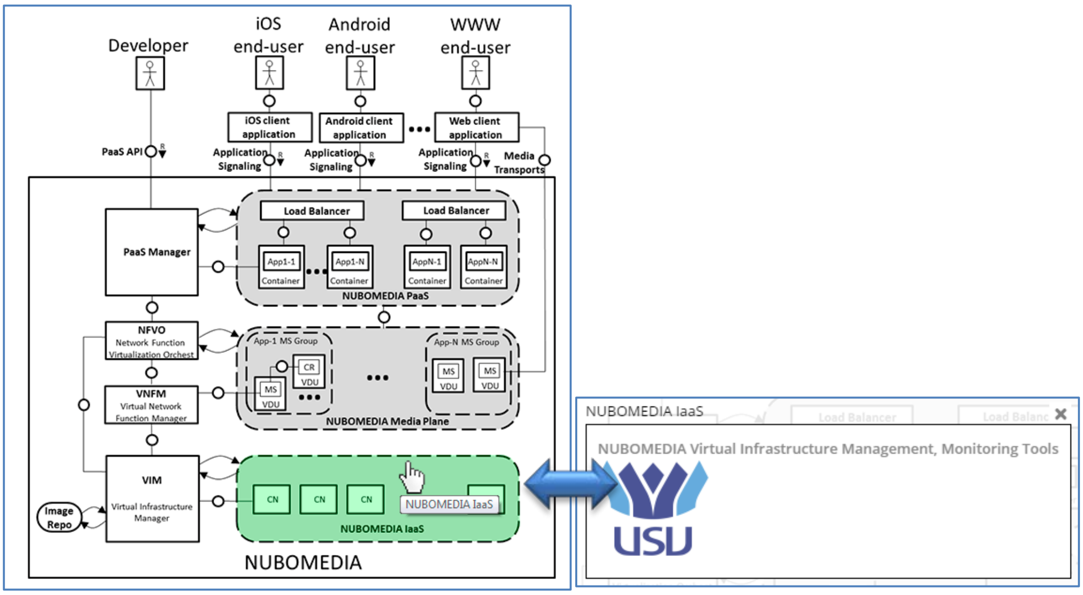
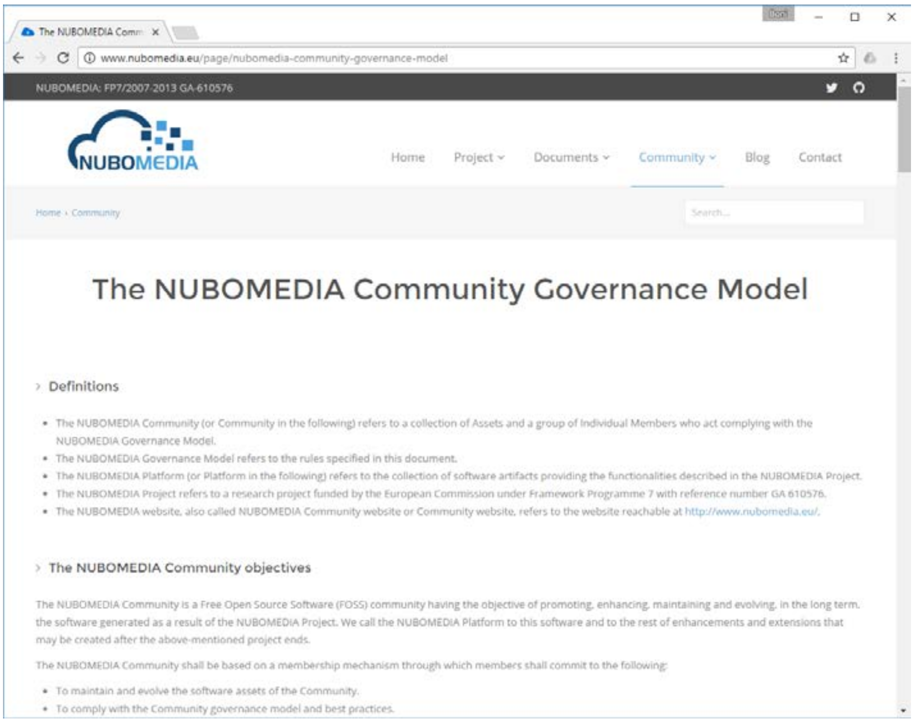


Figure 12 NUBOMEDIA Virtual Infrastructure Management

5.3 Community governance

In this section, we can find the NUBOMEDIA Community governance model approved by the project consortium (PC). The information contained in this page (see screenshot on Figure 13) is the same that contained in section 3.4 of this document.


The NUBOMEDIA Comm
www.nubomedia.eu/page/nubomedia-community-governance-model
NUBOMEDIA: FP7/2007-2013 GA-610576
NUBOMEDIA
Home
Project
Documents
Community
Blog
Contact
Home › Community
Search...

The NUBOMEDIA Community Governance Model

› Definitions

- The NUBOMEDIA Community (or Community in the following) refers to a collection of Assets and a group of Individual Members who act complying with the NUBOMEDIA Governance Model.
- The NUBOMEDIA Governance Model refers to the rules specified in this document.
- The NUBOMEDIA Platform (or Platform in the following) refers to the collection of software artifacts providing the functionalities described in the NUBOMEDIA Project.
- The NUBOMEDIA Project refers to a research project funded by the European Commission under Framework Programme 7 with reference number GA 610576.
- The NUBOMEDIA website, also called NUBOMEDIA Community website or Community website, refers to the website reachable at http://www.nubomedia.eu/.

› The NUBOMEDIA Community objectives

The NUBOMEDIA Community is a Free Open Source Software (FOSS) community having the objective of promoting, enhancing, maintaining and evolving, in the long term, the software generated as a result of the NUBOMEDIA Project. We call the NUBOMEDIA Platform to this software and to the rest of enhancements and extensions that may be created after the above-mentioned project ends.

The NUBOMEDIA Community shall be based on a membership mechanism through which members shall commit to the following:

- To maintain and evolve the software assets of the Community.
- To comply with the Community governance model and best practices.

Figure 13 NUBOMEDIA Community governance

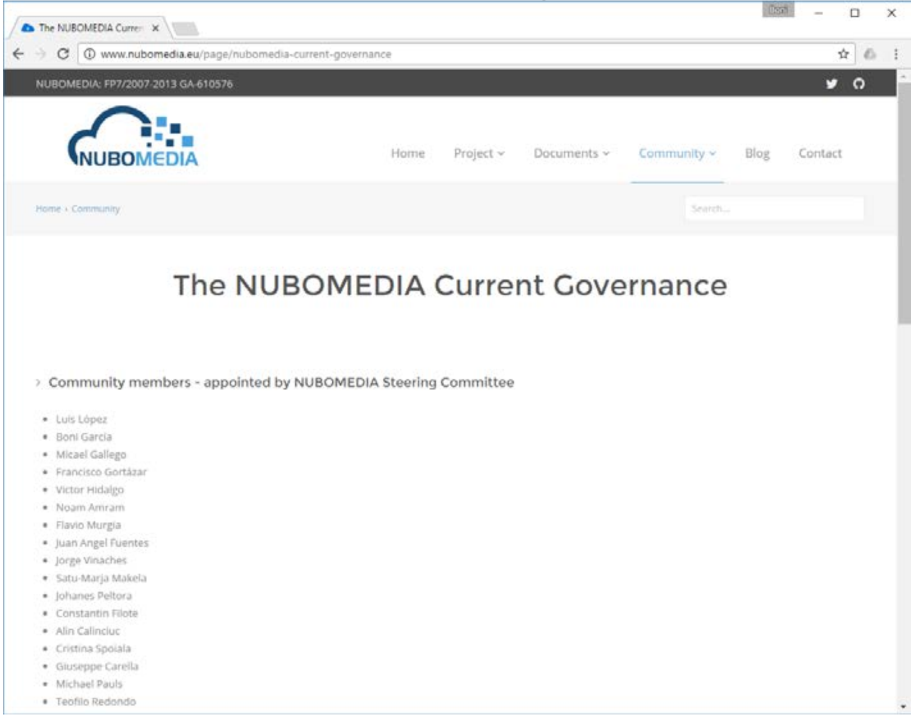
5.4 Current governance

In this section, we can find the list of Community members, Board of directors (BoD), and Project leads of the NUBOMEDIA project (Figure 14). This information is the following:

- Community members - appointed by NUBOMEDIA Steering Committee: Luis López, Boni García, Micael Gallego, Francisco Gortázar, Victor Hidalgo, Noam Amram, Flavio Murgia, Juan Angel Fuentes, Jorge Vinaches, Satu-Marja

Makela, Johanes Peltora, Constantin Filote, Alin Calinciuc, Cristina Spoiala, Giuseppe Carella, Michael Pauls, Teofilo Redondo, Claudio Venezia, and Fabio Luciano Mondin.

- Board of directors (BoD) - elected by members: Luis López, Micael Gallego, Giuseppe Carella, Alin Calinciuc, and Juan Angel Fuentes.
- Project leads - appointed by BoD:
    - nubomedia-controller/OpenBaton Project: Giuseppe Carella.
    - nubomedia-admin: Alin Calinciuc.
    - nubomedia-demonstrators: Boni García.
    - nubomedia-documentation: Boni García.
    - nubomedia-media-server/Kurento: Micael Gallego.
    - nubomedia-sdks: Satu-Marja Makela.
    - nubomedia-connectors: Flavio Murgia.



The NUBOMEDIA Current Governance

› Community members - appointed by NUBOMEDIA Steering Committee

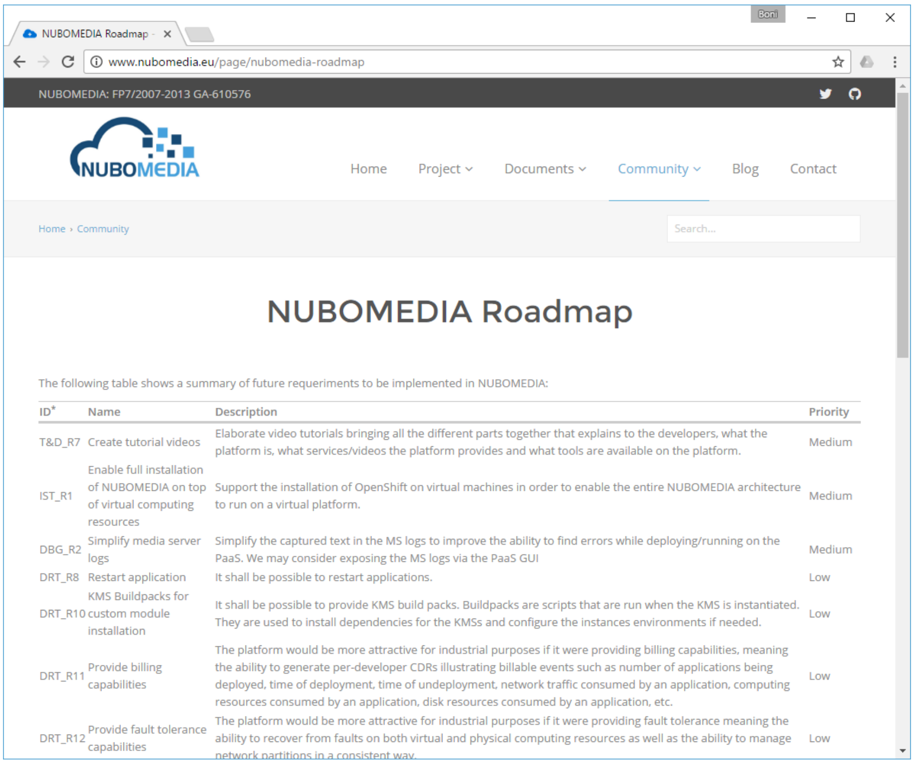
- Luis López
- Boni García
- Micael Gallego
- Francisco Gortázar
- Victor Hidalgo
- Noam Amram
- Flavio Murgia
- Juan Angel Fuentes
- Jorge Vinaches
- Satu-Marja Makela
- Johanes Peltora
- Constantin Filote
- Alin Calinciuc
- Cristina Spoiala
- Giuseppe Carella
- Michael Pauls
- Teofilo Redondo

Figure 14 NUBOMEDIA Current governance

5.5 Roadmap

This section contains a list of features of new features to be implemented in NUBOMEDIA (see Figure 15). These features are organized in a table containing the following information:

- Identifier: T&D (Tutorials and Demo Examples), IST (Installation Requirements), DBG (Debugging Requirements), DRT (Deployment and Runtime Requirements), KMS (Kurento Media Server Requirements), PGUI (PaaS GUI Requirements).
- Name and description of the new feature.
- Priority (low, medium, high).



NUBOMEDIA Roadmap

The following table shows a summary of future requeriments to be implemented in NUBOMEDIA:

ID*	Name	Description	Priority
T&D_R7	Create tutorial videos	Elaborate video tutorials bringing all the different parts together that explains to the developers, what the platform is, what services/videos the platform provides and what tools are available on the platform.	Medium
IST_R1	Enable full installation of NUBOMEDIA on top of virtual computing resources	Support the installation of OpenShift on virtual machines in order to enable the entire NUBOMEDIA architecture to run on a virtual platform.	Medium
DBG_R2	Simplify media server logs	Simplify the captured text in the MS logs to improve the ability to find errors while deploying/running on the PaaS. We may consider exposing the MS logs via the PaaS GUI	Medium
DRT_R8	Restart application	It shall be possible to restart applications.	Low
DRT_R10	KMS Buildpacks for custom module installation	It shall be possible to provide KMS build packs. Buildpacks are scripts that are run when the KMS is instantiated. They are used to install dependencies for the KMSs and configure the instances environments if needed.	Low
DRT_R11	Provide billing capabilities	The platform would be more attractive for industrial purposes if it were providing billing capabilities, meaning the ability to generate per-developer CDRs illustrating billable events such as number of applications being deployed, time of deployment, time of undeployment, network traffic consumed by an application, computing resources consumed by an application, disk resources consumed by an application, etc.	Low
DRT_R12	Provide fault tolerance capabilities	The platform would be more attractive for industrial purposes if it were providing fault tolerance meaning the ability to recover from faults on both virtual and physical computing resources as well as the ability to manage network partitions in a consistent way.	Low

Figure 15 NUBOMEDIA Roadmap

6 Community feedback

Different actions have been carried out to get valuable feedback from the NUBOMEDIA community. This section summarizes two of the most significant initiatives aimed to gather different types of feedback about NUBOMEDIA by developers.

First, a complete survey about the usability of the of the NUBOMEDIA Media API has been designed and carried out. Second, a Webinar about NUBOMEDIA has been performed, asking about the perception about NUBOMEDIA (PaaS, documentation, learning curve, etc.) to the participants. Next section contains detailed information about both actions.

6.1 NUBOMEDIA API Usability Survey

We present an evaluation of the NUBOMEDIA APIs usability based on the Cognitive Dimensions of Notations (CDs) [GREEN1989], which is a lightweight framework created for describing and analyzing the usability of notational systems, such as user interfaces, programming languages and APIs. CDs are used by designers for performing quantitative evaluation of API usability. The common practice for this is to use a questionnaire [BLACKWELL2000] requesting users to evaluate, through a Likert scale [ALLEN2007], how they experience CDs dimensions when performing their development activities.

Some authors propose an adaptation to the CDs framework based on transforming the dimensions into another base that is more meaningful for developers and that is compatible with shorter and simpler questionnaires [PICCIONI2013]. These new dimensions are called Clarke's dimensions and concentrate on 5 specific aspects of API usability: understandability, abstraction, expressiveness, reusability and learnability.

Understandability deals with evaluating the effort required for understanding how to use the API for achieving a desired functionality. This dimension encompasses aspects such as whether the API names are descriptive and the relation among API types and constructs are clear and unambiguous. This relates to the base CDs dimension called *closeness of mapping*. It also includes the ability of the API to avoid developers to manage hidden information not explicitly represented in the API, which is called *hidden dependencies* in terms of the base CDs dimensions. In addition, the base CDs dimension called *hard mental operations* also affects understandability. In brief, this dimension addresses how simple is to access API features through object creation, primitive invocations or other means.

Abstraction, which is itself a base CDs dimension, relates to the ability of the API to guarantee that programmers can use the API proficiency without requiring specific knowledge or assumptions in relation to its implementation details. Abstractions should match the conventions and practices of programmers, without being elegantly abstract at the expense of understandability or other practical concerns. Abstraction is typically correlated with the degree of comfort developers feel when using the API. Summarizing with a slogan, this research question asks whether the API "makes simple things simple, and complex things possible".

Expressiveness can be seen as the ability of inferring readily the purpose of an entity. This is related to the base CD dimension called *role-expressiveness*. Expressiveness is also related to how easy is for the programmer to build her code without needing to assume any specific cognitive model about API use. Intuitively, code written using expressive APIs tend to be simpler to read and transforming requirements into code is typically more efficient in expressive APIs. In terms of base CDs dimensions these properties are related to *visibility* and *consistency*. Moreover, expressive APIs do not impose constraints neither in the order or creation nor in the definiteness of the components comprising the code, which is related to the CDs dimensions called *premature commitment* and *provisionality*. We also consider the CDs base dimension called *error-proneness* to be part of the expressiveness properties of our API.

Reusability determines whether the client code is maintainable and extensible. In particular, this dimension address the typical concern on how hard is to modify pre-existing code and adapt it to slightly, extended or more general requirements. The main related base CDs dimensions is *viscosity*, understood as resistance to change, but it also involves other base dimensions such as *diffuseness* (i.e. the verbosity of the notation).

Learnability address the ability of the API learning process to be incremental. Learnable APIs enable developers to understand APIs in a gradual way without requiring initial disproportionate efforts, which is related to the CDs base dimension called *progressive evaluation*. Learnability also deals with whether performing a certain programming task using the API has a positive impact on performing other related but different tasks. This dimension might have some overlap with understandability, but emphasizes specifically the learning process rather than its practical outcomes.

6.1.1 Study design

Our study is based on a questionnaire that evaluates developers' experiences in terms of Clarke's dimensions in a Likert scale. The final objective of the study is to find answers to questions such as the following:

- Do developers feel that the API can be learnt in a simple, incremental and seamless way?

- Do developers feel that the API is helpful for the creation of clean and error-free application code without needing to manage low-level complexities?
- Do developers feel that maintaining and evolving code consuming the API is smooth and uncomplicated?
- Do developers have the same perception of the API usability with independence on their demographic characteristics (i.e. years of experience, nationality, etc.) and on the types of applications they create?

We created a questionnaire comprising 28 assertions which characterize the 5 target dimensions (understandability, abstraction, expressiveness, reusability, and learnability). For every assertion, users provide their degree of agreement or disagreement in a Likert scale from 1 (I fully disagree) to 5 (I fully agree). For assessing the internal consistency of the data, and following common practices in psychological research, some of the assertions are generated in negative terms. For example, if a responded expressed agreement with the claim "I feel this API is simple" and disagreement with "I find it's hard programming with this API" this would be an indication of internal consistency. We call these assertions and N-assertions. For the statistical analysis, answers of N-assertions are inverted (i.e. 1 is transformed into 5, 2 into 4, 4 into 2 and 5 into 1), so that consistency and coherence is maintained.

In addition to this, and with the objective of characterizing different aspects of participants, a number of questions were included for profiling demographic data and for evaluating their degree of experience. These questions are shown on the following table:

Dimension	ID	Assertion
Understandability	U.1	NUBOMEDIA APIs are, in general, easy to understand
	U.2	In NUBOMEDIA APIs object names are descriptive and unambiguous
	U.3	(N) I need to keep track of hidden information not represented by the APIs to create my applications
	U.4	(N) NUBOMEDIA APIs are obscure and it takes a huge effort to use them, even for creating simple applications
	U.5	NUBOMEDIA API objects, types, and primitives represent appropriately the underlying media-related concepts
	U.6	I understand the difference between a Media Pipeline and a Media Element
Abstraction	A.1	NUBOMEDIA APIs make simple to create applications without needing to worry about the low level media details
	A.2	(N) I needed to adapt the API (e.g. inheriting, overriding, etc.) for having it meet my needs
	A.3	(N) It's necessary to understand how codecs and protocols work for being able to use NUBOMEDIA APIs
	A.4	I like writing applications with NUBOMEDIA APIs. I'm familiar with their programming model
	A.5	I feel appealing and attractive the general approach of NUBOMEDIA APIs
	A.6	Creating simple applications with NUBOMEDIA APIs is simple. Creating complex applications is possible
Expressiveness	E.1	Developing with NUBOMEDIA fully matches the expectations I had
	E.2	I can translate my media application requirements into code in an easy way
	E.3	Reading an application code, I can understand what the application is doing in a simple way
	E.4	After creating an application, I can explain seamlessly to other people what I have done in terms of media elements and their interconnections
	E.5	(N) There are missing features in NUBOMEDIA APIs that make not possible to implement interesting applications
	E.6	(N) Programming with NUBOMEDIA APIs is error prone. You need to take into consideration a lot of details for having an application working
Reusability	R.1	(N) Creating applications requires too long and verbose code specifying too many things
	R.2	(N) Adding a recording capability to an non-recording application requires modifying a lot of code
	R.3	My code using NUBOMEDIA APIs can be maintained and evolved easily
	R.4	I can re-use NUBOMEDIA related code in a simple way
	R.5	(N) When using NUBOMEDIA APIs, there are many different ways of doing the same thing and I need to take too many decisions in the process
	R.6	Adapting my NUBOMEDIA-based application to new media requirements is quite simple
Learnability	L.1	I learned how to use NUBOMEDIA in an incremental way, starting with simple concepts and progressing towards complex applications
	L.2	(N) Programming with NUBOMEDIA requires learning a lot of classes and dependencies, even for applications.
	L.3	When I create a complex application, I can start by simple example and evolve it later in a seamless

		way
	L.4	(N) I needed to read all the documentations and tutorials to be able to create my first application
	L.5	Reading simple tutorials made me possible to understand better the complex ones and to create later applications complying with my requirements

The NUBOMEDIA APIs has been implemented as part of the Kurento project. This is a significant advantage because Kurento has been released as an Open Source Software and a community of developers has emerged around it. The size of the community is unknown but its main communication channel, the Kurento Public mailing list, has, at the time of this writing, 432 subscribers most of which are professional developers at different stages of the API learning process.

In this context, the survey protocol is simple. The questionnaire is designed for Kurento community members. This questionnaire is published as a web form and participants are invited to participate through an aseptic e-mail invitation sent to the Kurento Public mailing list. This mail is written trying to avoid any kind of bias on participants so that it just presents the survey objectives and exposes a privacy policy guaranteeing that no personal data is to be disclosed or used for other objectives than the ones of the survey. Participants are incentivized to participate only once by requiring logging into the web form system through a valid e-mail. The form makes mandatory to fill answers on all assertions for making submission possible (i.e. partially answered questionnaires are not considered). The web form system stores in a persistent database each participant's answers and makes possible to edit them during the duration of the survey period, which is limited to 2 weeks' time.

6.1.2 Results

After one week from the initial e-mail invitation a total of 17 participants had answered. Several reminders were sent to the Kurento Public mailing and the announcement was also published through different social channels, such as the Kurento Twitter account. In two weeks, 42 answers were received, which represents 9.7% of the number of Kurento Public mailing list subscribers. This is aligned with typical answer rates in surveys.

From a demographic perspective age of participants were distributed between 20 and 50 years old, being the more numerous group the one in the thirties, which account for 50% of the total participants. In relation to nationality, the poll was answered by developers from 20 nationalities on 4 different continents, being USA the country with more participants. When coming to expertise evaluation, participants accumulate, on average, 9.7 years of development experience, but with a relevant diversity (i.e. 1 year as minimum and 30 as maximum). Participants also declare to have invested, on average, more than 61 hours for learning and programming with Kurento technologies, which provides a reasonable guarantee on their ability to evaluate the API.

The results of the poll are summarized in the following table, where the main statistics for each of the assertions and their corresponding dimensions are depicted. As specified above, all N-assertions were inverted previous to the statistical analysis. Hence, magnitudes represent perceptions on API usability in positive terms (i.e. the higher the magnitude the better the developer's impression on the APIs). As it can be seen in Figure 16, on average, participants feel API usability properties are adequate, being reusability the dimension with highest rank and expressiveness the one with the lowest score. A detailed analysis for each dimension is presented in the following paragraphs.

Dimension	**ID**	**Mean**	**Median**	**Min**	**Max**	**STDDEV**

Understandability		3.46	3.58	2	4.67	0.58
	U.1	3.33	3	1	5	0.98
	U.2	4.26	4	2	5	0.86
	U.3	2.69	3	1	5	1.12
	U.4	3.26	3	1	5	1.17
	U.5	3.52	3.5	2	5	0.71
	U.6	3.71	4	1	5	1.20
Abstraction		3.39	3.42	1.83	4.23	0.57
	A.1	3.40	3.50	1	5	1.04
	A.2	3.24	3	1	5	1.28
	A.3	3.29	3	1	5	1.11
	A.4	3.26	3	1	5	0.86
	A.5	3.69	4	1	5	0.84
	A.6	3.48	4	2	5	0.92
Expressiveness		3.17	3.17	2	4.33	0.57
	E.1	3.29	3	2	5	0.71
	E.2	3.17	3	1	5	0.82
	E.3	3.43	4	1	5	0.97
	E.4	3.45	4	1	5	1.02
	E.5	2.76	3	1	5	1.19
	E.6	2.9	3	1	4	1.05
Reusability		3.47	3.5	1.67	4.67	0.52
	R.1	3.48	4	1	5	0.99
	R.2	3.67	4	1	5	1.16
	R.3	3.31	3	1	5	0.98
	R.4	3.6	4	1	5	0.7
	R.5	3.62	4	1	5	0.7
	R.6	3.14	3	1	4	0.75
Learnability		3.43	3.4	1.4	5	0.65
	L.1	3.88	4	2	5	0.77
	L.2	3.07	3	1	5	1.35
	L.3	3.67	4	2	5	0.95
	L.4	2.86	3	1	5	1.26
	L.5	3.69	4	1	5	0.95

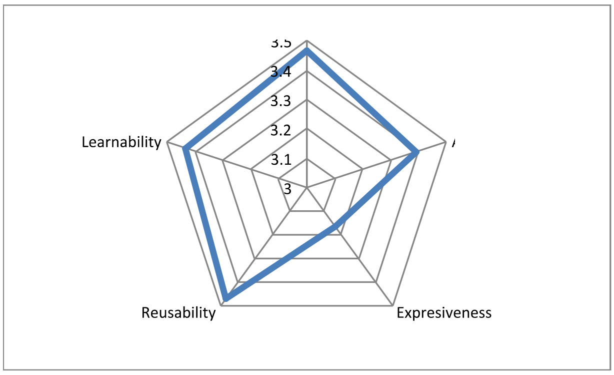


Figure 16. Radar chart showing average rankings on the 5 target dimensions of our questionnaire. Scale is set between 3 and 3.5 for evidencing the differences among dimensions.

Regarding the validity of this results, on the one hand, **internal validity** is a property associated to the extent to which a study minimizes systematic errors and avoid introducing bias into measurements. To formalize our internal validity analysis, we perform an additional test based on Cronbatch's alpha [CORTINA1993][CRONBACH1951], which is the most commonly used estimate for assessing the reliability of psychometric tests in social sciences. As Cronbatch's alpha is a measure of the internal consistency of data it needs all test items to measure the same construct. Due to this, its computation needs to be performed for each of our high-level dimensions separately. Cronbatch's alpha evaluation for our data is shown in the following table:

Dimension	**Cronbatch's alpha**
Understandability	0.77
Abstraction	0.73
Expressiveness	0.76
Reusability	0.76

Learnability	0.71

The thresholds for interpreting it are depicted in this table:

Cronbatch's alpha	**Internal consistency**
alpha >= 0.9	Excellent
0.9 > alpha >= 0.8	Good
0.8 > alpha >= 0.7	Acceptable
0.7 > alpha >= 0.6	Questionable
0.6 >= alpha >= 0.5	Poor
0.5 >= alpha	Unacceptable

As it can be observed, the reliability of the obtained data is inside acceptable margins, which is reasonable for our type of questionnaire and open research search methodology where there is no control on the who's, how's and why's of participants.

On the other hand, **external validity** refers to the extent to which a study can be generalized to other situations or populations. In relation to this, the main threads to external validity come from the research protocol, which was designed basing on specificities of our API implementation. In particular, the fact that we leveraged the Kurento Open Source software community for obtaining the test population is quite a strong restriction for the generalizability of our findings given that most newly designed APIs might not be Open Source and, even if they are, they do not need to have an active international community of more than 400 developers. Otherwise, the rest of our methodology as well as the analysis we performed and the conclusions we gathered from it do not assume any kind of specific requirements. This suggests that the gist of our findings are also applicable to different contexts and populations.

6.1.3 Discussion

We use the results of our study for answering the therein stated questions, namely:

Do developers feel that the API can be learnt in a simple, incremental and seamless way?

The evaluation of the learning ability of developers emerges mainly from two of the dimensions under analysis: understandability and learnability.

Regarding understandability, and as a general perception, participants feel the API understandability to be fine, with an average of 3.46 over all answers on this dimension. Through U.1 (3.33) we find a general declaration that the API is easy to understand. In particular, and as shown in assertion U.2 answers, participants feel outstanding (4.26) how object and primitive names are descriptive.

Regarding learnability, it is evaluated positively by participants (3.43 in average). The improvement area in this topic emerges from L.2 (3.07) and L.4 (2.86), which evidence the impressions of developers of needing to learn about a lot of API constructs and to read a relevant amount of documentation before being able to start using the API for useful things. On the other hand, as L.1 (3.88) evidences, the learning process seems to be compatible with an incremental approach where complexity is introduced in progressive steps.

Do developers feel that the API is helpful for the creation of clean and error-free application code without needing to manage low-level complexities?

The process of exploratory design, understood as the creative activity of creating application code consuming the API, is mainly related with two of our target dimensions: abstraction and expressiveness.

Regarding abstraction, we also find out a general positive evaluation (the overall average is 3.39). Answers on all questions are quite uniform, being A.5 the top ranked one (3.69), which shows that developers feel appealing the API approach; and A.2 the bottom ranked one (3.24) indicating that some developers feel the need of adapting the API to their needs. It is remarkable that A.2 has the largest standard deviation (1.28) of A assertions, which evidences some degree of controversy. This is confirmed when looking closely to answers: in A.2 answers, there are 12% of ones and 19% of fives, while considering all Abstraction answers the ratios for ones and fives are 4.3% 12.6 respectively.

The Expressiveness analysis also reflects positive evaluation but shows improvement areas. This is the less successful dimension with an overall average ranking of 3.17. Expressiveness limitations seem to emerge on assertions E.5 and E.6 (2.76 and 2.90 respectively). In particular, E.5 reveals that developers miss features that are relevant for their applications. E.6, in turn, manifests that the API gives not enough protection against failures. On the other hand, as demonstrated through E.3 and E.4, our API is easy to read (3.43) and is consistent when dealing with explaining code logic in terms of the API constructs (3.45).

Do developers feel that maintaining and evolving code consuming the API is smooth and uncomplicated?

Corrective and evolutive maintenance of the code is related to the dimension we call reusability. This dimension has highest ranking (3.47 in average). This is illustrated through results in assertions R.2, R.3 and R.6, which average to 3.67, 3.31 and 3.14 respectively. Also the API demonstrates nice properties in relation to verbosity, as shown by the 3.48 exhibited by R.1. Our API is considered concise and it is not too much verbose.

Do developers have the same perception of the API usability with independence on their demographic characteristics (i.e. years of experience, nationality, etc.) and on the types of applications they create?

We assume that API perception of usability is fine for all developers, with independence on their origin, culture or experience. For validating this assertion, we have performed several statistical analyses whose outcome is the following:

Correlation between API usability and programming experience.

There a tendency to positive correlation between the API usability scores and the developers' previous experience and knowledge. This is particularly true for the self-assessment expertise on Kurento technologies: developers having better perception on their expertise on Kurento feel the Kurento API to be more usable for their objectives. However, the negative effect is concentrated on users with very low expertise on Kurento technologies. In other words: as soon as a developer feels to have some initial knowledge on the API, his perception of the usability increases to be the same as the one of experts, which is a good symptom. Interestingly enough, this effect seems not to be correlated with the number of hours declared in learning or programming with Kurento. This means that the time invested by developers in learning or programming with the API does not have a strong influence on their perception of API usability. Hence, we can conclude that, as long as an initial knowledge about the API and its foundations is known, the API usability perception does not depend significantly on the proficiency of developers.

Relation between API usability and nationality/culture.

For analyzing this, we have consolidated developer's nationalities in continents. Surprisingly, there is a clear tendency of USA developers to evaluate the API usability with less score. We do not have a consistent explanation for this fact, but we believe it might be caused by some specific cultural bias that may be related by the fact that the API (and its documentation) creators are not native English speakers, which may decrease the perception of "quality" from native English speakers. In any case, this tendency is not quantitative significant and we do not consider it invalidates our independence-of-culture hypothesis.

Relation between API usability and the type of application being created.

The API usability scores do not seem to have a clear dependency on any of the analyzed dimensions with the type of application being created. Hence, we may conclude that the API usability perception does not depend on the specificities of the features a given application consumes.

6.2 NUBOMEDIA Webinar

In order to disseminate NUBOMEDIA, we have carried out an online webinar. Due to the fact that Kurento is the heart of NUBOMEDIA, and Kurento has already a mass of active developers, the Kurento mailing list was used to announce the Webinar (https://groups.google.com/forum/#!forum/kurento). On 3rd October 2016, we wrote this post to the list:

Dear developers,

As you might know, NUBOMEDIA is a private open source WebRTC PaaS enabling the development, deployment and execution of Kurento-based WebRTC applications with full auto-scaling.

With the objective of improving the awareness of NUBOMEDIA, we are going to celebrate a live webinar and training session where we'll be explaining to any interested developer how to install and use NUBOMEDIA for creating and managing large-scale WebRTC services based on Kurento.

This training session will last for 1 hour and will be based on the following agenda:
1. Introduction to NUBOMEDIA
2. How to deploy a Kurento application in NUBOMEDIA
3. Live demo
4. Questions and answers

6.2.1 Study design

The content of the NUBOEMDIA webinar was made public on Slideshare (http://www.slideshare.net/BoniGarca/nubomedia-webinar). During the webinar, the foundations of NUBOMEDIA is introduced to the audience, putting the accent in what is needed to transform a regular Kurento application into a NUBOMEDIA application ready to be deployed using the PaaS Manager. A live demo deploying a NUBOMEDIA application in the PaaS is done by means of desktop sharing.

In order to make possible the attendance to people from different time zones to the webinar, two sessions were planned:

- Wednesday 26th October at 17:00 CEST
- Friday 28th October at 09:00 CEST

In order to get the participants' valuable feedback, an online questionnaire was also created. This form contains two types of questions: close (i.e., questions whose reply should be selected within a set of possible answers) and open (i.e., questions in which the participant has a text box to freely write his/her reply). The former type of questions was mandatory to be filled, whilst the later was optional.

On the one hand, the close questions (and the possible answers) are depicted the following table:

Question	**Answers**
Kurento skills *Which are your skills with Kurento?*	• Beginner. I never used Kurento before • Intermediate. I managed to run Kurento tutorial(s) • Advanced. I managed to create my own Kurento application(s) • Pro. I use Kurento in production application(s)
NUBOMEDIA Development *Create applications to be deployed on the NUBOMEDIA PaaS is easy*	• Strongly disagree • Disagree • Neither agree nor disagree • Agree • Strongly agree
NUBOMEDIA Learning Curve *The learning curve for NUBOMEDIA is soft for a Kurento developer*	• Strongly disagree • Disagree • Neither agree nor disagree • Agree • Strongly agree
NUBOMEDIA PaaS *The NUBOMEDIA PaaS manager is usable (i.e. easy to use)*	• Strongly disagree • Disagree • Neither agree nor disagree • Agree • Strongly agree
NUBOMEDIA Debug *The NUBOMEDIA PaaS manager provides enough features to debug applications (access to logs, etc)*	• Strongly disagree • Disagree • Neither agree nor disagree • Agree • Strongly agree
NUBOMEDIA Documentation *The NUBOMEDIA documentation is convenient for developers*	• Strongly disagree • Disagree • Neither agree nor disagree • Agree • Strongly agree
NUBOMEDIA Use *If it were public available, I would be willing to use NUBOMEDIA*	• Only if it were free • Paying per usage (points consumed by the PaaS) • Paying per applications deployed on the PaaS • Paying per instances consumed in the PaaS (MEDIUM, LARGE, etc) • Other
About the Webinar	• Awful. It was a waste of time.

What is your opinion about this webinar?	• Bad. It does not meet my expectations. • Normal. It was as I expected. • Good. It was better that I expected. • Superb. It was much better that I expected

On the other hand, the open questions are the following:

- What is the feature (or features) of NUBOMEDIA you like the most?
- What is the most important feature (or features) you are missing on NUBOMEDIA?
- Do you plan to use NUBOMEDIA in your developments? If not, can you specify why?
- What NUBOMEDIA should be providing in order for you consider using it as your reference WebRTC PaaS?
- Open comments. Feel free to drop a line about your opinion about Kurento/NUBOMEDIA.

6.2.2 Results

A total of 21 developers from 11 different countries (USA, Jamaica, Mexico, France, Netherlands, Spain, Italy, Russia, India, and Indonesia) singed up the webinar. Finally, 11 participants (out of these 21) joined the webinar (6 in the first session, and 5 the second one).

In general, people attending to the webinar had a very positive attitude, asking interesting questions and showing interest about NUBOMEDIA. After the Q/A phase of the webinar the presenter kindly asked the participants to fill the feedback form. As a result, 6 complete answered (out of 11) was gathered. The following charts show the pie chart for each close question results:

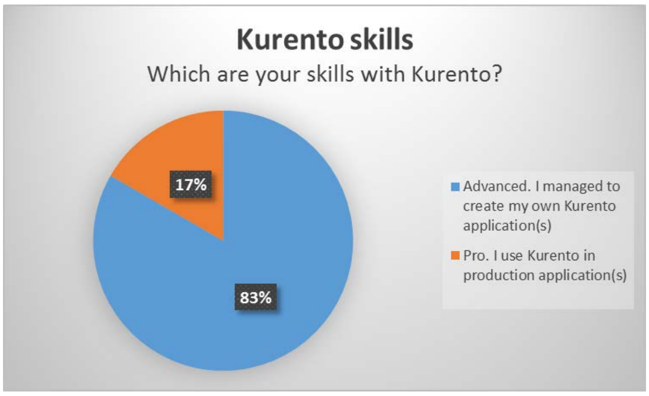


Figure 17. Kurento skills. Which are your skills with Kurento?

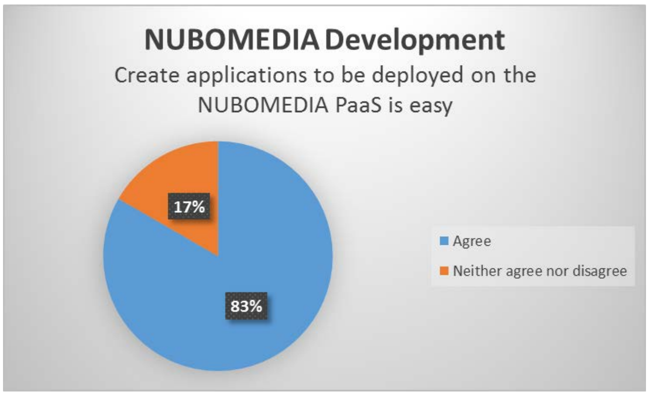


Figure 18. NUBOMEDIA Development. Create applications to be deployed on the NUBOMEDIA PaaS is easy.

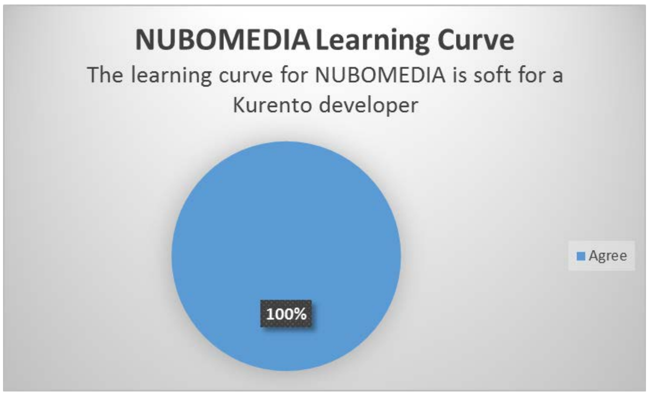


Figure 19. NUBOMEDIA Learning Curve. The learning curve for NUBOMEDIA is soft for a Kurento developer.

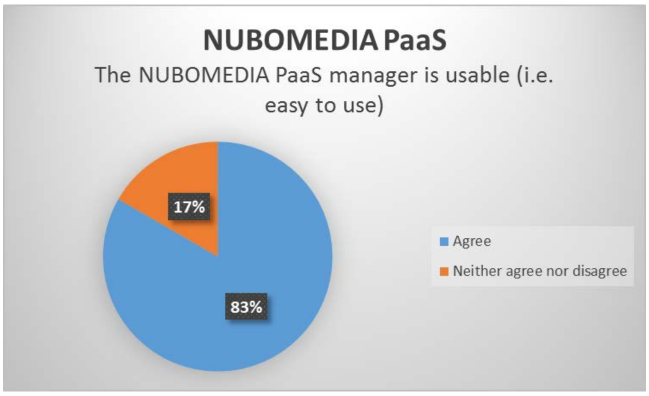


Figure 20. NUBOMEDIA PaaS. The NUBOMEDIA PaaS manager is usable (i.e. easy to use).

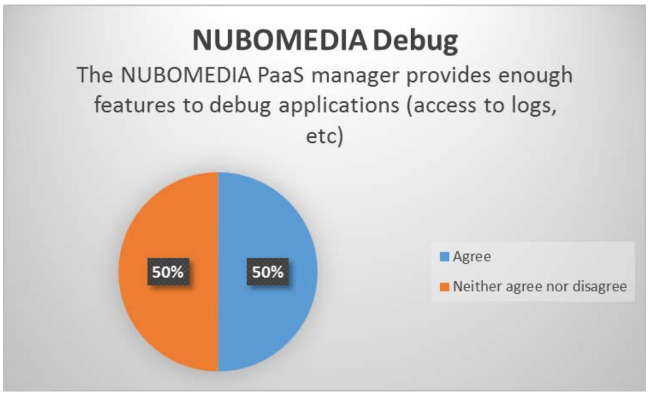


Figure 21. NUBOMEDIA Debug. The NUBOMEDIA PaaS manager provides enough features to debug applications (access to logs, etc).

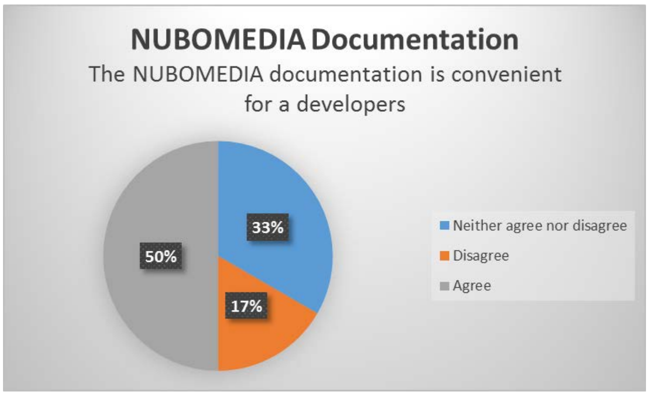


Figure 22. NUBOMEDIA Documentation. The NUBOMEDIA documentation is convenient for developers.

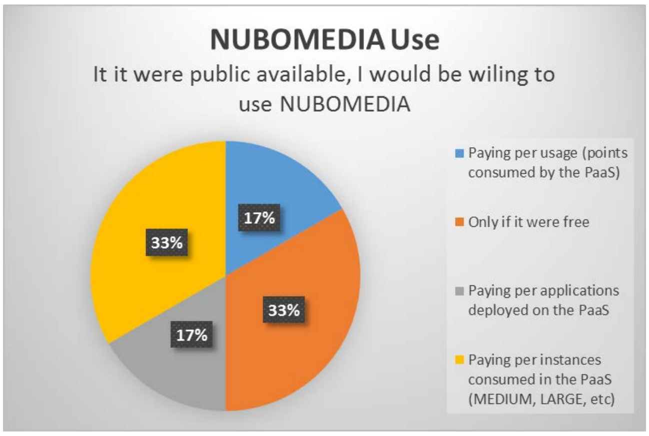


Figure 23. NUBOMEDIA Use. If it were public available, I would be willing to use NUBOMEDIA.

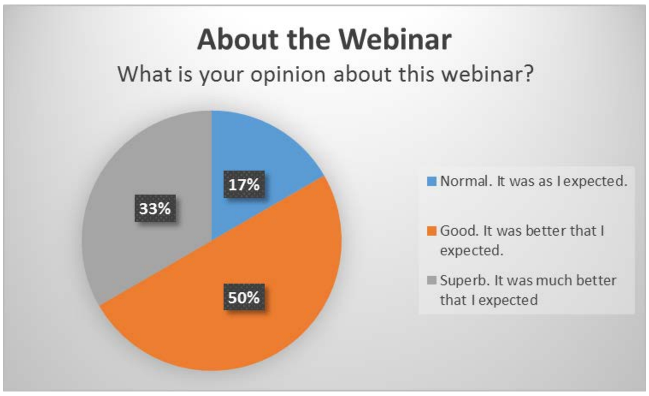


Figure 24. About the Webinar. What is your opinion about this webinar?.

6.2.3 Discussion

In the light of the feedback of the webinar we can draw some conclusions. First of all, it is important to notice that every participant (at least those who have completed the form) are experienced with Kurento (17% advanced and 83% pro users). This is an important indicator because NUBOMEDIA can be difficult to understand for a regular developer for the first time. The fact of being experience with Kurento allows to get rid of bias due to beginner doubts.

The next close questions give an initial overview about how NUBOMEDIA is perceived by Kurento developers:

- NUBOMEDIA Development. The 83% of replies agree that creating applications to be deployed on the NUBOMEDIA PaaS is easy, while the 17% are not sure (neither agree nor disagree). This can be seen as a positive indicator.

We need to take into account that attendances are learning about NUBOMEDIA for the first time, and at least none of them are considering the NUBOMEDIA development concepts hard to understand.

- NUBOMEDIA Learning Curve. In our study, the learning curve for NUBOMEDIA is soft for the 100% of Kurento developer. This a very good result, and shows that NUBOMEDIA can be easily adopted by Kurento developers.
- NUBOMEDIA PaaS. The result of this question is fully aligned the replies about the NUBOMEDIA development (83% agree, 17% nor agree nor disagree). Again, this can be understood as a positive feedback, due the fact the NUBOMEDIA PaaS is perceived as usable tool for developers.
- NUBOMEDIA Debug. The results of this question similar: 50% agree, 50% nor agree nor disagree. We can conclude at least are not missing any important feature to debug applications, otherwise some disagree answer should have been collected.
- NUBOMEDIA Documentation. There is no agreement in this point. The 50% of developers thought that the NUBOMEDIA documentation is convenient, while the 33% do not agree nor disagree and 17% disagrees. This might be an indicator that the documentation is quite stable and usable, but nevertheless some developers found some defects. Therefore, we can extract the conclusion that the documentation can be improved a bit.
- NUBOMEDIA Use. This is one of the most significant part of the survey. We found heterogeneous replies to this matter. It is quite remarkable that the 77% of the developers of this sample were willing to pay for production-ready version of NUBOMEDIA. This amount is distributed as follows:
  - 33% paying per consumed instances (MEDIUM, LARGE, etc.)
  - 17% paying per consumed points
  - 17% paying per deployed applications

  All in all, these figures can be seen as a very positive feedback, since the most of the developer feels that NUBOMEDIA is indeed providing a valuable service for developers, and they are willing to pay for it with different payment methods.
- About the Webinar. We can conclude that the webinar has been successfully carried out. The developers' opinion was quite good: 50% good, 33% superb, and 17% normal.

Regarding the open questions, as depicted in the design section (6.2.1), these questions was optional. Nevertheless, we obtained different answers that deserve to be compiled:

- *What is the feature (or features) of NUBOMEDIA you like the most?* Here we got 4 answers out of 6. These answers pointed out the importance of scalability and elasticity provided by NUBOMEDIA.
- *What is the most important feature (or features) you are missing on NUBOMEDIA?* The only comment we received here has to with the possibility of adding more analytical tools to the platform.
- *Do you plan to use NUBOMEDIA in your developments? If not, can you specify why?* 2 out of the 6 replies indeed was considering using NUBOMEDIA to deploy their applications.
- *What NUBOMEDIA should be providing in order for you consider using it as your reference WebRTC PaaS?* One of the developers asks for a transparent way of resource consumption (the points mechanism seems a bit unclear for that user). Other developer claims for an improvement in the documentation.

7 Community status

7.1 Community popularity indicators

For evaluating the community popularity, we use two indicators:

- Traffic on the main community websites.
- Traffic on the main public mailing list websites

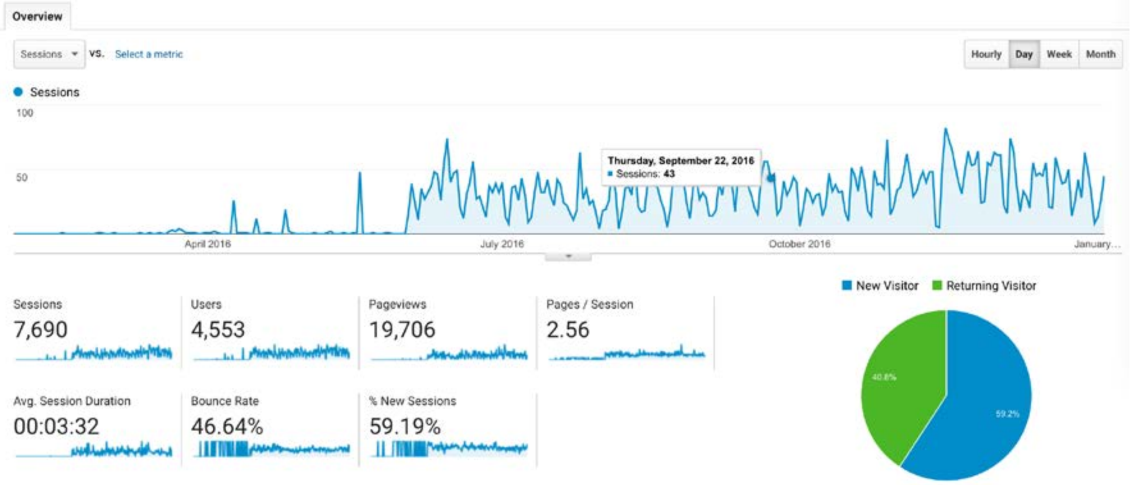
For completeness, these indicators are evaluated both on the NUBOMEDIA website and public mailing list, and on the Kurento website and public mailing list. This is justified given the intimate relationship of Kurento and NUBOMEDIA communities that is caused by the following facts:

- Despite Kurento being the result of different projects involving both public and private funding, NUBOMEDIA has been the project contributing to it so that Kurento, in its current form, can be considered as NUBOMEDIA foreground. Hence, the success of Kurento is also the success of NUBOMEDIA and vice versa.
- Developing Kurento applications is a required milestone for developing NUBOMEDIA applications. Due to this, many users understand both communities to be the same and consider NUBOMEDIA just as a cloud version of Kurento. Thus, the Kurento mailing list and the Kurento website is typically used by users interested in NUBOMEDIA.
- Kurento is part of the NUBOMEDIA ecosystem given that Kurento is a NUBOMEDIA associated project under the internal name “nubomedia-media-server”.

Due to this, it is fair to measure the NUBOMEDIA community indicators as the aggregation of both the NUBOMEDIA and the Kurento communities separately.

7.1.1 Community website indicators

The traffic on the NUBOMEDIA community website is illustrated on Figure 25.



Sessions 7,690

Users 4,553

Pageviews 19,706

Pages / Session 2.56

Avg. Session Duration 00:03:32

Bounce Rate 46.64%

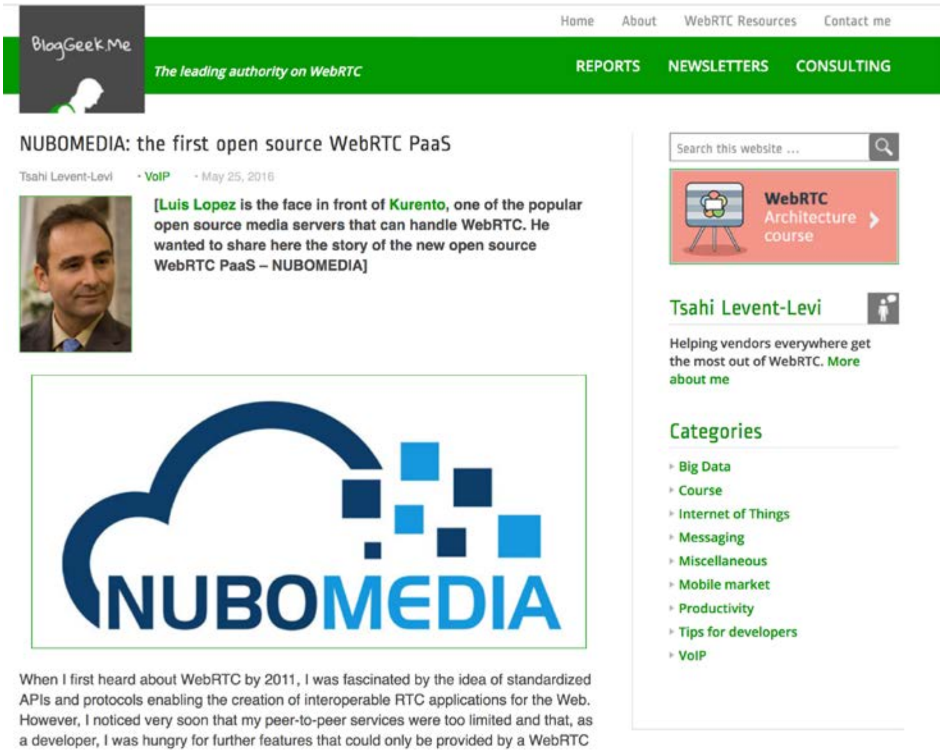
% New Sessions 59.19%

Figure 25. Traffic statistics showing www.nubomedia.eu website visitors in number of unique sessions per day. Statistics collection start on February 2016 and end on January 2017. As it can be observed, the NUBOMEDIA communication strategy that was activated by Spring 2016 successfully increased the traffic on the website.

As it can be observed, the website traffic increased significantly after the successful communication activities that took place during Spring 2016. This increase was caused by two posts devoted to NUBOMEDIA that appeared in two of the most important blogs devoted to WebRTC:

- bloggeek.me, see Figure 26

- webrtchacks.com, see Figure 27



NUBOMEDIA: the first open source WebRTC PaaS

Tsahi Levent-Levi · VoIP · May 25, 2016

[Luis Lopez is the face in front of Kurento, one of the popular open source media servers that can handle WebRTC. He wanted to share here the story of the new open source WebRTC PaaS – NUBOMEDIA]

When I first heard about WebRTC by 2011, I was fascinated by the idea of standardized APIs and protocols enabling the creation of interoperable RTC applications for the Web. However, I noticed very soon that my peer-to-peer services were too limited and that, as a developer, I was hungry for further features that could only be provided by a WebRTC

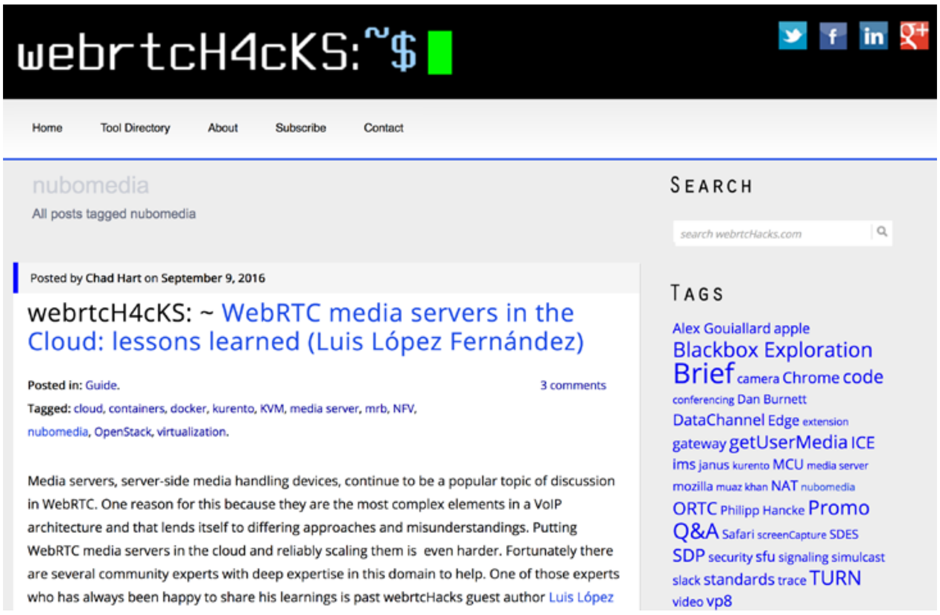
Tsahi Levent-Levi

Helping vendors everywhere get the most out of WebRTC. More about me

Categories

- Big Data
- Course
- Internet of Things
- Messaging
- Miscellaneous
- Mobile market
- Productivity
- Tips for developers
- VoIP

Figure 26. Post featuring NUBOMEDIA that was published in one of the most popular blogs in the WebRTC area. The post is accessible at this URL: https://bloggeek.me/nubomedia-webrtc-paas/



nubomedia

All posts tagged nubomedia

Posted by Chad Hart on September 9, 2016

webrtcH4cKS: ~ WebRTC media servers in the Cloud: lessons learned (Luis López Fernández)

Posted in: Guide. 3 comments

Tagged: cloud, containers, docker, kurento, KVM, media server, mrb, NFV, nubomedia, OpenStack, virtualization.

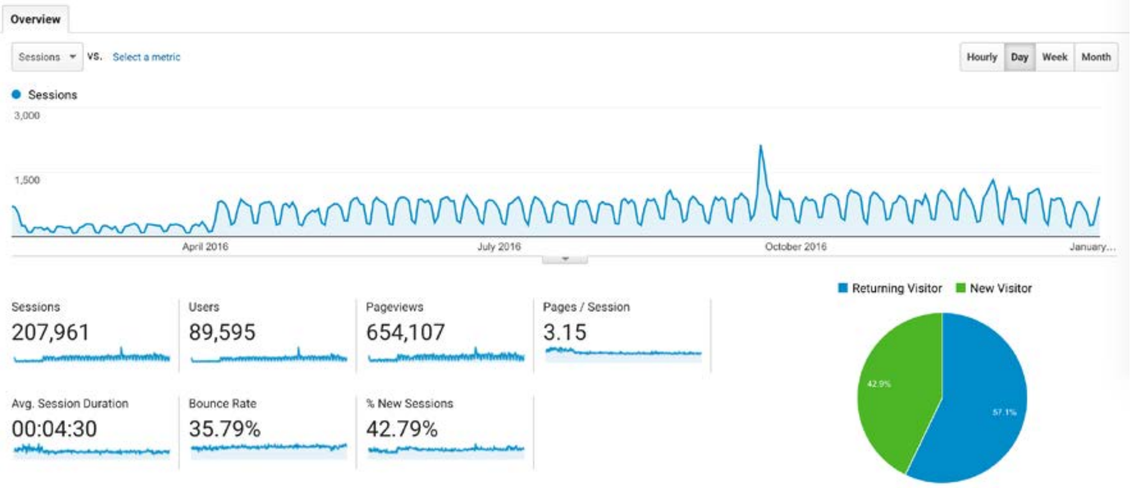
Media servers, server-side media handling devices, continue to be a popular topic of discussion in WebRTC. One reason for this because they are the most complex elements in a VoIP architecture and that lends itself to differing approaches and misunderstandings. Putting WebRTC media servers in the cloud and reliably scaling them is even harder. Fortunately there are several community experts with deep expertise in this domain to help. One of those experts who has always been happy to share his learnings is past webrtcHacks guest author Luis López

TAGS

Alex Gouiallard apple Blackbox Exploration Brief camera Chrome code conferencing Dan Burnett DataChannel Edge extension gateway getUserMedia ICE ims janus kurento MCU media server mozilla muaz khan NAT nubomedia ORTC Philipp Hancke Promo Q&A Safari screenCapture SDES SDP security sfu signaling simulcast slack standards trace TURN video vp8

Figure 27. Post featuring NUBOMEDIA that was published in one of the most important blogs in the WebRTC area. The post is accessible at this URL: https://webrtchacks.com/webrtc-media-servers-in-the-cloud/

In relation to the Kurento open source software community, the website traffic is depicted in Figure 28. As it can be seen, this community has a relevant popularity with close to 1,000 unique visitors per day on working days.



Overview

Sessions ▾ VS. Select a metric

Hourly Day Week Month

● Sessions

Sessions 207,961

Users 89,595

Pageviews 654,107

Pages / Session 3.15

Avg. Session Duration 00:04:30

Bounce Rate 35.79%

% New Sessions 42.79%

Figure 28. Traffic statistics showing www.kurento.org website visitors in number of unique sessions per day. Statistics collection start on February 2016 and end on January 2017. As it can be observed, the Kurento website also enjoyed a relevant traffic increase by Spring 2016.

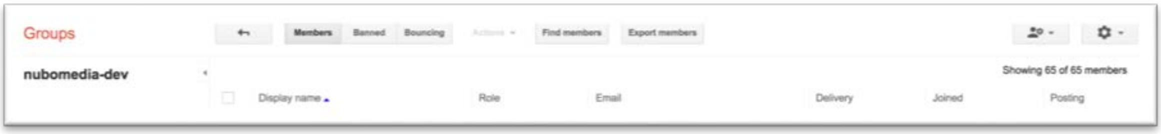
7.1.2 Mailing lists indicators

The NUBOMEDIA community main mailing list is the so called nubomedia-dev mailing list that can be accessed through:

- The nubomedia-dev@googlegroups.com e-mail address
- The https://groups.google.com/forum/#!forum/nubomedia-dev group

As shown in Figure 29, the nubomedia-dev mailing list has gained relevant popularity during its 6 months of activity accounting to 65 subscribers. 38 of which correspond to external developers (i.e. developers do not belonging to the NUBOMEDIA consortium). This mailing list has discussed around 100 topics during the last few months being the following the most popular community requests:

- Support related to Android and iOS SDKs
- Support related to installing NUBOMEDIA
- Requests related to the NUBOMEDIA concept and objectives



Groups

nubomedia-dev

Showing 65 of 65 members

□ Display name ▴ Role Email Delivery Joined Posting

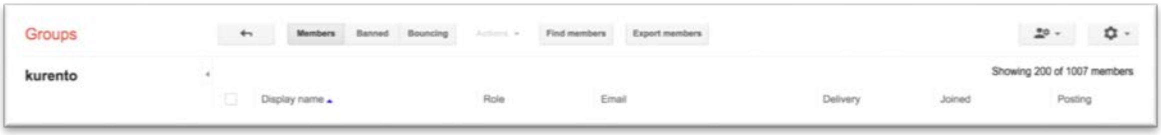
Figure 29. At the time of this writing, the nubomedia-dev mailing list had 65 subscribers, corresponding 27 of them to NUBOMEDIA consortium developers and 38 to developers external to the consortium.

In turn, Figure 30 shows the Kurento mailing list indicators. This list can be accessed through:

- The kurento@googlegroups.com e-mail address.
- The https://groups.google.com/forum/#!forum/kurento group

The Kurento mailing list has become one of the main discussion fora in the area of WebRTC media servers with more than 1,000 subscription and 2,800 discussion threads. In particular, the following topics are popular:

- Support related to installing and configuring Kurento Media Server.
- Support related to NUBOMEDIA Android and iOS SDKs
- Requests related to the relationship between NUBOMEDIA and Kurento.



Showing 200 of 1007 members

Figure 30. At the time of this writing the Kurento public mailing list had 1007 subscribers, being one of the most active communities in the WebRTC ecosystem arena.

7.2 Community SWOT analysis

In its current status, the SWOT analysis of the community is the following:

Strengths

- Concept: The NUBOMEDIA concept is attractive. Many relevant companies and institutions have shown interest the main ideas behind NUBOMEDIA.
- Timely arrival: WebRTC is currently exploding and thousands of users, companies and institutions are willing to use WebRTC enabled platforms.
- Traction: The NUBOMEDIA initiative, in general, and Kurento in particular, have gained relevant traction worldwide and both are a well-known name in the area

Weaknesses

- Fragmentation: The NUBOMEDIA project has not been fully able to eliminate the project fragmentation and the platform, as a whole, is still fragmented.
- Lack of commercial deployments: No commercial deployments for the platform have taken place.
- Stability and support: The platform is a successful proof of concept, but it lacks that appropriate stability and support for being used in production applications.
- Heterogeneity: The quality of the software artifacts comprising NUBOMEDIA is very heterogeneous. There are some of them that provide a clear value and have been created using reasonable software engineering practices, while others do neither provide a clear value over the state-of-the-art and nor comply with minimal software engineering requirements.

Opportunities

- Active research activities beyond the project: Several research initiatives have been launched for pushing further several NUBOMEDIA components beyond the project.
- Active exploitation activities beyond the project: Several partners have active and profitable exploitation activities based on NUBOMEDIA results.
- Active interest of relevant market stakeholders on the platform: Several large stakeholders in the WebRTC market have shown their interest in NUBOMEDIA opening a potential funding line for maintaining the community.

Threats

- Financial starvation: The lack of specific funding for maintaining and evolving NUBOMEDIA as a whole increases the risks of financial starvation of the community.
- Fragmentation after project ends: Several project partners do not show active interest/capability in evolving and maintaining NUBOMEDIA once the project ends. This further increases the risk of fragmentation.

7.2.1 Community sustainability analysis

Predicting the future is always complex. Due to this, we cannot assess whether the NUBOMEDIA community may be sustainable in the long term or not. However, we can remark a number of facts that emerge from the analyses above and that may shed some light on the real possibilities of the community to survive. These are the following:

- The NUBOMEDIA software stack is huge and complex. It's maintenance and evolution is costly and requires tight coordination among the maintainers of its different components.
- Some project partners are implementing successful exploitation plans for different NUBOMEDIA components, but there is not a clear exploitation plan for NUBOMEDIA as a whole.
- The idiosyncrasy of European Research projects makes complex to maintain the cohesion of the consortium after the project ends.
- The NUBOMEDIA platform is just the result of a research project and it lacks features that may be required for direct exploitation such as billing, fault tolerance, security enforcing mechanisms, etc.
- Some project partners have obtained relevant funding for pushing forward research ideas emerging from NUBOMEDIA or related to NUBOMEDIA components.

In this situation, the most probable forecast we can predict are the following:

- It is quite probable that significant parts of NUBOMEDIA may survive in the long term either as part of a reduced NUBOMEDIA community or as an independent community based on independent exploitation and funding schemes.
- It is quite probable that significant parts of NUBOMEDIA may continue to be used in research contexts.
- It is quite probable that the NUBOMEDIA community, as a whole, may not survive in the long term without further private investments. From this perspective, the first initial steps of the community should be devoted to finding the appropriate partners wishing to push NUBOMEDIA toward a full production platform and having the ability to fund such activities.

8 Conclusions

Given the analyses and indicators enumerated above, we believe that the project objectives related to the NUBOMEDIA Community have been largely satisfied. Thanks, the project effort, a vibrant ecosystem has been grown around NUBOMEDIA technologies articulated both through the NUBOMEDIA Community itself and through the Kurento Open Source Software Community. This ecosystem has gained worldwide relevance in the WebRTC arena with presence in the main blogs and events of this technological area and becoming a worldwide reference for WebRTC cloud platforms and WebRTC media infrastructures. We estimate that more of 3,000 developers have used somehow NUBOMEDIA generated technologies and that more than 5,000 companies have used them for prototyping and validating WebRTC services.

Despite this success as a project, NUBOMEDIA needs to mature as a technological platform. The long-term sustainability of NUBOMEDIA as a whole could only be guaranteed when a ecosystem of companies and organizations decide to push it forward to create a production-ready infrastructure. The first steps of the community once the project ends should be oriented towards generating such ecosystem gathering further public or private investments.

9 References

[ALLEN2007] Allen IE, Seaman CA (2007) Likert scales and data analyses. Quality Progress 40(7):64

[BLACCSPOT2015] http://www.blaccspot.com/blog/articles/over-1-billion-in-funding-to-webrtc-companies-in-2015/

[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

[BUBLEY2015] http://disruptivewireless.blogspot.com.es/p/blog-page_30.html

[CORTINA1993] Cortina JM (1993) What is coefficient alpha? An examination of theory and applications. Journal of applied psychology 78(1):98

[CRONBACH1951] Cronbach LJ (1951) Coefficient alpha and the internal structure of tests. Psychometrika 16(3):297-334

[GREEN1989] Green TR (1989) Cognitive dimensions of notations. People and computers V pp 443-460

[MOCKUS2002] Mockus, Audris, Roy T. Fielding, and James D. Herbsleb. "Two case studies of open source software development: Apache and Mozilla." ACM Transactions on Software Engineering and Methodology (TOSEM) 11.3 (2002): 309-346.

[OMAHONY2005] O'Mahony, Siobhán. "20 Nonprofit Foundations and Their Role in Community-Firm Software Collaboration." (2005).

[OMAHONY2007] O'Mahony, Siobhán, and Fabrizio Ferraro. "The emergence of governance in an open source community." Academy of Management Journal 50.5 (2007): 1079-1106.

[PICCIONI2013] Piccioni M, Furia C, Meyer B, et al (2013) An empirical study of API usability. In: Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on, IEEE, pp 5-14

[RAYMOND1999] Raymond, Eric S., and Thyrsus Enterprises. "The cathedral and the bazaar." (2012).

[SCACCHI2006] Scacchi, Walt, et al. "Understanding free/open source software development processes" Software Process Improvement & Practice 11.2 (2006):95-105.

[SCHILLING2002] Schilling, Melissa A. "Technology success and failure in winner-take-all markets: The impact of learning orientation, timing, and network externalities." Academy of Management Journal 45.2 (2002): 387-398.

[STDF2011] http://standardsandfreedom.net/index.php/2011/03/17/what-does-community-mean-part-2/