FLINDERS UNIVERSITY

COMP-9710A Master Project Thesis

by

Theo DE FRAMOND

Functional testing and qualification
of the Serval Mesh Extender. Submitted to the School of Computer Science,

Engineering, and Mathematics in the Faculty of Science and Engineering in partial fulfilment of the requirements for the degree of Telecommunication at Flinders University Adelaide Australia.

Supervisor : Paul GARDNER-STEPHEN

June 2, 2017

Declaration of Authorship

I, Theo DE FRAMOND, declare that this thesis titled, 'Functional testing and qualification of the Serval Mesh Extender' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

"Communications should not just be for the geographically, financially otherwise fortunate — for it is the unfortunate who need it most."

Serval

FLINDERS UNIVERSITY

Abstract

School of Computer Science

Master Project Thesis

by Theo DE FRAMOND

This paper is the thesis of my Master Project COMP9710A in Flinders University, School of Computer Sciences as an exchange student from France. It explains all the work, tasks and experiments I have done during my project. The remainder of this content concerns the perspectives and the motivations of such a project and also what I have retained about it.
My project takes place in the humanitarian and open source project called Serval. Serval Project purpose is to providing means of communication to people in the incapacity of having cellular coverage. To this end they have developed a smart-phone application and a special smart antenna called the Mesh Extender. My role here is to write automated tests to make sure these Mesh Extender are ready for their next use.

Keywords : Serval Project, Testing, Mesh Extender, Emergency Communications, Automatic Tests, Servald, Expect, Master Project.

Acknowledgements

I would like to thank the all telecommunication laboratory of Flinders in Tonsley for their participation in the project and for having supported my work and helped me get results of better quality. I am particularly grateful to my supervisor Paul Gardner-Stephen for his patience and support in overcoming numerous obstacles I have been facing through my research. Thanks him for having accepted me in his team.

I would like to thank my fellow students and the ISS of Flinders University for their feedback, cooperation and kind welcome.

Nevertheless, I am also grateful to my French school INSA Lyon for having given me the opportunity to get here and work in this wonderful country which is Australia. Thanks Isabelle Auge-Blum and the DRI for having organized my student exchange and thanks the jury for having changed their decision regarding this exchange which was at first forbidden for me.

I would like to thank my friends for accepting nothing less than excellence from me. Last but not the least, I would like to thank my family: my parents and to my sister for supporting me spiritually throughout writing this thesis and my life in general.

...

Contents

List of Figures

Chapter 1

Introduction

This chapter of my thesis is the "why?" of my project. I am explaining here what are my motivations that leaded me to this and why i wanted to chose this work as my end study project instead of an other one. The scope and the context of this experiment are also part of this chapter. The two last points i am discussing here concern all my research questions and the global structure of my thesis.

1.1 Motivation

1.1.1 Australia

Of course and since I live in France, an important motivation was to travel and discover Australia. But as this "touristic" motivation is not very appropriate in this thesis I will not develop it that much. But I still want to say that traveling is essential for a human to grow and get mature. This apply for becoming a engineer as well. Now that the semester is almost finished, I can tell that this abroad experiment has helped me to get prepared for my future life by making me think in a different way and more.

1.1.2 Telecommunication

Second motivation is the field of telecommunications. This is my studies and what I am interested in. It was thus obvious that this project will fit to me. Indeed, the Serval Project concerns an emergency phone network. It deals thus with antennas, communication protocols and signal processes. All this topics will be helpful for my

future jobs. All what I learned in the past years in my school, I can apply it here on this project which is totally on the continuous of my studies track.

1.1.3 Laboratory and Research

In my five years of study, I have never really worked in a laboratory as a researcher. I have done many tutorials and practical works with my class but it was just study and nothing to see with research. I wanted thus to discover the world of research and how all of this work. How is running a laboratory ? Who is in charge of it ? What are the relationships in it ? How are the projects financed ? I am really happy now to know much more about this part of the engineer world.

1.1.4 Humanitarian

Last and not the least, I really appreciate the humanitarian aspect of the project. Indeed, as I will explain in the next section, the Serval Project is here to help poor islands from the pacific to recover after a natural disaster for example. It can also be the first step for poor villages to have access to distant communication without having to invest into expensive and permanent infrastructures. If I can work for a good and human purpose, it is better. I am really happy to help in this adventure rather to work for big industries who just care about money despite the environment or human conditions.

1.2 Scope

The Serval Project is a suite of technologies designed to facilitate and sustain mobile telecommunications in the absence of supporting infrastructure, such as cellular networks or electricity.

The two main components of the Serval Project are the Serval Mesh Extender Hardware and the Serval Mesh App. Basically the Serval Mesh Extender is a low-cost communications relay device that extends the range of communications among phones using Wifi technology. The laboratory has a partnership with my university in France named INSA de Lyon so that each year, French students can help on the project as a one semester exchange program. I am one of these student.

For the year 2017, the Australian Department of Foreign Affairs and Trade have commissioned the University to pilot Serval in the Pacific. Consequently, we have to prepare

the Serval Mesh Extender technologies for field use in tropical-maritime environments, and without any dependencies on mains electricity. To this end the first Serval Mesh Extender is being redesigned to satisfy these requirements. However, this process is not yet complete.

Therefore there is an need to devise and apply a testing regime for the new Serval Mesh Extender design, to ensure that it meets the necessary functional requirements, and that the units are possible to easily manufacture. The focus of this project will be on the creation and application of such test protocols, to ensure that the Serval Mesh Extender devices are ready for deployment in the field pilot.

In the end Serval propose a solution to the problem of having difficult access to a communication network (or help) within certain tough conditions. However, solutions like these are useless if they are not working properly. It can be even worst to have a non-functioning system rather than none. The importance of automating the testing is to quickly know if a device is ready or not on the field and thus avoid time loss in problems fixing that can be really annoying while installing the network. Automatic tests are here to see further and prevent bugs from bad manufacturing.

1.3 Research questions

For this work I am following an Agile methodology. Therefore I have to organize my experiments in an iterative way and plan them with precise steps and precise goals. A list of the automatic tests I have to implement will help me with this.
The tests must be automatic and quick. That is why I need to implement them in a logical and low-level software environment. Basically I will connect the Mesh Extenders with a laptop and then run all the tests on it. In the end, it will display all the results so we can know if the extenders are ready to use or if we have to do some changes on the settings and if the manufacturing of the extender is okay.
Finally and with the help of my supervisor instructions, I used Expect and Shell scripts to code my test. These are the different parts of the Serval Project I will have to implement tests for :

- Mesh Extender hardware
- Mesh Extender cables
- Mesh Extender software
- Mesh Extender network functions

- end-to-end connection testing with various topologies
- manufacturing quality control
- acceptance testing

Since we have to be ready for the Vanuatu expedition which will happen in May, work must be done in two or three months. That makes thus two or three tests by months. According to the fact I have to see and learn how the Mesh Extender operates and the callback I need in Expect and Shell language.

1.4 Structure of Thesis

The structure of my thesis will follow a height points pattern. After this introduction, I'm going to do some literature review in order to get the context of the Serval Project and to know exactly where about we are in it. I will deal with the different versions of the Serval Mesh extender and introduce the new design of the last version. Then, the methodology will be an other part follow by the Pacific Humanitarian Project which is the reason why we can run this project at this level now. The next big part is the hardware tests split into two categories : the Environmental Testing and the Post-Assembly testing. The first one concerns the abilities of the Mesh Extender to run into a certain environment, for example rain or dust. The second one is more about the software and utility requirements. Next part is obviously the software tests. I will finally present the simulated field operation of the Mesh Extender before discussing all the project and to conclude introduce the future direction of the Serval Project.

Chapter 2

Serval

2.1 What is Serval ?

2.1.1 The Serval Project

Serval is a humanitarian open source project aiming at providing means of communication to people in the incapacity of having cellular coverage. "Cellular communication is great when it's available" but it is not always the case. This is why Serval offers an independent and autonomous cellular network which works with mesh extenders and normal smartphones without the need of external cellular infrastructures.
This project was created in 2010 by my supervisor Paul Gardner-Stephen and Romana Challans in response to the terrible and disastrous earthquake that happened in Haiti just before. Years after years, many people join the team to work and help on the project which is based in the telecommunications laboratory of Flinders University in Tonsley, Mitchell Park, SA.

FIGURE 2.1: Serval Logo

2.1.2 Why Serval ?

Serval is here to help with one main purpose: COMMUNICATE, regardless the conditions and circumstances. [1]

- Communicate anytime, even when the usual and main phone network is down or the cellular infrastructures are broken.
- Communicate anywhere, as cellular networks are not available everywhere. For instance, 75% of Australia is not covered as lots of poor countries in the world. [2]
- Communicate privately, with end-to-end encryption for phones calls and SMS.

2.1.3 How does Serval work ?

Basically people can use the service as a usual cellular network in calling or sending SMS to each other. The usage differences are that they need a special application to do it and that all these communications are not instantaneous. A message can take either two seconds or two days to reach its destination depending on the situations and locations of the users. The practical difference is that they do not need cellular infrastructures. They can communicate in a desert and uncovered environment or even after a natural disaster when all communication means are down.
Serval is composed of two main components : The Serval Mesh application and the Serval Mesh Extender device.

2.1.3.1 The Serval Mesh application

To be able to connect to the Serval communication network, the user has to posses a smartphone with this special application installed on it. The application is currently developed by Jeremie on Android but will be soon available for iOS. The reason why Jeremie does not want to migrate to iPhones is that you are not allowed to install untrusted and third-tier applications without a jailbreak version on iPhones. And this is obviously important for Serval since the goal is too deploy the network in case of natural disaster, e.g. when there is no data connection to download the application on the official store. In this situation the application will be downloaded directly on the Serval network or on a other device, which is not possible with iPhones because of its restrictions.
Serval Mesh allows you to send SMS (MeshMS), calls, updates or even allows you to ask for help to other connected devices and contact a help desk. If you do not have the application installed, you can still connect to the Serval captive portal served by the Serval Mesh Extender and do similar tings as with the application but with a less rich experience.

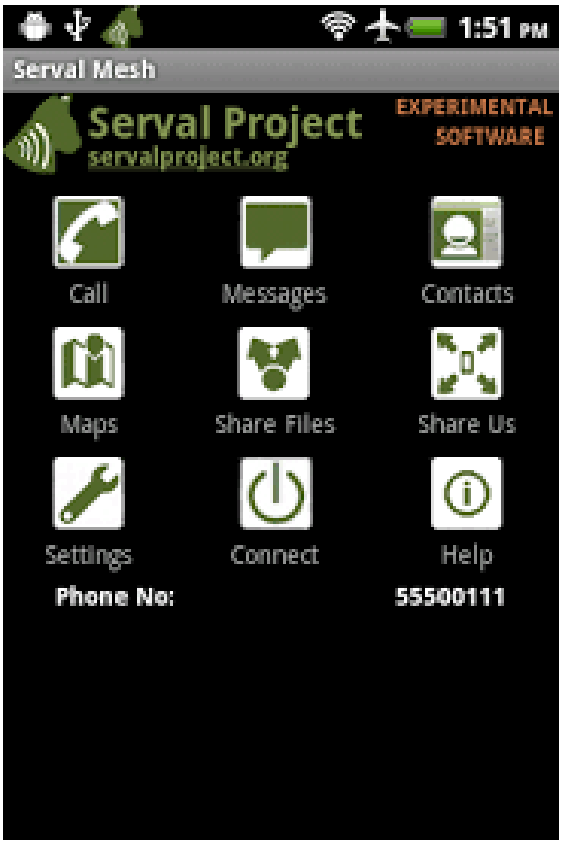

FIGURE 2.2: Mesh App screenshot

2.1.3.2 The Serval Mesh Extender

To be able to establish communication between users, they need to interact from smartphone to smartphone but only with the wifi ad-hoc connection. [3] Sometimes the range of these wifi are not sufficient to create an efficient cellular network. That is why Serval created the Mesh Extender which is a device designed to extend the range of the network with UFF radios. It can extends the range of every device nearby thanks to his two radio antennas and wifi antennas. With the Mesh Extender, the Serval Network can finally be used like a traditional cellular network with minimizing range problems. Basically smartphones connect to Mesh Extender with normal wifi at short distances while Mesh Extenders can talk to each other at long distance using the UHF. This is how we can establish contact between two distant smartphones.
This device is made on a PCB base including both RFD900X + Atheros 9k based embedded Linux computer, a wifi antenna, two UHF radios antennas, an SD card slot, a USB slot, an Ethernet port and a serial port.

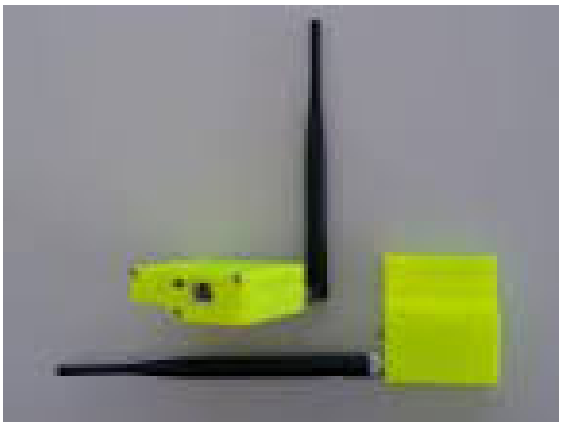

FIGURE 2.3: First version of the Mesh Extender

2.1.3.3 The protocols

The main mesh-based data distribution protocol developed by Serval is called Rhizome. It works like a store and forward file distribution system. It operates on data packets containing meta-data and of course the data. Rhizome, as a mesh network protocol, works without centralized infrastructures. It guarantees the integrity of the message payload during the transport but does not guarantee the time and the delay of transportation as mentioned before. It can also define which packet is more important to transfer if there is a queue according to its size and his arrival time. It is probably not expected but the shortest messages are actually here considered like the most important ones. This is Serval philosophy. [4]

The core component of the Serval mesh networking products including the app and the extender is called Serval DNA. It contains all the technologies needed such as the Mesh Datagram Protocol (transport), the Voice over Mesh Protocol and of course Rhizome. [5]

LBARD which stands for Low-Bandwidth Asynchronous Rhizome Demonstrator is the name of the transport protocol developed by Serval which is used between two low-bandwidth interconnected nodes. For example, two Mesh Extender which are communicating through the UHF radios. This protocol is running between to Rhizome instances and help it to synchronize. We need this because Rhizome was previously designed to operate on Wifi connection. Therefore when we use a narrow bandwidth channel, the data do not behave the same way at all. That is why they needed to develop a more appropriate one for slow data transfers. As Rhizome it is fully encrypted. [6]

2.1.4 Serval purposes

As cellular network is not everywhere and always accessible, the Serval Project is about creating an alternative communication network that does not depend on a big, official and expensive infrastructure. That means they do not need a license neither nor a carrier. In this conditions, Serval is using a Mobile Ad-Hoc Network (MANETs) which is appropriate for this.
An other goal of the Project is to connect sensors to this network in order to monitor wide remote area. For example, a farmer wants to know if a door is open on the other side of his huge farmland. By connected sensors to the Mesh Extender of the network we can in addition to the cellular network, create a sensor network. Of course we can imagine lots of other use to this network as far as it does not need a big amount of data speed.

2.1.5 Serval's funding

In the past years, Serval had several partnerships which include the Shuttlerworth Foundation, NLet Foundation, the Awesome Foundation for the Arts and Science, the New America Foundation, Internews, Open Internet Tools Project, Flinders University of course and many other institutional and private contributors. But this year is a special year since the Serval Project has been nominated by the Australian Government in the Pacific Humanitarian Challenge which I will explain in the next point. The Australian Department of Foreign Affairs is thus one of the more important funder this year thanks to this challenge.

2.2 The Pacific Humanitarian Challenge

2.2.1 The challenge

In November 2015 the Australian government called on innovators, entrepreneurs, designers, NGOs, and academics to rethink humanitarian response. They received 129 applications from 20 countries across five continents. Ten First Round Winners were selected to attend a Design Sprint in March 2016 where they further refined their applications with coaching from advisors. After much consideration, five teams were selected to share in a 2 million AUD fund for running pilot projects in the Pacific. Serval is part of them.
As a consequence, they have to pilot a first experiment of this project in Vanuatu in May 2017 to test and qualified their work. If this is a success, Serval Project will have

FIGURE 2.4: Pacific Humanitarian Challenge

more help from the government in the future to make the project evolve further.
To this end, there is a need to develop and increase the Mesh Extender so it will be ready for the pacific conditions and can handle it.

2.2.2 Mesh Extender, second version

The first version of the Mesh Extender was not ready to face the conditions of the pacific challenge. Therefore there was a need to prepare it for this. By this, i mean the tropical climate, the heavy rains and the lack of electricity in some places. For this points we need to improve the Mesh Extender in order to make it resilient against the rain, the dust and the wind. Moreover we also have to make it autonomous in terms of energy.
For the first issue, Serval visited the world of injection moulding with a colleague from the university. Together with a local injection moulding company, they have found a way to get full-custom polycarbonate injection-moulded cases designed and manufactured at a relatively affordable price. With this, we have now an IP65 or IP66 rating for the case, so that it can be safely used in dusty outback conditions, as well as in tropical maritime climates.
The second issue has been fixed with the interesting idea to include a solar and battery controller in the unit. We can now just plug in a solar panel, car battery or other supply to run the unit, in addition to the normal 5V USB supply. We are also able to connect two LiFePO4 cells, that the unit would charge from the supply, and use to operate when there is no power supply available.

FIGURE 2.5: The new polycarbonate injection-moulded case

2.2.3 Scope and project overview

With this mission in the Vanuatu, we expect the manufacturing of maybe one hundred new Mesh Extender in order to install them in the field pilot. Therefore there is a need to test all of them at the reception to be sure they are functional and ready for the Vanuatu experiment. That is why I will have to develop manufacturing test in addition to the software tests.

To realize this master project, four and half months were allocated. Given the complexity of the project, the understanding of different aspects of the project was not immediate. Indeed I had to learn lots of different things before starting the project for real. And this learning period was way longer than expected. It took me maybe two weeks just to understand how Serval exactly work. Then I had to learn how to use Latex, bash script and expect script as well. The setting of the work environment has also taken a bit of time as i had for example to change the hard drive and the RAM of my laptop since he was about to die at anytime.

Chapter 3

Literature review

3.1 Emergency calling system

After a disaster or any incident, people are in a need of emergency services. They are supposed to be allowed to call emergency phone numbers for free but this is not always the case. Imagine a serious tsunami that break down all communication means. Even if these numbers are free charged, you will not be able to call them because cellular networks will not be operating. This is what happened in Haiti 2010 for example.

FIGURE 3.1: Tsunami in Haiti, 2010

Internet is an other way to try to call for help in case of emergency. During terrorist attacks or disaster, Facebook allows people to mark them as safe to let their friends and family know how they are going. But if the cellular network is down, there is a chance that internet connection is also compromised.
This is where Serval network takes place and overcomes these problems by proposing alternative and autonomous communications means.

3.2 Competitors projects

Obviously Serval is not the only one on the subject, there is other projects working on similar topics. Here are some of them. I call them competitors here but there a not really in competition since Serval is more a research and humanitarian project than a business.

ElectroSense Technologies. They sell products for domestic, scientific and commercial purposes. These allow people to monitor and control equipment remotely. It uses SMS, 3G and UHF to access data such as water level, soil moisture or state of electric fences. [7]

GoTenna. Really similar to Serval, they offer a device that extends the range of your smartphone and make you capable of communicate with distance without external infrastructure. This is more focused on leisure activity though. [8]

Farm Monitoring Solutions. This is basically the same kind of technologies but apply to farm activities such monitoring water level, power, stock and equipment remotely. They also have a captor powered with solar energy. [9]

Observant, Ranch Systems and uSEE. Very similar to the Farm Monitoring Solutions and maybe the first rivals in terms of agriculture purpose. [10]

Beartooth is probably the number one competitor of Serval. This deals with precisely the same aspects than GoTenna but is a bit more famous. Their slogan is "Because life does not stop where the network ends". [11]

3.3 Mesh network

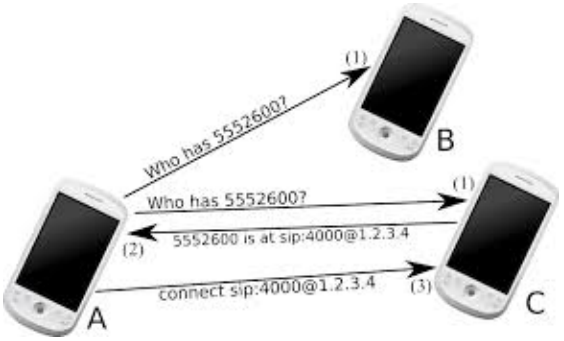

FIGURE 3.2: Example of a small mesh network

Mesh networks are networks with a particular topology. The nodes do not know the whole network but just their direct neighbors. Each device relays data for the network.

Packet goes from node to node with as many hop they need to reach the desire destination. This is not centralized. To shut down the network, it is then necessary to turn off every devices of the network. This is why we can say that mesh networks are resilient. Serval is using this technology so they do not have to use the usual cellular networks. The network is therefore created by every devices running the Serval Mesh app and the advantage is that they do not need any external infrastructure to communicate. The disadvantage is that the network can sometimes be too small or incomplete. If you want to reach someone who is located too far from you and there is no ones in between to relay the message, this person will not receive the message. Or at least not as far as there will be a gap in the mesh topology. That is why Serval can not promise an instant delivery messaging experience. But this is not the main goal actually.

3.4 Press

Paul and his project has been mentioned many times in fews articles since more than 5 years ago. For instance in Indiegogo.com, MakeUseOf.com and dev4x.com. With the Pacific Humanitarian challenge, they also had a boost of notoriety. We can even notice that Serval appeared on a French Android-dedicated website and on a famous French newspaper called Le Monde. This is abroad communication is probably due to the partnership Flinders University has got with my school INSA Lyon. Serval also have lots of contacts with New-Zealand according to the Red Cross and now with the Vanuatu as well since this year.

FIGURE 3.3: Paul Gardner-Stephen presenting

Chapter 4

Method and Material

4.1 General Functioning

4.1.1 Management

My master project is a bit special in a sense that it is not my personal project. I chose to join an existing one. That means that I have to follow the rules who already run the telecommunication laboratory where I work. At the moment we are a dozen of students/researcher working here on either independent projects either some linked project. As far as I am concerned, Paul and Jeremy are working on the Serval Project as well but I am the only one student working on the tests this semester. Usually they are many French students as the same time and they can therefore help each other, which I can obviously not since I am the only one.

Paul is the laboratory director. To keep himself updated with all the projects of his students and fellow, he organizes every Monday a "Lab Meeting" at 2.00pm. Everyone has to be here in order to make a brief summary of its work progress by presenting a three point report: "what I did last week, what I will do next week, do I have difficulties ?". This helps him to know the ones who are late, blocked or unwilling to work and the ones who are motivated.

There is no really strict rules concerning the management of my project. Paul just let me know what he was waiting for and what did I have to do. I am quiet free to handle it as I want as far as I finish the work. Therefore I decided to just take care of one task after the other by following a procedure which I will explain in the next section of this chapter.

4.1.2 Means and documentation

As a master student, I can access the laboratory from 6am to 10pm all the week with my student card. In there, I can use lots of different equipment like laptops, screen monitors, tools, phones, printers, antenna ...etc. Some documentations is also available and blackboards help us to brainstorm and through quick drafts on them.

The Serval documentation can be found in three different places: internet, papers and local pdf documents.
On internet we can find first the Serval website which is kind of public access to the project with simple and "business" explications. Second we have the Serval wiki which is the developers documentation side of the project. It explains everything with more technical details. As Serval is an open source project, this wiki is quiet exhaustive and obviously completed by a Github repository where all the codes of Serval is stored and totally public. Everyone can access it and add some modification to it. I spent quiet a lot of time to read the wiki in order to learn how Serval work and I also had to clone some depo from the Github to install and work on it. Finally, Paul owns a blog where he posts every step of the project. It may be the place the more up to date and the more illustrate with pictures. This is where we can follow the adventure in real time. A Google group also exists but it is not very active. I tried to start one discussion about my project but got no response.
Finally, the best way for me to find informations was face to face discussions in the laboratory. Although the community is almost inexistent and inactive on the web, Paul and Jeremy were very helpful in the lab and were happy to answered my questions. Nothing is better than real human contacts. It was at beginning really hard for me to understand my fellows because of their strong Aussie accent and their fast speaking. Added to the fact that I did not know anything about Serval and the technologies I had to use, it was really hard to jump in the bath. I had to ask them to slow down a bit for me because I could not understand anything. After five months in Australia, it is now easier for me to understand them.

4.2 Test methodology

My work methodology is following an Agile method. Every tests can be considered as a single task or a user-story such as my supervisor can be considered as the scrum-master. Each time I chose a new ticket, I try to chose a deadline to fix the ticket according to its complexity. Obviously it is not often easy for me to respect this delay since I am not familiar with the technologies I have to use here. For example, a really simple task can

take me a huge amount of time to execute because of my lack of knowledge in these new languages.
The test methodology is not completely defined and can be different depending on the test himself. Nonetheless, the global aspect follows the same basic pattern every time.

4.2.1 Test technology

4.2.1.1 Bash

As Serval project use lots of different languages and devices, we can not use a normal test frameworks like JUnit for example. Indeed we are not doing a test driven development or just testing codes. My goal in this project is to test the whole thing, not just some codes. This is why I need to use something more general and less specialized than a test framework. To this end, I will use basics shell scripts which are very useful to communicate with the Mesh Extender since it is running on a Linux kernel. Bash is the most famous and simple shell language that I can use. It allows me to call directly lots of Linux command and high level instructions. This is very practical with what concerns Wi-Fi connections for example.

4.2.1.2 Expect

Expect is a tool for automating interactive applications such as telnet, ftp, passwd, fsck, rlogin, tip, etc. Expect really makes this shells interactions trivial. It is also useful for testing these same applications. And by adding Tk, we can also wrap interactive applications in X11 GUIs, but this won't be useful in my case. Expect can make easy all sorts of tasks that are prohibitively difficult with anything else.
I was first thinking that it was a useless tool but after using it, I realized that it was not the case at all and that it really helps with my tests. Moreover I even found myself using expect for others purposes like automate my thesis save on Github. Now, with my simple Expect script I can add, commit and push my thesis on the repository with just one script without having to enter my Github password or commit message.

Expect is based on the TCL language, so I had to learn this one as well before starting to use it. Expect add some new command to TCL that are really helpful for automation like the commands : spawn, expect and send. Basically and the most easier way to understand it is as follow. First you start and open a shell command with the "spawn" command, then you wait the shell to respond a particular message with the "expect" command and you react according to this respond with the "send" command which actually just type something for you automatically in the shell.

4.2.2 Test overlook

The first task to do before each test is to sort it and classify it between hardware and software categories. This also helps to chose which language will be more adequate to test it and in which environment I have to code it. Sometimes just a bash script will be enough to reach my goal, sometimes I will need an expect script to go further. Sometimes expect does not react like it is supposed to with the Mesh Extender, and I have thus to fix this issue by just using the shell.
This is also the phase in which I will talk with my supervisor Paul in order that he explains me all the aspects of the tests and what are his expectations. This is a kind of user story creation where we establish the goals and the tasks to implement in the test.

4.2.3 Tools

To write my tests I just need a text editor and the terminal. My favorite text editor is Sublime Text. It is very light and helpful with everything. I use it to write either my thesis in Latex or either my Expect and Bash script. It has an auto-completion feature and an integrated spell checker which is really interesting for my thesis writing.

Before getting started with the development I had to install my environment. Serval OpenWRT which is a light Linux distribution used by the Mesh Extender was the first package I have to install on my laptop in order to create and export the Linux disc image on the PCB. I also had to install Expect of course and many different other tools like for instance Airport which offer a wifi scan service from the shell which will help me with testing the wireless connections.

I have a personal Github repository to save and share all my thesis work. Here is the link : https://github.com/tdeframond/mesh-extender-provisioning .

I also created a discussion on the Serval Google group about my work to try to interact with other member of the project : https://groups.google.com/forum/#!topic/serval-project-developers/qUYktshW1x4 .

4.2.4 Tests pattern

My tests are just scripts written as follow : one small script per test so it is easier to code and organize them. Each Expect script is therefore a text file with the name of the test and an ".exp" extension. These Expect scripts can make use of other small Bash script for special tasks.
There is no input for the moment, I will have to implement that later. If the script needs some inputs, It have them directly in its code. The scripts are thus not really instantaneous adaptable to different Mesh Extender. For example if the test need the mac address of the Mesh Extender, I will give it to him in the code. In the future it would be better that the script can retrieve it automatically.
I design all the outputs with the same graphic charts in order to have a unity in the tests. This is a really simple design pattern made of # and ¿ but it is simple, clear and easy to exploit and understand. These outputs keep us knowing how is the test going throughout the whole testing and display the final results at the end. Each specific test has its own and separate output but the goals in the end of the project would be to join all this small tests in a bigger one so we can launch all of them with one command and see all their results in a one place.

Chapter 5

Functionnal Hardware testing of Serval Mesh Extender

5.1 Test 1 : Boot-loader and flash firmware update

This is the very first task I have to implement. The goal is to prepare the manufactured Mesh Extender units for the tests and use. In order to do it, we have to install a special Linux distribution on it called OpenWrt. OpenWrt is an embedded operating system based on Linux, primarily used on embedded devices to route network traffic. All components have been optimized for size, to be small enough for fitting into the limited storage and memory available in home routers as the Serval Mesh Extender.

In order to install the latest version of the OS, we will use here the command-line interface Shell. This will allow to boot directly the Domino from a laptop. But at this point, we need to make these two entities capable of communicate to each other. That is why we need cables and especially one that can deals with the extender serial port. For this one, we have for the moment a home-made cable that use a D-SUB 25 pins male port on one side and whatever we want on the other side by cutting the edge of the cable and weld components on the desired nude wires. That is why we have welded here a serial port to USB adapter and a power plug to power the PCB.
So now we can talk to the PCB with the laptop through command lines. But there is a problem remaining. Serial port is too slow to transfer data and at some point, we will have to download on the Domino the OpenWrt software binaries. That is why we also use an ethernet cable to install the distribution in addition to the serial cable.

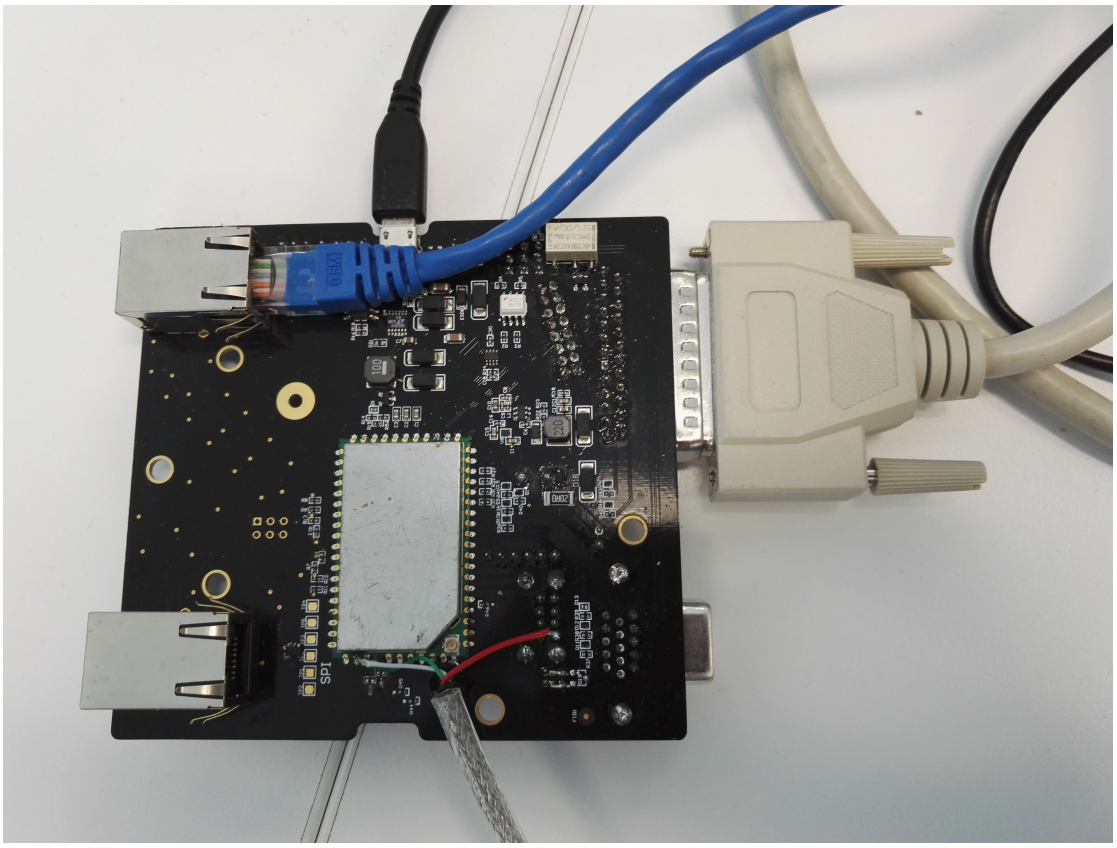

FIGURE 5.1: The PCB of the new Mesh Extender

5.1.1 Procedure

The procedure is simple, we first connect the laptop and the Mesh Extender with the serial and ethernet cables. Then we plug the power cable in order to make the PCB run. We have to install the Serval software which create the OpenWrt files to transfer to the PCB on a linux environment. To this end, we have to install lots of dependencies like for example GNU awk, SVN, OpenSSL library ...etc. That is why we need to install VirtualBox on Macintosh laptop first. Then, just clone the repository OpenWrt from the Serval Project Github source and run the following commands :

```
./scripts/feeds update serval
./scripts/feeds install -p serval
./scriptes/feeds install -a serval
make world (long time running...)
```

Therefore, the binary for installation should then be in: bin/ar71xx/openwrt-ar71xx-generic-gl-ar150-initramfs-kernel.bin

To flash the PCB, we have to connect to the serial port at 115200 by typing:

```
cu -l /dev/cu.usbserial -s 115200\\
```

or if we have trouble with cu:

```
screen /dev/cu.usbserial 115200 \\
```

In either case, we will have to reboot the node in some way, so that we see the uboot prompt. Then we have to press any key to interrupt the boot process.
The Mesh Extender node will have an IP of 192.168.1.1, so we should pick another IP address on that subnet for the connected computer and type "httpd" to start the firmware update webserver. Then, just browse to http://192.168.1.1 and select the firmware file to upload, and trigger the firmware update.

5.1.2 Code

Here is the expect script for that :

```
#!/usr/bin/expect
#Expect script installing an openWrt image on the Serval Mesh Extender

set timeout -1
log_user 0

spawn cu -l /dev/cu.usbserial -s 115200

expect {
   "Connected."
   {
      puts "\n################################################\n# MESH EXTENDER
    FIRMWARE UPDATER\n#\n# 1/7 > Please, boot or reboot the PCB"
   }
}

expect {
```

```
   "Hit any key"
   {
      send "\r"
      puts "# 2/7 > Autoboot well interrupted"
   }
}

expect {
   "uboot>"
   {
      send "httpd\r"
      puts "# 3/7 > Server HTTP well started"
   }
}

expect {
   "HTTP server is ready!"
   {
      puts "# 4/7 > Uploading the image..."
      system curl --silent -o /dev/null -F
    'firmware=@./openwrt-ar71xx-generic-gl-ar150-squashfs-sysupgrade.bin' -F
    'filename=\$openwrt-ar71xx-generic-gl-ar150-squashfs-sysupgrade.bin'
    http://192.168.1.1/
   }
}

expect {
   "upload is done!"
   {
      puts "# 5/7 > Upload successful\n# 6/7 > Upgrading the firmware... DO
    NOT POWER OFF "
   }
}

expect {
   "done!"
   {
      puts "# 7/7 > HTTP upgrade is done! Rebooting..."
   }
}

expect {
   "Hit any key"
   {
```

```
      send "\r"
      puts "#\n# FIRMWARE SUCCESSFULLY UPDATED,
    ENJOY!\n################################################\n"
   }
}
```

5.1.3 Output and results

```
################################################
# MESH EXTENDER FIRMWARE UPDATER
#
# 1/9 > Please, boot or reboot the PCB
# 2/9 > Autoboot well interrupted
# 3/9 > Server HTTP well started
# 4/9 > Uploading the image...
# 5/9 > Upload successful
# 6/9 > Upgrading the firmware... DO NOT POWER OFF
# 7/9 > HTTP upgrade is done! Rebooting...
# 8/9 > Please reboot the node one last time in order to finish the installing
    process
# 9/9 > Second and last reboot. Almost done...
#
# FIRMWARE SUCCESSFULLY UPDATED, ENJOY!
################################################
```

We now have a brand new and up to date Serval Mesh Extender. If something goes wrong in the installation, the script just exit and tell you to launch the test again.

5.2 Test 2 : Wireless connections

The second automatic test i have to implement are related to the Mesh networks. Indeed we have to make sure that, after an upgrade or any random boot, the Mesh Extender will provide the expected networks. That means two wireless connections and one Ethernet connection. The first wireless connection is actually a hotspot on which every device can connect. The second one is the ad-hoc peer connection in order to communicate with other Mesh Extenders. This is the one who will diffuse all the Rhizome messages. Then the Ethernet connection is here to enable the transfer of data on the Mesh from a laptop. It will be a way to connect whatever else devices locally to the Extender as well.

5.2.1 Procedure

This test is a bit more complicated than the first one because it needs the output of the program to react according to it. This is a highest level of expect and did a lot of research to finally understand how we can fix this issue. Actually with expect, we can definitely not read the output of a shell. Or at least not directly. What we can though is to record or redirect this output into a file and then, read this file with expect. It is not very practical, it consumes time and memory but this is the only way...
In this program, first of all i need to check the presence of the wireless networks. For this purpose, i use the airport tool available in the Unix environment. There in one little problem though. The airport software is most of the time already installed on laptops but we have to add it to the global PATH before we can use it. Once this is done, we can check for the Wifi networks from the shell with the command :

```
> airport -s
```

We filter the result of it by piping the output with the grep command which i redirect directly into a file in order to use it with expect after :

```
> airport -s | grep servalproject > networks-found
```

Now, we have to check if the networks-found file contains indeed the networks we are looking for. We also have to make sure that, if there is many Mesh Extender working at the same time, the networks we are looking at is really the one from the Mesh Extender we are testing and not the one from another. For this, we can just compare the mac address of both the Mesh Extender and the one from the Wifi network. For testing the test script, i just simulate and assume that the mac-address of the testing Mesh Extender will be in an other file. I just have then to open both of these files and compare them as simple strings in iterative loops. If both of the addresses match, test is successful. For this test I have thus to make use of file writing and access procedures in the TCL language which is really new for me.

5.2.2 Code

```
#!/usr/bin/expect
#Expect script checking the wifi and ethernet connections of the Serval Mesh
    Extender

set timeout -1
```

```
#log_user 0

spawn cu -l /dev/cu.usbserial -s 115200

expect {
   "Connected."
   {
      puts "\n################################################\n# MESH EXTENDER
    WIRELESS CONNECTIONS TEST \n#\n# 1 > Please, boot or reboot the PCB"
   }
}

puts "\nCHECKPOINT1\n"

expect {
   "Starting kernel..."
   {
      puts "# Please wait while the Mesh Extender is booting..."
   }
}

puts "\nCHECKPOINT2\n"

expect {
   "nonblocking"
   {
      puts "# Mesh Extender started. Searching for Serval networks..."
      system airport -s | grep servalproject > networks-found
   }
}

set results(wifi-AP) "fail"
set results(wifi-AH) "fail"
set results(ethernet) "fail"

#read the file
set file [open "networks-found"]
set content [read $file]
close $file
set file2 [open "mac-address"]
set macs [read $file2]
close $file2

#exctract lines
```

```
set lines [split $content "\n"]

##Iterate over the lines
foreach line $lines {

    ## Split into fields on colons
    set fields [split $line " "]
   set stripped [lsearch -inline -all -not -exact $fields {}]
    lassign $stripped \
            networkName macAddress

    puts "Network name = $networkName and mac address is $macAddress"

    #tests
    if { $networkName == "mesh.servalproject.org"} {
      set results(wifi-AH) "succes"
    }
    if { $networkName == "public.servalproject.org" } {
      set results(wifi-AP) "succes"
    }
}

parray results
```

5.2.3 Output and results

```
################################################
# MESH EXTENDER WIRELESS CONNECTIONS TEST
#
# 1/5 > Please, boot or reboot the PCB
# 2/5 > Please wait while the Mesh Extender is booting...
# 3/5 > Mesh Extender started, looking for Serval networks...
# 4/5 > Wifi-AP "succes"
# 5/5 > Wifi-AH "succes"
#
# END OF THE TEST
################################################
```

Wifi-AP stands for the wifi access point which has got in this particular test this MAC address : public.servalproject.org e6:95:6e:40:5b:e2 .
Wifi-AH stands for the wifi Ad-Hoc network which has got in this particular test this

MAC address : mesh.servalproject.org 02:ca:ff:dd:ca:ce .
If the script does find the required network, it goes to "success" status. If not, it goes obviously to "failed" status.

5.3 Tests in development

5.3.1 Test 3 : Ethernet connection

This test is making sure that the Ethernet connection is working and operating properly. To this end I connect the PCB to the laptop again but with an Ethernet cable this time. The script will try to ping and open a SSH session between the laptop and the Mesh Extender. Once connected, it will try to run a couple of commands to make sure the SSH tunnel is usable.

Actually I did not finish to implement this test because I could not manage to ping the PCB nor succeed to open an SSH session. There must be a kind of protection or incompatibility with the devices. My supervisor told me that the next version of the OpenWRT image for Serval will probably fix it by allowing SSH connections. For now I am stuck and I just know that the Ethernet connection works well but without knowing how to test it. I give this next steps to my French fellows for the next semester.

5.3.2 Test 4 : Serval server

Now that we know that the Serval access point is running, we have to test the server it is actually running. For this, I will just run a Bash script who will call some curl commands in order to fetch the main urls from the server and test them.
The server is actually reachable on the IP : 10.0.0.1 and depending on the port we try to access, we have different pages. Likewise we can also access different application on this server depending on the speed connection of the Ethernet connection we established with the 'cu' Linux command. If I want to test the normal aspect of the Mesh Extender and launch Uboot, we need the speed of 115200. If now I want to access the Lbard application I have to double the speed and get to 230400.
On this server, we can also have informations about the hardware and other specificities of the Mesh Extender like information about the SD card mounting or the state of some others services. This is actually quiet useful for the tests but hard to exploit as this is presented as HTML documents which are not easy to deal with in a script.

Chapter 6

Results and discussion

The main part of my work and results are available on the Github repository I introduced in Chapter 4. We can find there the codes and the scripts I have written.

Results are like expected scripts we can run on a laptop to test the Serval Mesh Extender. The first one works very well and, as expected, we can run it without issues and update the firmware of any Mesh Extender with the image we want. We can use it by just running the command :

```
./nom-du-test.exp
```

To improve the testing experience, we should automate this test without rebooting manually the PCB. Indeed, to start or boot it, we have to manually replug it, to shut it OFF and ON again. We should be able to do it automatically in the future by adding a switch on the "hardware test unit" that Ryan was suppose to develop and in which will be plugged the PCB. With this, we will be able to command from the outside and without human interactions the boot of the PCB that will allow us to run all the tests in a row.

I finished to code the second script for the connections check at the same time Paul released a new version of the Mesh Extender software. This new version was better in terms of functionality but brought a big issue in my code. Indeed this update is communicating with Lbard, which is a good thing. But it also prints on the standard output every single piece of signaling characters that the communication produce. And this is not good because Expect is monitoring every characters on this output. It appeared that some special combination of these special characters was a secret command of expect that makes it stop running or exit the program... Since this moment I could not do anything because I was not able to test my tests myself and even just run them.

I learned here something really important. When you are coding something which depends or interact with an other software or device, you should definitely take into consideration the fact that this second software could change anytime in a way or another as with this serious update. Next time, I will consider this thing more seriously and try to develop something more universal or easier to adapt to changes. I understand now why programs have to be written clearly and in an easy way to understand the philosophy of it. It is because we need to adapt to the exterior environment. Therefore we also need a flexible program.

I did not know at all that I was lacking of time. But this is a real thing that every little notion you have to learn take so much time. It is exactly the same with informatics issues and bugs. When something does not work in your code or if you can not managed to install a software, this take an humongous quantity of time. For example I have been stuck for one week by just wanted to install the openWRT image creator on my laptop. What was at first just a software installation turned into a giant puzzle. Indeed for some mysterious reason this distribution can not be created from a mac. I had then to install a Linux Virtual Box and to install the distribution on it. First lost of time, it took one entire day just to create the image. After this, I obviously have to transfer the created image from the Virtual Box to the laptop. Here again, I met an issue with the 'scp' command cause the SSH protocol was not possible on the mac laptop. I had to find a other way to transfer it. In the end something that should have take me one hour to do took me three days. And this is just one example out of so many more. In informatics, you are always confronting to various and numerous problem where you were probably not ready to see them. And one problem always bring an other one with him. So if there is one rule I have learned from that, it is that we have to be very careful and meticulous and everything little thing you do in order to avoid problems and therefore loss of time.

Finally this lack of time bring us to this last point: the fact that I did not do as many tests as I initially wanted to do. I wish I could have written at least one or two more scripts but I was definitely too busy with all my issues to do it. Fortunately it is not that a big deal since I have learned so much with all this problems and issues I met during my project. This is probably due to the fact that it is my first researching experience. I am pretty sure now that if I do it again, I would be a lot more efficient and would get my work further. It is a matter of experience.

Chapter 7

Conclusion and future directions

My master project is now completed and I can say that it was a really good experience for me. Some of the tests are finished and uploaded, we can now check automatically various features of the new Mesh Extenders to bring them to Vanuatu. We can automatically check the firmware of a device and update it with the last version. Admittedly some of the tests are not working because of numerous issues I met during the project like the "update issue", but they are still here and part of my work. Thanks to them, I have learned so much about the world of research and about how do I have to organize my work as being part of it.

However this is just the beginning of the work. My project is finished but THE project has just started. More tests have to be done. More Mesh Extender have to be checked and send to Vanuatu. More people have the right to use distant communications. This become now the work and the project of the following French students who will come here for the next semesters. I will give them my work so they have something to start with and I have faith in what they will accomplish as I have faith in the Serval Project which is a really was a really good adventure for me. I sincerely hope that this adventure is just about to become bigger and bigger.

FIGURE 7.1: Good perspectives for Serval

Chapter 8

References

[1] Serval Project, "What is the Serval Project?" 2016. [ONLINE] http://servalproject.org/

[2] P.Gardner-Stephen, R. Challans and M.Loyd, "The Serval Mesh: a platform for resilient communications in disaster and crisis", ***Proceeding of the 3rd IEEE Global Humanitarian Technology conference***, GHTC 2013, pp. 162-166, 2013.

[3] P.Gardner-Stephen, "The Serval Project: Practical Wireless Ad-Hoc Mobile Telecommunications", no. June, pp. 1-29, 2011

[4] Serval Project, "Rhizome Journal", 2013. [ONLINE] http://developer.servalproject.org/dokuwiki

[5] Serval Project, "Serval DNA", 2014. [ONLINE] http://developer.servalproject.org/dokuwiki

[6] Serval Project, "LBARD: Low-Bandwidth Asynchronous Rhizome Demonstrator", 2016. [ONLINE] http://developer.servalproject.org/dokuwiki

[7] Electrosense Technologies, "Electrosense Technologies", 2017. [ONLINE] http://www.electrosense.co

[8] GoTenna, "GoTenna", 2017. [ONLINE] http://www.gotenna.com/

[9] Farm Monitoring Solution, "Farm Monitoring Solutions", 2017. [ONLINE] http://farmms.net/

[10] Observant, "Observant", 2017. [ONLINE] http://observant.net/

[11] Beartooth, "Beartooth", 2017. [ONLINE] http://www.beartooth.com/