Architecture Design of Real-Time Communication for Organizations with WebRTC

Pedro Vílchez

TFG UPF / YEAR 2015

DIRECTOR/S OF THE TFG:
Miquel Oliver, Victor Pascual
DEPARTMENT:
Departament de Tecnologies de la Informació i les Comunicacions (DTIC)

upf. Universitat Pompeu Fabra Barcelona

Dedicated to my family.

Acknowledgments

Special thanks to Victor Pascual and Miquel Oliver for his mentorship. Thanks to Victor Oncins and Angel Elena for his feedback and help.

Thanks to Daniel Pocock for its work on rtcquickstart.org. Thanks to webrtchacks.com and all its team for the useful articles.

Thanks to all the people that works for the democratisation of communications

Thanks for reading. Thanks for your time.

Abstract

The present project introduces the disrupting technology WebRTC (Web Real-Time Communication), that supports browser-to-browser applications without need of third party plugins. It is detailed how, since its release by Google in 2011, it's evolving and changing the way communications are understood. How to materialise a Real Time Communications in organizations with WebRTC and the use case of video and audio calls, taking as example guifi.net and the opportunities that it offers: requirements, architecture design, component selection, implementation and demo.

Resum

Aquest projecte

Resumen

Este proyecto

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Motivation

In the other bachelor's thesis [Vílchez, 2014] I worked 2 ideas: together we can do networks, it is simple to start. Even so, in cities like Barcelona it is easy to find persons that do not understand what is guifi.net, probably for two reasons: there is a very competitive communications market in cities and the complexity to understand a commons model from the consumer point of view. In the case of zones where there is no investment of large operators, the things change: persons need internet, and they struggle to understand the commons model.

The implementation of other services besides Internet, could help certain persons to understand the advantages of commons network, add value to the network. There are services based on contents that are difficult to maintain but there is a service that can be easily applied on a large scale: RTC. It is something to plan and wait until people start using it. Because is communication in a close environment. We are social.

Also, from the guifi point of view, is a challenge, there were some attempts to implement a RTC system but they failed. With WebRTC, let's try again!

1.2 Objectives

- Free and secure communication between users using own RTC system and commons network infrastructure.
- Backward compatible with VoIP network. Hence, users can communicate to other VoIP operators from inside and/or outside guifi.net.
- An architecture design of an RTC system that fits the community network scenario.

- Ease of installation and use of the RTC.

1.3 Outline

This document is organised in 5 chapters.

Chapter 1 Introduction introduces the project. The motivation to start this project and the high level objectives to achieve.

Chapter 2 Fundamentals of RTC presents the basics of the concepts communications and real-time .

Chapter 3 State of the Art discusses the state of the art of WebRTC and its associated technologies.

Chapter 4 Methodology contains a description of the processes involved to achieve this project.

Chapter 5 Contributions and Results develops the theory and practice of the objectives proposed by this project

Chapter 6 Conclusions and Future Work provides a general assessment of the project

Chapter 2

FUNDAMENTALS OF RTC

2.1 Communications

- **Circuit switching and traditional telephony**
- **Packet switching and Internet**

2.2 Real-time

Chapter 3

STATE OF THE ART

3.1 Important mechanisms

3.2 Voice and video calls

- **VoIP**

3.3 Instant messaging

3.4 WebRTC

- **Security** problems hiding public IP
- Blocks
  - Application
  - Signalling
  - Gateway
    - Media Gateway
    - Signalling Gateway
  - Transport

Chapter 4

METHODOLOGY

4.1 SWOT Analysis of WebRTC

An analysis of Strengths, Weaknesses, Opportunities and Threats (SWOT) will help the decision-making and tasks for the project.

- Strengths
  - Ease of use: real-time communication is supported without the need for additional applications or plug-ins.
  - It helps to solve connectivity problems caused by NAT, Firewall, etc.
  - It solved the problem with audio and video codecs.
  - It is based on open technology.
  - It has well general acceptance in both worlds: enterprise and community.
  - The communication between peers is bidirectional and can be P2P
  - WebRTC standard do not specify signalling: it can be used in very different scenarios.
  - The communication channel between peers is encrypted
- Weaknesses
  - There are detractors of WebRTC, this includes some web browsers.
  - There is a a long discussion about what video codec should be supported in web browsers.
  - WebRTC uses different RTP packets than SIP. So a gateway is needed.

  - Security compromised in VPN-tunnels, where the true IP address of the user can be read.

- Opportunities

  - For VoIP, WebRTC could be a web softphone. Easy to install, easy to update.
  - A WebRTC audio call could be routed to traditional telephony.
  - It uses javascript as programming language, this language has the wider community.
  - It encourages a new generation of web applications using its strenghts.

- Threats

  - WebRTC standard do not specify signalling: this can produce a positive or negative fragmentation of projects. Positive fragmentation: different projects for different applications. Negative fragmentation: divided effort.
  - Is a work in progress technology, it is being changed.

4.2 Scope

There are a lots of RTC systems for different purposes. This project focuses in the work of IETF organization and Internet. Guifi.net is part of Internet, and has additional constraints to take in account.

Dismissed RTC standard systems from ITU[1], referred to in the 2, and XMPP Standards Foundation, small references.

Preferences in terms of technologies chosen, they should be open standard, open source software, with royalty free patent. If this is not possible, it will be notified, and will be used temporarily another technology until a new one appear and substitute it.

4.3 Resources

There are costs related to the activity of this project in terms of equipment and human effort.

Table 4.1 shows the equipment resources and its economic estimation. Observations:

[1] International Telecommunication Union, formerly the International Telegraph Union

- Guifi.net connectivity to Barcelona, a reachable IPv4 10.0.0.0/8[2] has not direct cost.

- Nearly all software involved is open source and has no direct cost.

- Usually the cost of installation it's greater or equal than the cost of equipment.

Table 4.1: Equipment resources

Material	Estimated cost (euro)
guifi.net equipments in my home	200
PC with virtualization capabilities [home]	1000
guifi.net equipments in university	1000
PC with Internet public IPv4 [university]	300
Laptop	400
ATA x 2	60
Old phone x 2	2
Total	2962

The human effort part was financed by the university in the form of a grant to the author, representing a cost of 2800 euro. A bachelor's thesis corresponds in Europe to 500 hours of work.

This implies a total cost of approximately 6000 euro

4.4 Planning

The project can be separated in two phases. The first phase is a long preamble of studying VoIP and WebRTC. The second phase is an agile plan. Figure 4.1 shows the two phases in a gantt chart.

In the first phase, while studying VoIP the intention was to work about VoIP and guifi.net. But Miquel Oliver encouraged me to do it about WebRTC. He presented me Victor Pascual, a SIP and WebRTC expert. It was hard to realise a convenient project, because this technology involves lots of protocols, other technologies, and it's being modified now. In this phase It were settled the necessary concepts to start the project.

[2] range of IP's used by guifi.net and private networks

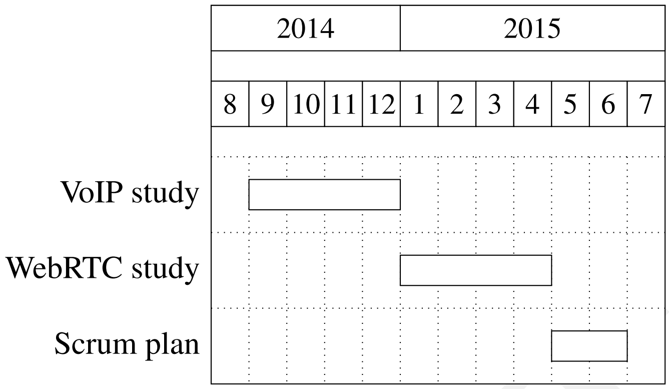

Figure 4.1: General gantt chart

The second phase is an agile plan, inspired by the Scrum methodology. Scrum is one of the Agile methods[3] used for software development. The important fact is that promotes adaptive planning and flexible response to change. Scrum, particularly, is a general method that should be adapted to a concrete scenario.

The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master. The work of the scrum team according to the Scrum Guide[4] is *deliver products iteratively and incrementally, maximizing opportunities for feedback. Incremental deliveries of "Done" product ensure a potentially useful version of working product is always available*. The roles are

- Product Owner: *is responsible for maximizing the value of the product and the work of the Development Team*
- Development Team: *consists of professionals who do the work of delivering a potentially releasable Increment of "Done" product at the end of each Sprint*
- Scrum Master: *is responsible for ensuring Scrum is understood and enacted*

*The heart of Scrum is a **Sprint**, a time-box of one month or less during which a "Done", usable, and potentially releasable product Increment is created*

[3] There are different metodologies grouped into agile. The process started with the write of the Agile Manifesto (12 principles) http://agilemanifesto.org/iso/en/principles.html. Since February 2001, this manifesto remains unchanged.

[4] http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf

4.4.1 Scrum plan

It is necessary to adapt the different concepts that comprise the scrum methodology for this particular project.

Roles:

- Product Owner (in some way, stakeholders): Mentors, University, people interested in the project. The author is interested in the output of the project because is volunteer in guifi.net.
- Development Team: assumed by the author
- Scrum Master: assumed by the author, optionally could be assumed by mentors.

This means that the author has to see the project with different points of view.

The Sprint time is approximately one week, because it is assumed that the minimum time-box possible to do a release of the product is one week. The product comprise two major tasks: the theory (documentation, memory) and practice (how this theory is fitted to the real world experiments). The tasks are explained with more detail in the next section `Tasks`.

Figure 4.2 shows the Scrum plan with the different sprint phases (s1, s2, s3, s4) and important milestones:

- d1: project charter and tasks, delivery to mentors
- d2: first consistent draft memory, delivery to mentors
- d3: set title and abstract to the thesis, delivery to university
- d4: thesis, delivery to assigned tribunal

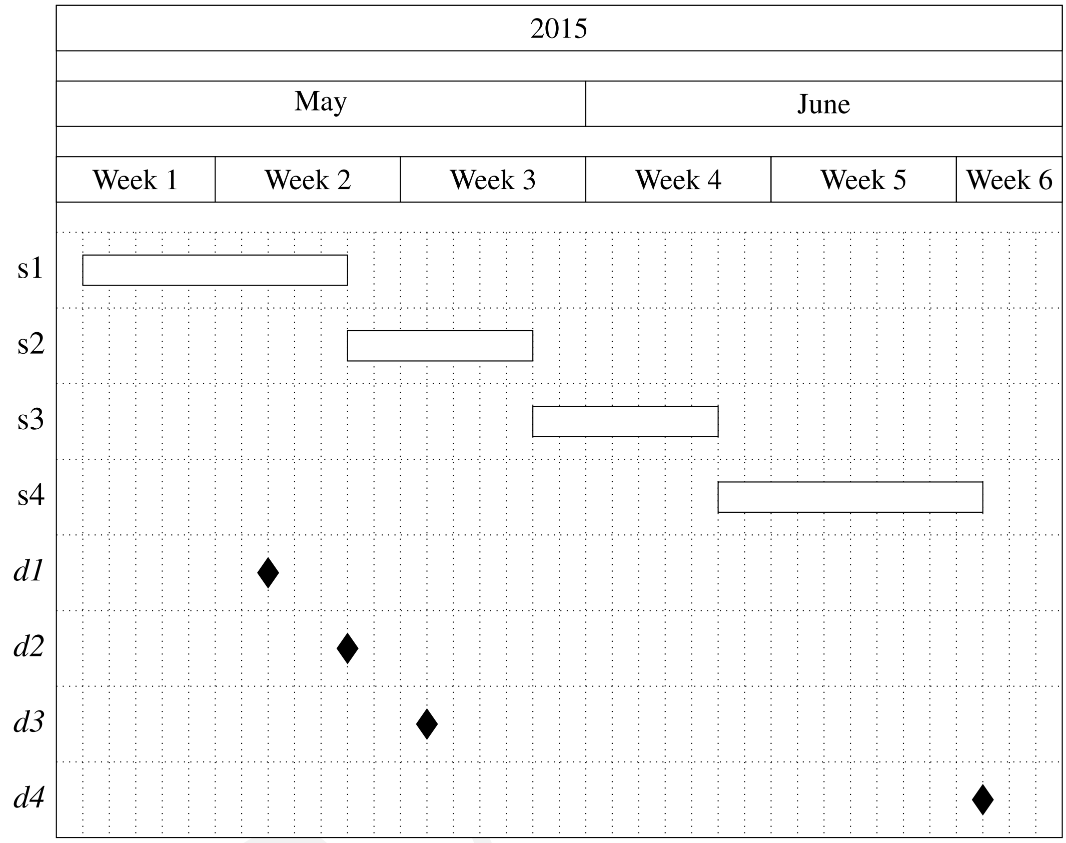

Figure 4.2: Scrum plan gantt chart

4.4.2 Metatools

To ensure the scrum plan and the project, different tools were used:

- Emacs orgmode: is a plain text syntax and software that facilitates different operations
  - nested concepts: It is possible to fold and unfold nested concepts different parts. This brings facilities to take different points of view of the project.
  - write the memory of the project and export to UPF publication constraints.
  - diary: used as autoevaluation tool. Time spent in some operations. Place to record when was discovered something.
  - tasks: to write things to do and mark them as TODO and DONE. To see overall progress of the project.

- Git: is a distributed version control system that helps to ensure the work is not lost. It can has a local and remote copy of all different states (commits) of the project. It is very flexible to do changes and apply.

- Github[5] repository: is a social network that uses git and has the largest community. A place to host and share open source projects. This project is hosted as a repository in `https://github.com/pedro-nonfree/guifi-webrtc`. Featured files:

  - diary.org: record of activity in time
  - tasks.org: parts to do for the project
  - doc directory: independent parts written before starting the memory, or that needs isolation
    - doc/index.org: organise the different files of this directory
  - latexbuild directory: place where emacs orgmode thesis file is exported to latex and compiled to PDF
    - thesis.org: source code of memory
    - thesis.pdf: memory

4.5 Tasks and work style

The tasks for this project are divided in two components: theory and practice.

Inside **theory**, there is:

- Documentation
  - Things to say/explain: what should be said that at the moment is missing (checklist)
  - Parts to Fill: developed parts that are missing few details (checklist)
  - Parts to Fix: developed parts that are incorrect and should be fixed (checklist)
  - Questions: related to the writing or the theory part, that it is needed an answer (checklist, done when answered)
  - Review: concepts that should be reviewed again, after a scheduled date (checklist)

[5] the web implementation is proprietary software, but it can be easily migrated to other open source tools such as `http://gitlab.com` or `http://gogs.io/`

  - Memory document has tools to track the state of different sections. For theory, it will be specially important:
    - Fundamentals
    - State of the Art
    - Result and Contributions

- Search of information
  - What things should be read (checklist).

Inside **practice**, there is:

- WebRTC POC: what WebRTC Proof Of Concepts that have been executed, and wishlist (checklist). What signalling was used. The POCs are web applications that have library linking with signalling. Interested in SIP (jssip) and XMPP (strophe) signalling.

- Tested components: what specific components that have been executed, and wishlist (checklist). LDAP Authentication, SSL/TLS certificates, STUN/TURN server, DNS.

Figure 4.3 shows the **work style**, how the objectives will be accomplished and its quality. Stable means that it should be clear and complete its content; best effort that it will work in the best way possible but with less priority:

- Requirements: use cases, constraints needed for the chosen organization. (quality: stable)

- Design: arquitecture design that fits the requirements. (quality: stable)

- Implementation: component selection, protocols (quality: best effort)

- PoC: applications that shows some of the results (quality: best effort)

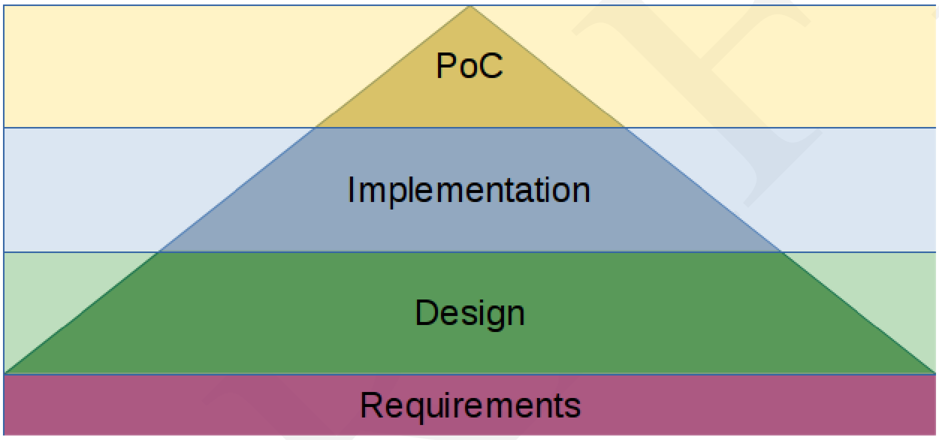

Figure 4.3: Work style diagram

Chapter 5

CONTRIBUTIONS & RESULTS

It is wanted to define the architecture of a general RTC service, as set of different services and applications.

5.1 Architecture of guifi.net

5.2 Requirements

5.2.1 Generic use cases

Definitions:

- Actors or Roles have a **user** and/or **admin** account. The user has minimum permissions in the application, the admin has all the permissions in the application. It can be defined one or more middle actors that have more permissions than user and less permissions than admin.
- A **call** refers to an audio or video call, bidirectional communication with video and/or audio channel.
- A user is **available** if is connected to the service and is not in a call.

A general RTC service could be defined as it follows:

1. **Send calls: a user call another user with an audio channel. Optional channels of communication if available: video and chat.**
    - Access to the service through an application.
    - Authentication.
    - Decision of which user it is wanted to call.

- The call is accepted by the other user.
- Bidirectional communication.
- One of the two users stop the communication.

2. **Receive calls: a user receive a call only if is connected to the service with at least one device and is available.**

    - Access to the service through an application.
    - Authentication.
    - A lapse of random time until it is received a call. If there are more devices of the same user, all of them receive the call, but only one can accept it.
    - The call is accepted.
    - Bidirectional communication.
    - One of the two users stop the communication.

3. A user can subscribe or unsubscribe to the RTC service.

4. Call history: a user can see the calls sent, received, missed. He can delete it.

5. Missed call notification

    - When a user call another user that is not available (not connected to the service or in another call). A ”missed call” notification is generated to be delivered to the other user.
    - A user has been notified by a missed call if the device is compatible with this service and he is available.

6. Contact list

    - A user can see the status (online, offline, busy, etc.) of another user in its contact list if he is allowed by the other user.
    - A user can add or remove another user from the contact list.

7. Chat rooms

    - A user can be in a public place where there are rooms and people talk openly.
    - A user can speak privately to the users connected to this place.

- The identity in the chat room is the same as in the contact list.

8. User preferences

   - User can set its own photo, nickname and description.
   - Users can set if a room is able to record a history conversation (and files) such that users that connect and disconnect can follow the conversation.
   - User can change password of its account.

9. Share advanced media

   - User can share its screen with another user.
   - User can share files of limited size in a room or privately to another user. The data is temporarily stored.
   - Share N streams to N users (Multiuser bidirectional videoconference).
   - Share one stream to N users (Streaming).

10. Administration

    - User can only change its settings. Admin can change configuration of all users.
    - Users can report other users because of a social conflict, admin is notified.

11. **Integration: all the services are integrated and is the same account.**

Guifi.net service is defined as it follows:

1. A user can connect to a server if he could reach it with good quality, if not, he can easily install it in its zone.

2. If a server reach another, the users of a server can communicate to the users of another server.

3. The service is compatible with VoIP guifi.net project.

5.2.2 Component requirements

The use cases developed are those that are bold.

Requirements to send, receive calls, subscribe, unsubscribe and have integration. For a WebRTC scenario, the components are distributed. In brackets, name of the components in the component graph.

- Application interface: gives appropriate interaction to the actors in order to perform the different operations. Is distributed, a web server (WSRV) offering a page, and a client executing the web application (WAPP) through the web browser.

- Authentication service (AUTH): restrict access of service only to permitted users. Differentiate available operations depending on if is user or admin. Single sign on, after the web page is accessed, the user can operate.

- Signalling protocol (SIG): manage the side to side connections and logic to establish the call. The two peers must use a compatible signalling protocol.

- Transport protocol (TP): between users and between user and server. Compatible, secure if possible.

- Database (DB): store and encrypt personal information or preferences for a particular user. Accessible through web application if succeed in authentication.

- Connectivity solver (CS): a set of tools to avoid common communication problems that appear in networking scenarios. The most common problems are NAT and firewall.

- Gateway (GW): adapt or convert the communication to work with different communication systems. The most important difference between communication systems is the signalling protocol.

5.2.3 Network requirements

number and math justification?

- DNS (multiple servers)

- Quality of Service (QoS): put priority to real-time traffic

- Throughput (more used than Bandwidth)

- Delay

- Jitter
- Loss
- Congestion

5.3 Component selection

5.4 Architecture design

general diagrams

Each link has associated transport protocol

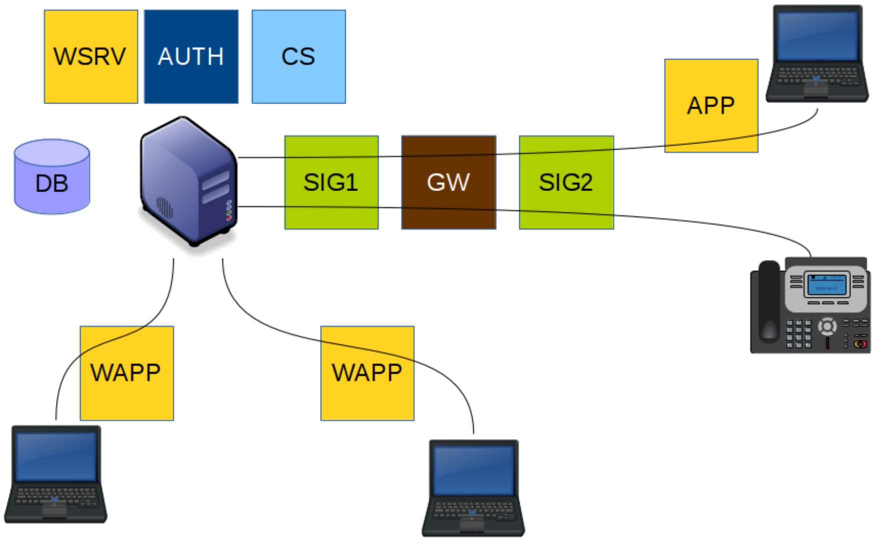

Figure 5.1: Component Diagram

flow chart communication

5.4.1 WebRTC to SIP case

Authentication *authentication diagrams*

Gateway *gateway diagrams*

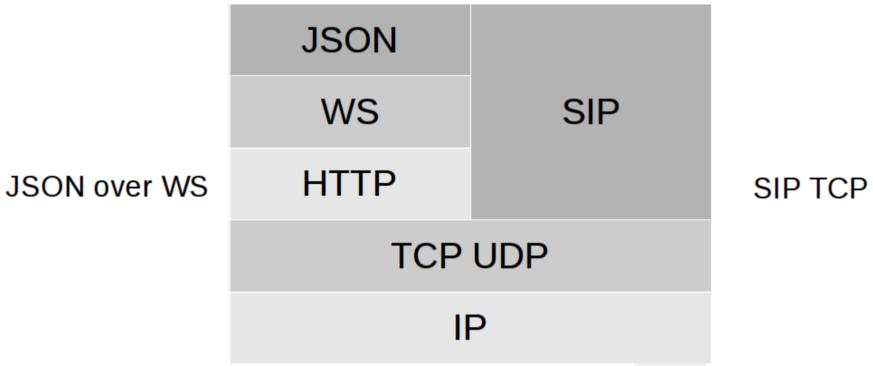

Figure 5.2: SDP exchange during SIP signalling

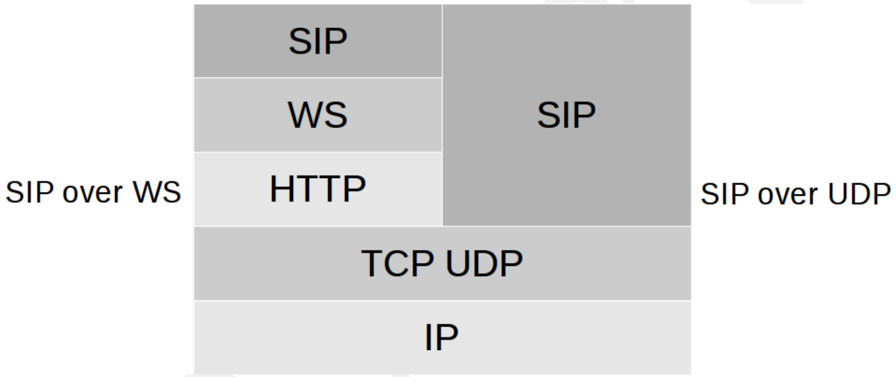

Figure 5.3: SIP transport

5.5 Applications available

5.5.1 POCs

5.6 Implementation

5.7 *Demo*

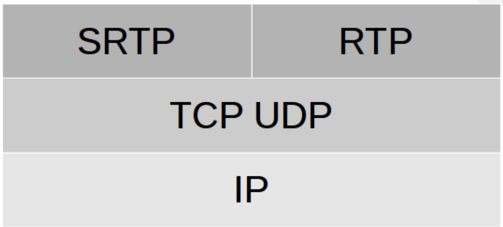

Figure 5.4: Transport

Chapter 6

CONCLUSIONS & FUTURE WORK

6.1 Conclusions

6.2 Future Work

Chapter 7

FAKE

[To be removed]

Fake section to put the last latex code of the document

Bibliography

[Vílchez, 2014] Vílchez, P. (2014). Starting, contributing and empowering community networks in cities. Bachelor's thesis. Universitat Pompeu Fabra (UPF).