

E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

A friendly approach to Open Government: Chatbots



Grado en Ingeniería Informática

Trabajo Fin de Grado

Javier Bermejo Razquin

Enrique Armendariz Iñigo

Pamplona, 7 de junio de 2018



ACKNOWLEDGEMENTS

After an intense and long semester, the day has come to finish this final degree document and to thank to all of those who have helped me in this project.

I would like to start saying thank you to all my colleagues from my internship at IECISA. They have always helped me, giving me advice and supporting me not only during this semester, but also in the ten months I have been here at IECISA.

I would also like to thank my final degree tutor, José Enrique Armendariz Iñigo, who always supported me across my university studies. He has been my mentor and thanks to him I have been able to develop my knowledge far beyond the degree syllabus.

Also, I would like to thank all my friends; not only to the current ones, but also to the ones who have been there, even if the time, distance or situation have separated us.

Finally, I would also like to thank all my family, for their support and wise counsel. You always have been there for me, no matter the distance, time or situation.

Thank you, if now I am here is thank to each and every one of you.

ABSTRACT

This end-of-degree project presents an idea to assist citizens in the use of Open Government online portals. These serve, among other functionalities, to do bureaucratic processes without needing to go to their offices. Most of the users do not know how to use these services or what they can do with them. It is in the interest of the government to make the inhabitants use these utilities in order to reduce bureaucracy.

This problematic is found in one of IECISA (*Informática El Corte Inglés S.A.*) clients: The Commonwealth of Pamplona (*Mancomunidad de Pamplona*). The author of this research, as an intern of this company, will develop a prototype of a possible solution for the client.

The suggested solution is the use of virtual assistants for the guidance of the users. These are called chatbots. You can communicate with them and they will facilitate you the desired function, either indicating you how to do it or fulfilling the task for you. Nowadays we have several services like the one proposed, for example Siri or Google Assistant, but these are focused on daily tasks. This project takes the technology of those assistants (Natural Language Processing) and uses them to accomplish government procedures.

To determine if this solution is a proper one, the author will build and publish online a chatbot that performs simple actions. In order to know if the software developed is effective, the testers will receive a poll where they will evaluate the solution. Then, these results will be analysed and some conclusions will be developed.

The conclusions will address if the project is useful and if it is a viable option to increase the use of these portals.

Keywords

- Chatbots
- Conversational Bots
- Virtual Assistants
- NLP – Natural Language Processing
- Machine Learning
- Open Government

CONTENT

Acknowledgements.....	II
Abstract	III
Keywords	III
Content.....	IV
Chapter I: Antecedents	1
Chapter II: Introduction	3
2.1 Background	3
2.1.1 The Administration and the Open Government	3
2.1.2 Virtual Assistants	4
2.1.3 Chat Services	5
2.1.4 Chatbots	5
2.1.5 Summary	6
2.2 Definitions	6
2.3 Motivation.....	7
2.4 Goals.....	8
2.5 Methodology.....	8
2.6 Conventions	9
Chapter III: State of the art.....	10
3.1 Review of the Actual State	10
3.2 Proposition	11
Chapter IV: Analysis	12
4.1 Requirements Analysis.....	12
4.1.1 Functional Requirements	12
4.1.2 Non-Functional Requirements.....	12
4.2 Solutions Analysis	13
4.2.1 Chatbot	13
4.2.2 Analysis	13
4.3 Proposed solution.....	14
4.3.1 Chatbot	14
4.3.2 Analysis	14
4.4 Security and Data Protection Analysis.....	15
4.5 Budget.....	15
Chapter V: Solution Design.....	16

5.1 Architecture	16
5.2 Software tools	17
5.2.1 Chat Application.....	17
5.2.2 NLP.....	18
5.2.3 Business Logic Server	19
5.2.4 Forms	20
Chapter VI: Implementation	21
6.1 Dialogflow.....	21
6.1.1 Description.....	21
6.1.2 Design	23
6.1.3 Major Design Decisions	27
6.2 Facebook Messenger	28
6.2.1 Integration.....	28
6.2.2 Graphic Interface	28
6.2.3 Tests.....	29
6.3 AWS Lambda and Client SOAP Calls.....	29
6.3.1 Lambda Function	29
6.3.2 SOAP Service	30
6.3.3 Endpoint API Gateway	30
6.4 Forms, Interviews and Testing.....	31
6.4.1 Forms	31
6.4.2 Testing.....	31
6.4.3 Interviews.....	32
Chapter VII: Results.....	33
7.1 Demographics	33
7.2 Chatbot	35
7.3 Idea and Future Improvements.....	41
7.4 Administrative Portals.....	44
Chapter VIII: Conclusions	51
8.1 Chatbot Development and Design.....	51
8.2 Administrative Portals and Idea Execution.....	53
Chapter IX: Future Lines.....	55
Chapter X: Bibliography	57

CHAPTER I: ANTECEDENTS

Along thousands of years, the humans have developed abilities that have helped us to organize our actual society. People specialise in certain areas and form groups of experts, developing new abilities. The mix of these experts creates new possibilities and tools that help us evolve. One crucial step in the human evolution was the development of civilizations.

A civilization is “that complex of cultural phenomena which tends to occur with the particular form of socio-political organization known as the state” [1]. A civilization is characterized by an important urban development, cultural expansion, communication strategies and social, laboral and territorial administration [2].

As the previous definition says, the State is the entity who organises a civilization. This body is formally defined as “A nation or territory considered as an organized political community under one government” [3]. This political community, divided in several entities, is the one who is going to fully manage the state.

Finally, a government is “The group of people with the authority to govern a country or state” [4]. This means that this group is going to be the responsible of managing and commanding the several entities that form the state.

Summing up, a civilization is organized by a community called the State, who is divided in several entities that are managed by the Government.

The Government, being one of the previously mentioned groups of experts, is the body who rules and decides complex decisions for the citizens of the state. This body is helped by another one, who carries their orders and manages all the governing process, called the Administration. Those are also in charge of talking with the citizens and process all their petitions.

The Government needs information from their citizens, their feedback and process their demands. But as they have a lot of responsibilities, they are constantly helped by the Administration organ. When a citizen needs to contact with the Government or collaborate with them, he will contact the Administration. This State organ will be the responsible of managing and processing his inquiry. The main problem with this system lies in the huge distance between the citizens and the administrators, both in interest and skills.

To the best of this author’s knowledge and according to the client information, it seems that the clear majority of citizens do not know how the administration works or how to execute a procedure in order to achieve a desired service. They are lost, and they usually fear these processes. Most of the individuals skip these operations if they are not mandatory, and if so, they will go to a physical office, looking for help and guidance. Even if multiple governments have set up online systems in order to facilitate these tasks and reduce times, most of these sites are confusing for the average user.

These online systems are part of the Open Government System. This system tries to “secure concrete commitments from governments to promote transparency, empower citizens, fight corruption, and harness new technologies to strengthen governance” [5].

Part of these systems are focused on citizen engagement and participation, but there is a big obstacle to solve to make this solution viable.

An average citizen finds the language and the online features of these services quite complex. Currently those websites are not prepared to be browsed by inexperienced users. Also, most of the operations require knotty procedures in order to achieve the desired service, like e.g. acquiring special computer certificates. It is usual to find instructions to achieve those actions, but they are usually long processes and not straightforward. So, we might ask ourselves why this system is still being used.

It is a complex and slow one, but it is also the only one that we know that actually works. It is hard to manage the population information and their needs and objectives, and even if it is automated via computer programs, citizens will still need the help from experts. Changing the entire system would be an expensive solution, but it could be plausible to seek systems that could guide or educate the citizen.

Our society uses constantly their smartphones; these are mobile devices that are not only capable of calling and sending text messages, but also of connecting to the internet or executing applications. There is a new technology on those devices called Personal Assistants. These are services that are integrated into applications or chat programs that help you achieve different tasks using natural language. This could be a solution to the aforementioned problem.

The assistant could guide the user across the system, helping him to achieve the task or even explaining him how to do it. Some actions could be automatized by the assistants and others could be interactive guides, where the user could ask questions and demand for a more detailed explanation in certain steps.

But even if this solution is simpler than changing the whole system, it is still an intricate one. Before developing this whole solution, we need to make sure that it is a viable one and that the user will understand it and take advantage of it.

CHAPTER II: INTRODUCTION

This chapter will present a project overview. It will be explained the actual background in order to understand the motives and goals, which will be also assessed in his chapter. Finally, the used methodology and the conventions of the study will be introduced. The rest of the work is organised as follows:

- **Chapter I: Antecedents**
A prelude of the project, where the reader can get an insight of the main idea.
- **Chapter II: Introduction**
The research will be briefly explained here. The background needed for understanding the issue will be presented and the solution will be shortly explained. The motives of taking this project and goals that it tries to achieve will be also assessed.
- **Chapter III: State of the art**
Here the reader will be able to read an analysis of the state of the art of the technology and what tools are currently available. With this information, the author will set up a project proposal and it will be assessed what place it takes in the development area.
- **Chapter IV: Analysis**
Once is known what is the actual state of the art and what can it be contributed, the proposal will be detailed. Requirements and different solutions will be analysed, in order to propose a final one and calculate his cost.
- **Chapter V: Solution Design**
In this chapter the tools and architecture will be chosen to develop the aforementioned proposal.
- **Chapter VI: Implementation**
It will be explained how the selected tools have been used to fulfil the proposed solution
- **Chapter VII: Results**
The outcome of the project will be presented and analysed in this chapter.
- **Chapter VIII: Conclusions**
The project will be closed with some concluding thoughts about the development and the nature of the concept
- **Chapter IX: Future Lines**
What could be addressed in future continuations of this study
- **Chapter X: Bibliography**
All the information and resources that have been used are in this chapter.

2.1 Background

2.1.1 The Administration and the Open Government

The citizens of a country have to deal with several government procedures in order to have an organised and effective society hierarchy. Those procedures are managed by The Administration, an important governmental body.

Administration requires of the citizen active participation to finish all formalities on time. They expect information from the citizen or the confirmation of it; and usually this cooperation must be done in a not intuitive and specific way.

Users have to deal with complex and tedious forms, which must be fulfilled in an exact process and delivered in a precise time and format. It is very common to see that a user has failed to accomplish a specific task and has to restart all the process.

Also, the big influx of people doing all this paperwork does not give a better experience. It is usual to go early in the morning and find long queues or delays. The timetables of the administration offices are not very flexible, finding frequently an only-morning schedule. Not all the people are able to do the paperwork in the morning, as they are working, and they are obliged to take vacation to try to put all their formalities in order.

The Government and the Administration have been trying to fix these problems creating the Open Government idea, where these entities try to involve citizens actively. One of the methods to connect with the civilians is building online portals, where they can learn how to do those procedures and even do them at home. Nonetheless, it is still complicated. Not everyone knows how to browse the web with ease and the language used in the portals is usually hard and complex. The requirements are diverse and not easy to achieve, the paperwork is not guided and the users are inexperienced in the field. Without the human interaction they feel alone and lost.

2.1.2 Virtual Assistants

“An intelligent virtual assistant is an engineered entity residing in software that interfaces with humans in a human way. This technology incorporates elements of interactive voice response [...] to deliver full-fledged ‘virtual identities’ that converse with users.” [6]. During these last years we have seen this as a trend in our smartphones, helping us to accomplish daily actions or automating tasks in order to boost our productivity. Even if at first these did not work as good as planned, lately they have become more powerful and they are starting to become an essential part of a phone.

But this technology is not only present on our mobile phones; it can be also found on other platforms, as in websites, applications or even in hubs (machines that their only purpose is to give the user the possibility to talk to their assistant around their house). Those companions are usually build one time and adapted to several platforms, in order to give a seamless experience across all the systems supported.

With the raise of this technology, there are several tools that help us build a multiplatform bot [Image 1]. Those tools constitute a framework, and depending what type of bot we want to build, we can use one or another. Dialogflow is one of the most versatile; but there are others like Chatsuite, focused on marketing and user experience, or IBM Watson, centred on support and business.

2.1.5 Summary

The solution proposed is to utilise the Chatbots technology and use them to bring closer the administration to the citizens. Those bots could assist the citizens and help them process correctly all the paperwork they have to do for the administration.

2.2 Definitions

Term	Synonym	Definition
API		A set of defined functions which uses data from one or more online services with the objective of developing applications
Administration		The governmental body who carry the State orders and manage this process.
Chat Services		Online services that offer online communication over real-time messages.
Chatbots	Bot	Chatbots are Virtual Assistants that work on messaging applications and that they receive all information via text.
Citizen		A person who lives in a State or city
Civilization		“That complex of cultural phenomena which tends to occur with the particular form of socio-political organization known as the state” [1]
Context (Dialogflow)		Dialogflow tool who serves to indicate the current state of a user request
Entity (Dialogflow)		Dialogflow tool used to extract parameter values from the text sent by the users
Government		“The group of people with the authority to govern a country or state” [4]
Intent (Dialogflow)		In Dialogflow, relation between the input of the user and the action that should be taken
JSON		JavaScript Object Notation. A text format for data exchange
NLP		Natural Language Processing. A tool who transforms the text introduced by the user in data that the machine can work with.
Open Government System		“Multilateral initiative that aims to secure concrete commitments from governments to promote transparency, empower citizens, fight corruption, and harness new technologies to strengthen governance” [5].
POST Request		Request method. Serves to publish information in a web service.
SOAP		Simple Object Access Protocol. Protocol who defines how two objects can communicate between them.
State		“A nation or territory considered as an organized political community under one government” [3]
Tester	Evaluator, Auditor	Person who is going to evaluate something
URL		Uniform Resource Locator. Specifies the location of a website or system
User	UX	It refers to the satisfactory experience of a person

Experience		using a software system
Virtual Assistants	Assistants	“An intelligent virtual assistant is an engineered entity residing in software that interfaces with humans in a human way. This technology incorporates elements of interactive voice response [...] to deliver full-fledged ‘virtual identities’ that converse with users.” [6]
Webhook		Event notification, typically sent as a POST request, which contains data that is posted to a specific URL. This URL defines the location of a script, which will process the data in the POST request. [8]

2.3 Motivation

I decided to develop my final degree project at IECISA (*Informática El Corte Inglés S.A.*) because I think that developing a final degree project in these conditions would give me a real insight of how a private company works. Also, I did an internship during my first semester of the university course 2017/2018 and my overall experience was very good. They offered me the possibility to develop a Chatbot demo for a plausible client, the Commonwealth of Pamplona (*Mancomunidad de Pamplona*), and I accepted rapidly.

There are several reasons why I selected this topic and why I decided to do it in a private company, but they can be categorized in 3 parts: Interest, Work and Education/Investigation

Since the emergence of smartphones and social networks, I am interested in those technologies. I use them daily and I find that they still have a huge potential. And personal assistants are a great idea that is developing very fast and that it could be the future of smartphones. Developing an assistant that could be used in a mainstream messaging application felt like a very interesting project. With this plan, I could include nearly all the aspects from my degree: Software Engineering, AI, Computer Networking and Project Management.

The company where I was doing my internship was developing a huge project around Chatbots for numerous clients. The plan was to reduce paperwork at administrations and bring closer the administration to citizens. They were going to do this using several assistants adapted to each organisation and task. As my time for developing my final degree project was limited, and in my office there was no one taking part in the project, I could not join the project per se. So, they decided that I could develop a demo using simpler tools and reducing the scope, trying to analyse if it was a useful tool or not.

Finally, I could develop my personal knowledge about a new technology that is starting to grow. It is something new and I could do a little contribution on the field of the company. The chatbots are new tools that we know almost nothing about and they look like a great solution to numerous problems. However, they are still unexplored and there is no proof that they can be useful in a daily basis.

2.4 Goals

The goals of this project can be resumed in two points: First, build a chatbot that works in a mainstream application and thus resolves the problem of the client; and, secondly, see if the use of chatbots increases the use of administration portals.

The client with whom IECISA works has interest in increasing the use of the administration portal, to reduce time needed to perform all the paperwork and increase productivity. Also, the client wants to raise the usage of these systems in young citizens, as according to the client, most of them are not interested in these platforms.

It is important to note that the goal of this project it is neither to develop a complex bot nor doing a complicated statistical analysis of the use of administration portals before and after the use of the chatbot. The bot will be a demo project, a working idea to show to the client what the company could build in the future. This is a decision taken due to the reduced time to develop this concept (4 months) and the wide scope of it. Finally, the analysis will be a skim one, studying if the use of this bot encourages the citizens to use administration portals.

2.5 Methodology

This project is going to be split into 3 phases, which will be subdivided in several tasks. A phase will not be finished until we end all the tasks from it, but it is not necessary to accomplish a task to start the new one.

Phase 1: Investigation and Planning

This phase will go from the 15 of February to the 1st of April.

In this phase the technology will be studied and a first approximation to the problem will be sought. The project will be documented, writing the planning, the design, etc.

The first task will be to do a research about how this technology works and what is needed keeping in mind the creation of the chatbot. The author will learn how to create a system like this, what type of architecture is the right one and what are the main components and their use in this software. Different tools will be sought in order to build this system, compare them and, finally, choose the one that adapts better to the project demands.

Another task will be skills acquisition. It will be analysed what is it needed to learn how to create this software. This will be done by developing small test programs in order to test the technologies and doing researches on good practices for each technology.

Then we will unite and document the findings and decisions to deploy the system. Once a little from the project is known, it should be possible to prepare a presentation.

Phase 2: Development

This phase will go from the 1st of April to the 1st of May.

The objective of this phase is to build each part of the chatbot and integrate all their components.

The priority of this phase is the development of the bot logic. The robot must learn what to answer in each case and the information that it has to ask for the user. For each

administrative process, the assistant will have to ask for optional information, like a photo or a postal address, and it should be prepared for it. Also, it will be necessary to do some testing in order to check if the robot behaves correctly.

Another task is the messaging app integration. In order to get the best user experience, some tweaks should be made. If the application offers the possibility to include buttons, images or cards to create an engaging interface, they should be included.

Finally, the last part of this phase is the communication between the bot and the client. All the actions made in the assistant must have an impact on the client's system. Nonetheless, the author depends on the collaboration of the entity and it is impossible to assure a seamless integration.

Phase 3: Analysis and Improvement

This phase will start on the 1st of May and will last till mid-May, leaving some time to finish and polish the final degree project document.

In this last phase, the chatbot will be analysed and improved according to the data extracted from the forms sent to the users. In this phase, only the analysis will be indispensable. The chatbot improvement will depend on how much time is left.

The first and main task is the analysis of the software developed and the user behaviour with it. Some forms will be sent to the users in order to know their feedback and give some insights. In case that the project does not achieve its goal, it will be important to know if it is because of a lack of quality in the developed software or due to a lacklustre idea.

The final and optional assignment is the improvement of the assistant. Looking at the data extracted from the aforementioned task and at the client requirements, it is possible to expand the functions of the bot. But as it was said before, this task is optional and it should be done only in case that the rest is finished

Constant Work

During the entire project, some constant work will be made. These tasks do not take part in an exact phase, but during all or most of them.

This final degree project document will be constantly updated with all the development actualisations. This will not be written at the end of the project, in order to capture all the details of it.

2.6 Conventions

During all this document, some conventions will be used.

- Text will be in bold when we are referring to code variables or parameters. This will be seen regularly in chapter VI
- Cursive will be used for all the foreign words
- Quotes (“”) will be used for citations.

CHAPTER III: STATE OF THE ART

In this chapter it will be assessed the actual circumstances of the technology used and other similar cases as ours. After the analysis of the situation, a proposition will be defined in order to improve the wide scene, or at least to try it.

3.1 Review of the Actual State

It could be said that chatbots are a new tool created with AI (Artificial Intelligence) technology, but the truth is that it is not exactly like this. The current version and uses of them are innovative, but talkative robots and artificial intelligence have been a constant challenge in the informatics field [9].

During the whole history the human being has tried to develop intelligent machines. It could be said that the creator of the Modern Computing was Alan Turing. It was in 1936 when he published his paper [10] where he explained the theoretical basis of the devices that became known as Turing Machines. And even if these machines were not intelligent per se, they were the first ones to solve mathematical problems and could be categorised as such. Turing continued his research in computation and smart machines till his death, and he developed a tool that is even used nowadays: The Turing Test.

The Turing Test is an evaluation of a machine ability to show an intelligent behaviour equivalent or indistinguishable from the one of a human [11]. The development of the examination is simple: there is an interrogator who is going to communicate with a machine and a human (without knowing who is who), and he has to guess which is the machine. If the evaluator thinks that the robot is a human, the agent will pass the Turing Test. And even if some investigators think that it is not the best way to test the intelligence of a computer (it can be tricked), it can test if the machine behaves like a human being, and that is already a huge milestone, one that is about to be achieved.

Currently we have several virtual assistants, chatbots or robots that can talk to us and follow a normal conversation. They are not as powerful as we can see in movies or series, and can be easily identified as artificial machines, but nonetheless those are actually functional and can help a lot of people in a day to day basis. People are becoming used to unlock his phone and ask to his virtual assistant to send a message for him [12] [13].

And that's because virtual assistants are becoming something normal in our society. Everyone has one in their mobile phones (Siri from Apple, Google Assistant from Google, etc...) and even we are starting to have them at our home in hubs, televisions, fridges, etc. It is a new trend that is starting to grow and a lot of companies are adapting and developing innovative functions. [14]

Some of the best assistants are being opened to developers in order to expand their functions and make them more powerful. There are various that are integrating AI techniques as image recognition or photo processing, in order to integrate visual assistance to the user [15]. But nonetheless, there are several problems with these assistants.

Even if those are very powerful and have plenty of functions, most of them have issues dealing with daily problems, especially with context recognition [16]. This, mixed with a cumbersome integration with other apps, creates confusion on the user. For example, a person might use only an instant messaging application (like Whatsapp Messenger) to communicate with his contacts, but if he tells his assistant to send a message to one of them, it is very probable that it will send it via SMS.

This might happen because the industry has not set up a standard for this type of development. In many technologies there are paradigms that the developers follow in order to bring the best possible product and hoping that it can integrate seamlessly with other market solutions (for example, SMS or charging cables are standards from the mobile industry). This makes that most projects are developed individually and might not cooperate correctly and work differently ones from the others.

This creates confusion on the daily user, who does not know if the assistant is malfunctioning or if it just is not able to accomplish the task. Sometimes they have to try different commands in order to achieve the action asked, like saying “Play some music using Spotify”, instead of saying “Play some music”. This is a proof that this tool is not fully developed and the industry has to cooperate in order to make a coherent system.

Chatbots, virtual assistants and all the derivatives from this new trend are a promising feature that needs a lot of polishing. It needs to have an industry standard to make the user comfortable and to change their customs. Nonetheless, the industry has to keep pushing forward this software and continue improving it, especially expanding their progress to other languages different to English and enhance the context recognition.

3.2 Proposition

However, those are no small accomplishments, and it needs to be proved that the public is ready to utilise this technology and that they can be able to interact with the assistant without getting lost. There is no sense in improving a tool if at the end nobody is going to use it. Smartwatches are an example, a tool enhanced and developed during years but that no one uses because there is no interest [17].

Also, in order to make something popular, it needs to be easy to replicate. We want to see developers and companies creating small chatbots and to integrate them in their daily use. If the industry is not able to build robots like these, it is very probable that the technology does not take off.

So, it is very interesting to know if users are able to interact with these chatbots and if the companies/governments are able to create automata easily. If those two are given, it is highly probable that the technology continues its evolution and in some years we could find a society totally adapted to these new machines.

The aim of this research is to know if these two conditions have been achieved and to what extent. This will be done by creating a simple chatbot and analysing the behaviour of the testers, evaluating if there is an increase of participation in their online platform. The functions of this bot will be related with the objective of our client, the Commonwealth of Pamplona (*Mancomunidad de Pamplona*).

CHAPTER IV: ANALYSIS

In this chapter the requirements will be studied and solutions will be proposed in order to achieve them. Then, those will be assessed and one of them will be selected and completely documented. Security and budget will also be discussed.

4.1 Requirements Analysis

In order to prepare a solution for the client, we need to do a requirement analysis. We will see what functions our bot has to accomplish and document them. All of these are a blend from the client and the researcher.

4.1.1 Functional Requirements

- 1) REQF1: The bot has to be able at least to receive complaints from the users. More functions could be added in the future (like sending congratulations)
- 2) REQF2: The bot has to help and guide the user to fulfil his petition.
- 3) REQF3: The bot has to be able to detect about what service we are talking. If someone complains about something, the assistant will assign it automatically to the correct department.
- 4) REQF4: The bot will reduce to the minimum the information asked to the user in order to reduce the time needed to fulfil the task.
- 5) REQF5: The bot has to be able to receive images from the user and store them.
- 6) REQF6: The bot has to use the recourses of the selected application and adapt to it. For example, it could take advantage of the User Experience (UX) buttons provided by the platform.
- 7) REQF7: The bot has to send all the information to the client via an SOAP service.
- 8) REQF8: All information has to be confirmed to the user before sending it.
- 9) REQF9: It has to be measured if the bot has helped the user to get closer to the digital administration.

4.1.2 Non-Functional Requirements

- 1) REQNF1: The bot will be able to answer the user instantly, without making him wait long periods of time. All the answers should be resolved in the next 10 seconds of the query
- 2) REQNF2: The bot has to be integrated in a mainstream application, in order to be accessible to a maximum number of users.
- 3) REQNF3: The bot should be accessible via smartphone, but it would be a positive point if the selected application for the deployment could be multi-platform.
- 4) REQNF4: The bot will not reveal personal information from other users than himself.

4.2 Solutions Analysis

Now that the requirements for this project are known, some solutions have to be proposed. It could be said that there are two divided objectives. The first, create a chatbot, and the second, measure the effect of the bot on the tester according to their use of digital administrative platforms.

4.2.1 Chatbot

It has been mentioned repeatedly that the chatbot is a straightforward solution for this problem, so it could be said that there are not a lot of things to discuss.

The only thing that needs to be decided is if the assistant needs Natural Language Processing (NLP) or not. It can be necessary the inclusion of a NLP in the bot, depending on its features. For example, the assistant might need to map words to entities in order to understand what type of complaint is executing.

- If NLP functionalities are introduced, the development time will be longer, but the robot will be able to understand about what the user is talking.
- If these are not introduced, the development will be shorter but it is probable that the assistant is not going to understand what the user wants, excepts if he uses keywords. This could be fixed using buttons on the application user interface, but the quality of the bot would be reduced and the user would be obligated to see the screen to answer (and lose the opportunity of using voice commands)

All the other decisions related to the tools selection and software architecture for the chatbot will be discussed on the next chapter (Chapter number V)

4.2.2 Analysis

In order to analyse our chatbot, data needs to be collected. The methods to do it have to be decided. There are several ways, like install analysis tools like chatbase, send forms or do interviews:

- Use of analytics tools: Those are tools that are integrated in the solution and analyse each of your queries, using statistical methods in order to extract information and rate the precision and use of the bot. The use of this tool is a very good idea, but it has several problems. The installation of these tools is usually complex and you have to implement and manage each case individually in order to have precise and relevant information. Also, this would require launching the chatbot to the general public to have enough information to generate relevant data.
- Interviews: Each tester will use the chatbot at the same time that a member of the team is interviewing him. Questions can be asked before, during and after the auditor uses the assistant. Also, the interviewer can analyse the reactions of the tester, finding unexpected behaviours. Nonetheless, this method implicates that the auditors have to set up an appointment with the team, and not a lot of people are usually willing to give so much of their time.
- Use of forms: In this case pools would be created where each bot tester would answer one of them giving his opinion. It is easier to implement, but questions

must be precise and each one of them must extract relevant data. Also, it is probable that most of the testers do not answer it and information is lost, unlike other methods where they are answering at the same time they experience the bot.

Another decision to be taken is how many people are going to participate in the testing phase:

- **Public testing:** The chatbot is open to anyone willing to participate. They can access publicly to it and answer the poll (in case there is one) once they have used the app. With this method more testers will participate in the analysis. However, this means that our chatbot has to be approved by the chat application company where our application will be deployed. Also, permission from our client will be needed.
- **Private testing:** The chatbot is only open to a limited number of testers. They can only access if they have been selected to the testing. This option means less people participating but more precise feedback.

It is important to note that we are unable to use publicly the name of the client without their authorization.

4.3 Proposed solution

4.3.1 Chatbot

In this case the selected solution will be the use of NLP in the chatbot. The reason is simple: we want the user to be able to talk with the bot naturally. The objective of this project is not the implementation of an interactive form, but of an assistant who guides you naturally to your final objective. If we do not implement a Natural Language Processing, this effect might be extremely reduced.

4.3.2 Analysis

As we do not have the authorization to use publicly the name of the client, the proposed solution is forced to be the private testing.

The first option about data collection to be taken out of the list is the use of analytics tools. These ones are hard to implement and do not offer so much information in order to be worthwhile. If we had thousands of testers, maybe this would make sense, because it will be easier to have all the information recollected, but it is not the case.

The proposed solution will include use of forms. With this option we can recollect all the information needed easily and at the same time manage the people that will participate in the private testing.

We do not rule out the option to realise private interviews if we see that the information is scarce or we see strange behaviour on the data recollected with the forms.

Later in the document (Chapter V), the proposed solution will be explained in detail, adding the selected tools and architecture for the development.

4.4 Security and Data Protection Analysis

In this project we will use sensible data, like personal information of the testers. This information must not be lost or stolen and has to be secured. Nonetheless, this is not a long investigation, time is limited and we cannot cover integrally each aspect of it.

As the main focus of this research is to find out if this solution could be applicable to the real world and if the people would react positively to it, a lot of time cannot be spent in this feature.

In the next chapter, where the solution design will be discussed, the security and data protection will be taken as an important aspect of the design.

More actions should be taken in this aspect, as encrypt the data or take into account security local laws, but the time is limited and as it said previously, the scope of the project has to be reduced in order to finish it on time.

4.5 Budget

As this is a project demonstration, its budget should be small. In the solution design chapter, licence software prices and server costs will be taken into account. Nonetheless, the selected tools should be able of managing high network loads and concurrent users. The system should be scalable and prepared for working in a production environment.

This is also done to demonstrate that a small company can develop and maintain a chatbot service, as this is one of the objectives of this research.

4.6 Collaboration

This research project has been realised in an enterprise (IECISA) who is developing a full program to introduce chatbots to the public administration and maybe to other companies. This concept is being developed in the headquarters of Madrid, while this research project is being created at an office in Pamplona. Collaboration between the two teams is complicated by the distance and because the internship offered is reduced in time (only 4 months).

So, it has been decided that this research will be part of a pilot program, where for first time a small prototype will be developed to show it to a client.

This prototype has been show to the client, the Commonwealth of Pamplona (*Mancomunidad de Pamplona*), and they are highly interested in continuing with this project. Also, this has been shown at IECISA Madrid Headquarters and they are very happy with the result.

CHAPTER V: SOLUTION DESIGN

In this chapter a solution will be designed in order to fulfil the proposed solution on the last chapter. Tools will be selected taking into account their pros and cons and the architecture will be planned.

This chapter will focus mainly on the design of the chatbot, and it will be only talked about the forms for selecting the tool where they will be managed.

5.1 Architecture

In order to select the precise tools needed for this solution, it is needed to see what kind of architecture it is going to be developed and deployed

For executing the proposed solution, several tools are needed to be used:

- A chat application, in order to deploy and consume the chatbot
- A Natural Language Processing (NLP), to transform the text introduced by the user in data that the machine can work with and recognise his intention.
- A Business Logic, to execute the actions asked by the user.

All these tools need to be used in a specific order. First, the user will write his query in the chat application, and then the text will be send to the NLP tool. The latter will process the text, pick the necessary information and recognise what the user wants to do. After, it will send this information to the server and the business logic will be processed. Once it finishes, the NLP tool receives a response from the server, and depending the result, sends a different message to the user. This can be seen in [Image 2].

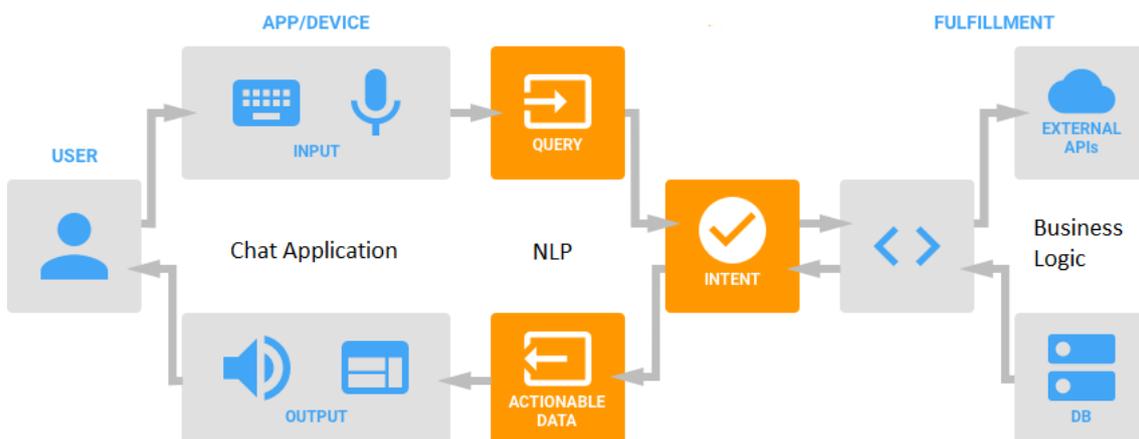


Image 2: Chatbot Architecture [18]

Before selecting which tools are going to be used for each part of the architecture, it needs to be decided if this architecture is going to be placed in a normal server/PC or if it is going to be deployed in cloud services.

Due to the time restriction of the project, the NLP tool is not going to be developed and, instead, it is going to be consumed using one from the market. Taking into account that most of the NLP tools are built on the cloud and focused on cloud services, the logical

action would be to act correspondingly. Also, this will solve partially the security problem, as those services will be externalised and the responsible for it will be the company who host our code.

5.2 Software tools

Now that the architecture is decided and the tools are known, we need to make more decisions. For each element of the scheme, a few tools will be compared and the one who adapts better to the proposed solution will be chosen.

5.2.1 Chat Application

This element of the architecture will be the application where the user is going to communicate with the bot.

It is important that the application is a famous one and that is available on several platforms. If the application selected only runs in a restricted number of devices, our user scope will be highly reduced and the problem will not be solved.

The User Experience (UX) and interface of the application must be easily compressible and adapted to chatbots. Extra functions as cards, buttons or in-app web browser will be very useful to our assistant.

Finally, an easy or automatic integration with the NLP tool would be extremely useful. It would reduce hours of work that could be used in enhancing the bot.

Currently there are several famous chat applications as Whatsapp, Facebook Messenger or WeChat. The latter is only famous in Asia, so it should be discarded for this case. If we must choose between Whatsapp and Facebook Messenger, the straightforward solution for this case would be Whatsapp, because it is the most famous and most used one in Pamplona, city where our client operates. But the problem is that, up to the author's knowledge, it does not support any kind of bot. The platform is closed to third party support, even if Facebook (the company who owns the app) is planning to start opening the software to other developers.

Facebook Messenger [Image 3] is an extremely good application, used worldwide and with an excellent UX. The company has spent a lot of resources in adapting his application to chatbots, and it has a very easy integration with most NLP tools. The only problem it is that it requires a Facebook account in order to use it.

Considering that Facebook Messenger meets all our requirements, this will be the selected application.

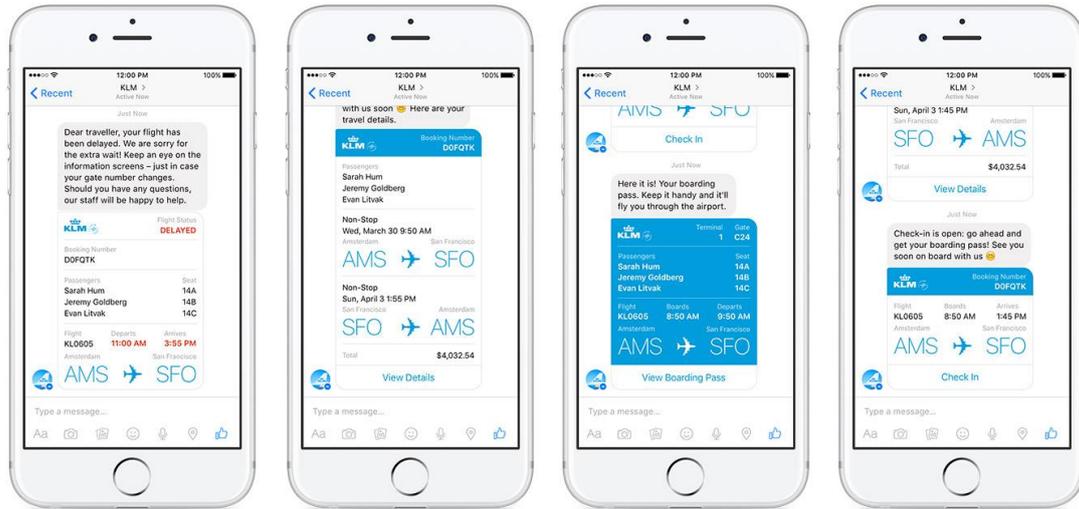


Image 3: Facebook Messenger graphic interface [19]

NOTE: This decision was taken before the Cambridge Analytica scandal. If this would have been known months before, another platform would have been chosen. Even if this application is an excellent one, this scandal has caused several modifications in Facebook Developer portal and the public image of his app has been deteriorated. Facebook API has changed and testing methods have been closed temporarily.

5.2.2 NLP

This component is the one who is going to process the text sent by the user in order to recognise his intention and extract the necessary information to accomplish his query.

The most important factor in this element is the compatibility with Spanish language. The application is focused on a Spanish client and the consumers of this app will talk Spanish, so we need the NLP tool to be compatible with this language.

Another important factor is the learning process. The tool must be able to learn from the user and identify easily what the user wants to do. A rich visual interface in order to understand better the logic behind would be something great, as this will help the developer in the development process. It is also important that the technology is not very hard to learn, but also complete enough to solve the problematic.

Finally, the price is another factor to take into account. This project cannot afford expensive licenses, as it is a demonstration for a client. This will not be a big issue, as most of the products available are free to use, or do have free versions that are sufficient enough for this case.

Alexa, one of the most important tools and the most used in USA, should be ruled out because it only works in English. Another technology that we are going to discard is LUIS.AI, because of steep learning curve.

Dialogflow and Wit.Ai are two excellent free tools that fulfil our demands. The differences between them are reduced, but Dialogflow has a slightly better learning process and an easier webhook configuration.

Dialogflow [Image 4] will be the selected tool for this element of the architecture. It is free, easy to learn, it adapts perfectly to other platforms and it has Spanish support. Also, up to the author's knowledge, it is one of the most used NLP tools on the chatbot community.

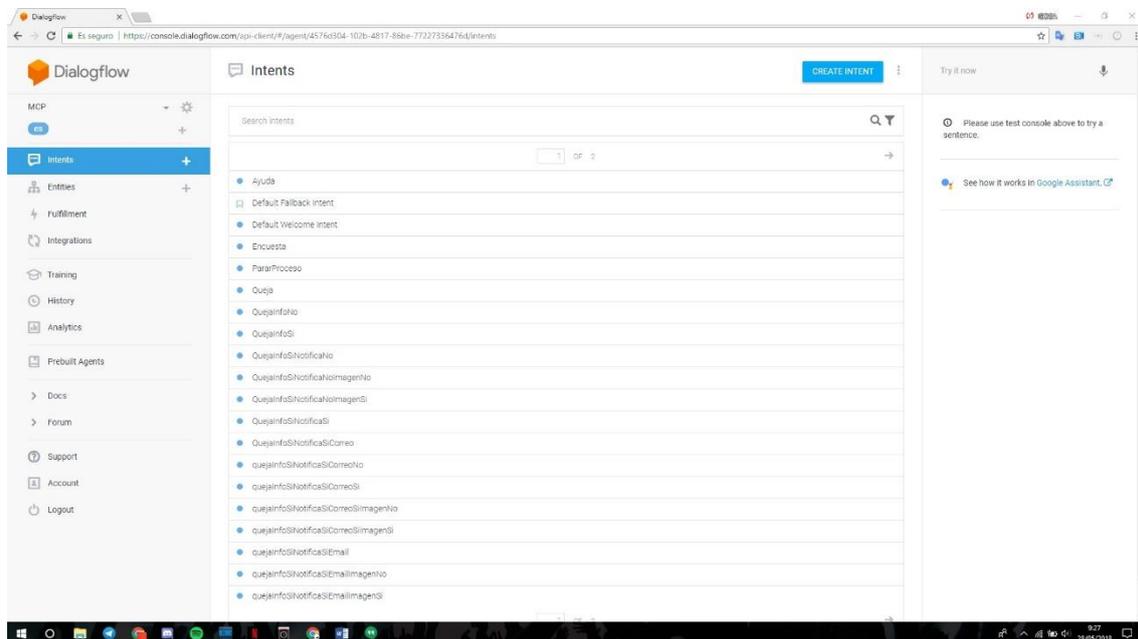


Image 4: Dialogflow console interface

5.2.3 Business Logic Server

This last component has the function of executing the logic of the chatbot. It will receive the parsed information from the NLP, process the data and send all the necessary information to the SOAP service of the client. Then it will send a confirmation to the NLP, informing if the transaction has been successful or not.

On most of the NLP tools (including Dialogflow), the Business Logic must be implemented in a webhook. A webhook is an event notification, typically sent as a POST request, which contains data that is posted to a specific URL. This URL defines the location of a script, which will process the data in the POST request. [8]

Before selecting the server where the webhook will be deployed, the language for the script has to be chosen. As the code will be somewhat simple and this is a research project, the main interest is to select a new framework/language/technology in order to test it. Finding a lot of documentation or case studies in order to simplify the development and formation time would be favourable. The actual trend is to use Javascript with Node.JS as running environment, and it could be a great option to discover this technology.

Now that the technology is chosen, the servers have to be selected. Previously it was said that the architecture it was going to be based on the cloud, so it is needed to choose a service for it. Currently there are a lot of them, but the two most important ones are Azure from Microsoft and Amazon Web Services (AWS) from Amazon. According to [20], AWS scales better than Azure. Also, AWS is cheaper (the first year is free) and it has more usage in the industry.

AWS has several services, but AWS Lambda is perfect fit for our project. AWS Lambda is a server less compute service that runs your code in response to events and automatically manages the underlying compute resources for you [21].

In order to execute the Lambda function, an API will be needed. This API will be created with another service of AWS: API Gateway. The reason of using this service is that is easy to configure, the integration with AWS Lambda is direct and the service is free.

Summing up, the business logic will be coded in JavaScript language, running in a Node.JS runtime environment and stored in AWS Lambda servers, which will be activated using an API generated with AWS API Gateway.

5.2.4 Forms

This tool will be the responsible in creating the forms and managing the answers of the testers.

After some time comparing different options, it can be seen that most of the forms tools ask a plan subscription in order to create complete questionnaires. Only Google Forms [Image 5] offers full functionality for free. This tool is completely sufficient for this project. It is capable of recovering all the answers and all the data can be analysed within the application or exported to datasheets.

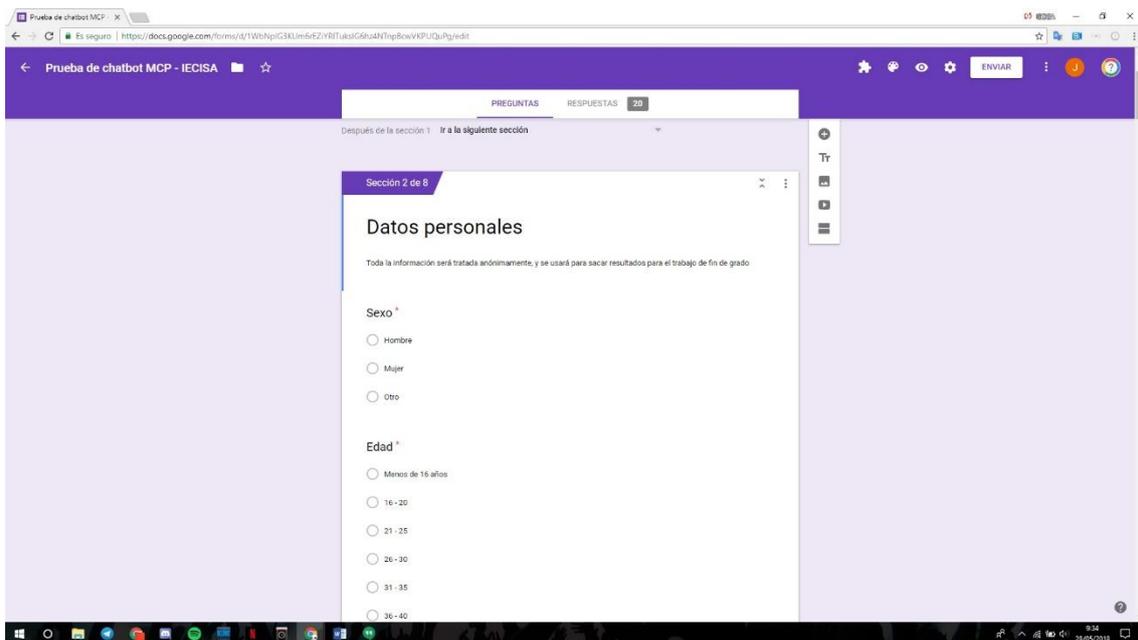


Image 5: Google Forms welcome page

CHAPTER VI: IMPLEMENTATION

This chapter will be the one where the implementation will be addressed. Here the use of each tool will be discussed, all the major design decisions explained, errors along with their solutions will be argued and the flaws and strengths of each utensil will be analysed.

6.1 Dialogflow

This tool was the first one to be used on this project. It is the Natural Processing Language (NLP), but also the core of all the logic involved in the process.

6.1.1 Description

The tool is divided in these different sections:

- Intents
- Entities
- Training
- Integrations
- Analytics
- Fulfilment

6.1.1.1 Intents

Intents are a relation between the input of the user and the action that should be taken. For each action an intent should be created.

Intents can be triggered with user inputs or with user events. When a user starts using the chatbot or sends a photo, an event is prompt. If this event is mapped to an intent, the action assigned to it will start. The same happens when the user sends a text to the tool; the NLP analyses if the text matches or it is similar to a training phrase and then executes the assigned action.

Training phrases are phrases that the developer inserts in an intent to make the NLP learn when it has to trigger the specified intent. For example, some phrases can be “I want a t-shirt” or “I would like to buy a shirt”. Then, if a user types “I would like a t-shirt” or “I want to buy a shirt”, the intent will be recognised as these phrases are similar to the ones introduced by the developer.

If the phrase does not match with any intent, the **Default Fallback** Intent will trigger. This intent has the objective of notifying the user that his phrase has not been understood.

But there is also another constraint in order to execute a specific intent, the context. The context represents the actual state of the user request. For example, a user might be asking for a t-shirt, and during the shopping process he can say “I want a red one”. In this case the tool is able to understand that he wants a red t-shirt, because the previous intent has added the **t-shirt** context. Each intent can have input and output contexts. Input contexts are the ones that are needed in order to trigger the wanted intent. Output contexts are the new ones that appear after activating an intent. So, in

order to begin an intent, you need to coincide in the input context and in a phrase or event.

An intent can have a fulfilment. This means that when an intent is activated, some code can be executed in order to accomplish a task. For example, if a user asks to a chatbot what the weather will be like in a week, a code has to be ran to query this info. The fulfilments work using webhooks, an API concept that it is growing in popularity [10]. Webhooks are a tool who delivers information as something happens. In this case, intents can have a webhook assigned and they will trigger when an intent is activated.

But in order to send the information, some parameters have to be recollected. The webhooks might need information from the user that the chatbot will gather. These parameters must be specified in the intent in order to activate the webhook only when all the information is compiled.

Finally, intents have responses. These responses are the ones that are sent to the user after an intent is activated. These ones can be adapted to each platform where the service is going to be consumed and take full advantage of the tools that each messaging application offers. For example, in Facebook Messenger we can add quick replies (little buttons where the user can click for giving a quick answer) or cards (a message with image, URL, title, subtitles and buttons with answers).

6.1.1.2 Entities

Entities are a mechanism used to extract parameter values from the text sent by the users. The idea is that a text sent by a user can be mapped to a value. For example, a user might want to buy a product and say “shirt” instead of “t-shirt”, and this is totally valid, as they are synonyms. The problem is that these options should not be handled in our code. Instead, the NLP is going to analyse the user phrases, and change all the parameters to the ones expected to finally store them.

Entities also serve as a way to detect about what the user is talking. For this case, it is going to be supposed that a shopping chatbot it is being developed. The user said previously that he wanted a **shirt** that is a synonym of **t-shirt**. And **t-shirt** (in this case) is part of the entity **Clothes**. The chatbot might have several entities, as **Food** or **Toys**. Each entity can have several entries. These entries are the ones created by the developer.

The advantage of having entities with different entries is that synonyms can be grouped by the theme of our choosing. This will help create training phrases. For example, in the shopping chatbot the next phrase could be a training phrase for the **buy** intent.

“I want to buy a **@clothes**”

Where **clothes** in the entity with several entries as **T-Shirt**, **Pants**, etc. Each entry can have synonyms.

Also, entities serve to verify the user information. If the chatbot asks the user for his personal email, if this parameter uses the **@sys.email** entity, the chatbot will only accept the user email if it follows the regular expression ***@*.*** (* = any string). These entities are the ones offered by the NLP tool.

The user cannot create entities that verify regular expressions, but he can use the ones that Dialogflow offers.

6.1.1.3 Training

This section is used to train the chatbot. The functions of this feature are limited as it is in beta.

Here you can select dialogs that a user and the chatbot had, and check if all the phrases went to the correct intent or if some of them failed and activated the **fallback intent**.

6.1.1.4 Integrations

Here the developer can start the integration of the chatbot in one (or more) application of the list.

Normally these integrations are quite simple. The developer has to configure some parameters, but it does not take a lot of time to set up it.

There are a lot of platforms available. The most famous ones are Facebook Messenger, Twitter, Skype or Google Assistant, but there are other ones.

6.1.1.5 Analytics

Here the developer can see how the chatbot is doing. Graphs can be seen where it indicates the number of active users. Also, it can be seen how many times an intent has been used and how much time it has taken to answer to the user. Finally, there is a tool called Session Flow who indicates different flows and the percentage of use of each one.

6.1.1.6 Fulfilment

In this last section, all the information of the webhook can be configured. Here the developer can introduce the URL of the webhook and the headers needed in order to use it.

Also, Google gives the possibility of using their own webhook servers (Firebase). In this case, the developer can code the functions using their inline editor. In this development this feature has not been used, as the AWS servers were the chosen ones.

6.1.2 Design

Dialogflow is the responsible of the chatbot's logic. A flow had to be designed in order to guide the conversation and [Image 6] represents it. Each rectangle denotes an intent and the rectangles with two interior lines denote a parameter of the previous intent.

The **Welcome Default Intent** is the one who introduces the bot to the user and explains him his basic functions. This is: Complaints, Help, Commonwealth Services and the Answer Form. The basic functions are showed in clickable buttons, except the complaint function, where the chatbot encourages the user to use natural language (and like that show the abilities of this system).

When the user clicks in any of the three available buttons (or writes one of the words), the chatbot activates the corresponding intent and message will be sent to the user with the demanded information.

Before talking about the main feature of the bot (complaints), it is important to talk about the **Fallback Intent**. This intent is activated when there is no match with the user

behaviour. This intent will inform the user that he does not understand what is saying and then it will offer him help.

The main feature of this software is the formulation of complaints using natural language. This is done using several intents. Each intent, once activated, inserts a new context into the user conversation. Like that, the system is able to know if the user has the info validated, if he wants to be notified with the result of the complaint or if he wishes to add an image to the file. Also, some intents demand some parameters to the user and these have to be fulfilled before passing to the next intent.

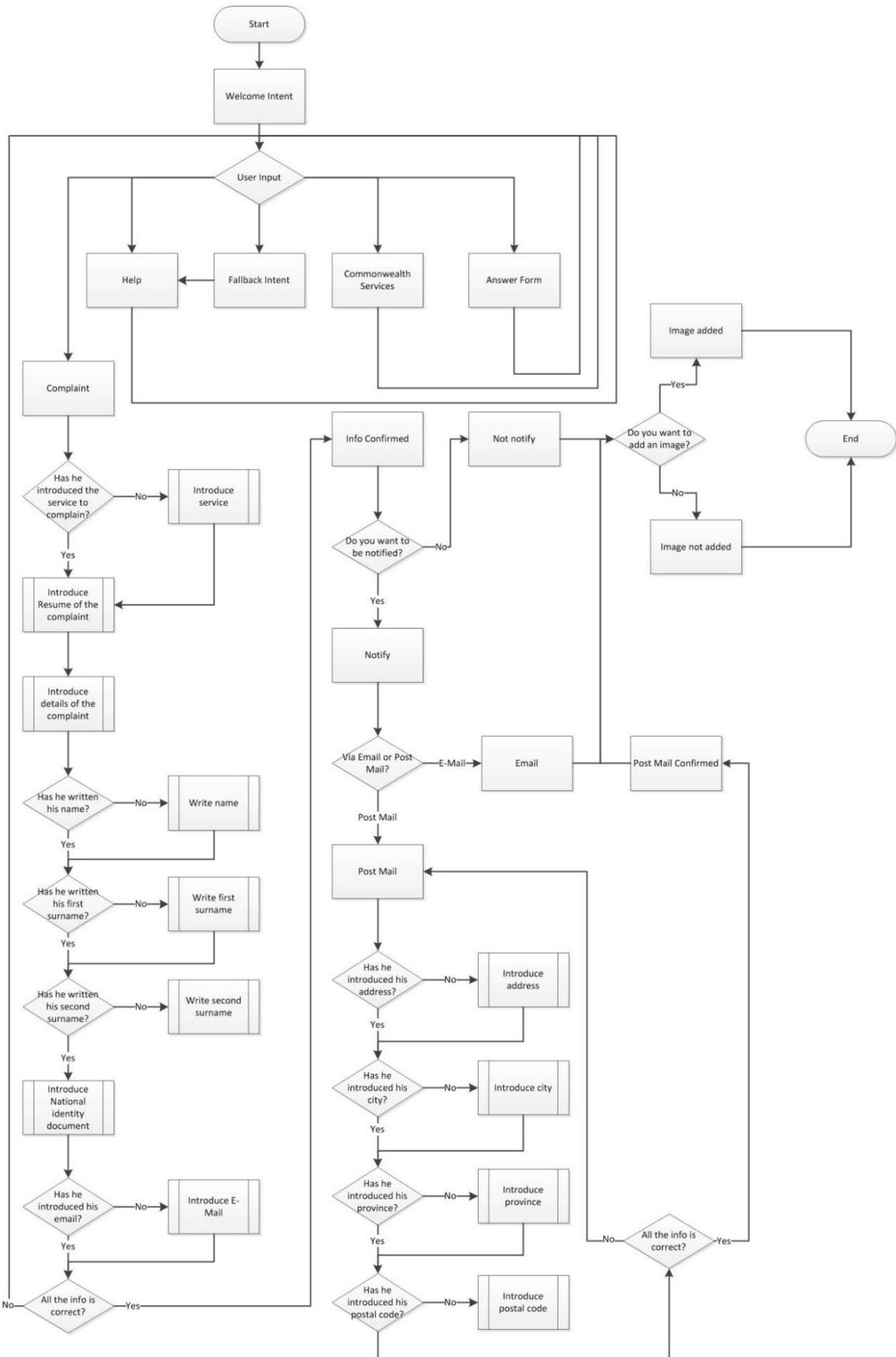


Image 6: Dialogflow Flow Diagram

The main intent **Complaint** serves to collect the user complaint and most of his personal information, as name, email and national identity document number. Once the intent **Complaint** is activated, the context **ComplaintInfo** is created. After compiling all the information, the chatbot will ask the user to check the information he introduced. If everything is correct, the chatbot will continue to the next intent.

The next intent is **Info Confirmed**, and serves to ask the user if he wants to be notified with the result of the complaint. To start this intent, the user has to confirm the aforementioned information (using the words “yes”, “correct” or similar) and have the context **ComplaintInfo**. When this intent starts, a new context **ComplaintInfoYes** is created. If the user does not want to be notified, the chatbot will pass to the **Not notify** intent, that will be explained later. If the user desires to be notified, it will pass to **Notify** intent.

Notify asks if to the user if he wants to be notified by email or post email. This intent will only activate if he answers positively to be notified and if he has all the precedents contexts. At the start of this intent, a new context called **ComplaintNotifyYes** is created. If the user wants to be notified via email, **Email** intent is activated. If instead, he wants to be notified via post mail, the **Post Mail** intent will trigger.

In order to activate the **Post Mail** intent, the user has to answer “post mail” (or similar) to the last question and have all the previous contexts. When the intent starts, the context called **ComplaintNotifyPostmail** is created. In this intent, the chatbot will recollect all the necessary information to send the notification via post-mail (address, city, etc.). Once all the information is gathered, the user will be asked if the information is correct. If it is not, the user will have to introduce his postal address again until it is correct. If it is, a new context **ComplainNotifyPostalYes** will be created and the user will be asked if he wants to send an image with the complaint. If he answers positively, the **Image added** intent will be activated. If not, **Image not added** intent will trigger.

The **Email** intent can only be activated if the **ComplaintNotifyYes** context exists and the user answers **email** (or a synonym) to how he wants to be notified. The intent will create the **ComplaintNotifyEmail** and ask the user if he wants to add an image to the complaint. The possible answers and future intents are the same as in the **Post Mail** intent.

Some paragraphs before, the **Not notify** intent was mentioned. This intent only starts if **ComplaintInfoYes** and the previous contexts exist and if the user answers negatively to the option of being notified. This intent only asks the user if he wants to add an image to the complaint and creates the **ComplaintNotifyNo** context. The possible responses and next intents are the same as the **Email** or **Post Mail** intent.

Finally, we have the last two intents: **Image added** and **Image not added**. Once again, these can only be activated if all previously contexts have been created and, depending on the answer of the user, the one or the other will trigger. Depending on the prompted intent, a context will be created: **ComplaintPhotoYes** or **ComplaintPhotoNo**. Finally, the chatbot will inform the user that the process has been finished and the info sent.

Most of the intents collect user information, as might be the user name and surname, email or the complaint motive. To verify that the user provides the information correctly, Dialogflow entities are used. In these intents several entities have been used, some

from the system and other created for this case. Most of the information has been recovered with system intents (email, address, city, zip-code). But 4 other entities were created for this case: **Names**, **Surnames**, **Commonwealth Services** and **Provinces**. The four of them served as dictionaries, to verify that the user only gave what the system expected.

The entity **Commonwealth Services** consisted of the five different entities of this governmental body: Taxi, Urban transport, Waste, Fluvial Park and Water. All of them had several synonyms; the user could say Dumpster and the system will detect that he wants to make a complaint about the Waste services.

The entity **Provinces** served to verify that the user introduced correctly a province of the Spanish state.

Finally **Name** and **Surname** entities served to detect if the user is inserting a name or a surname when asked. This was made to force the user to insert the data correctly, but mostly as an experiment to try different features of the NLP, study the user behaviour and try the raw mode, where the information could be added as an JSON or CSV file.

6.1.3 Major Design Decisions

Some important decisions had to be done during the implementation of this section. Before starting this project, the tool was studied and some small projects developed on it, to fully understand the platform and be able to design it correctly. Nonetheless, some constraints were found on this software and the design had to be adapted to these.

The most important design decision was to ask for the image as the last step of the process, removing the possibility to the user of confirming all the information that he was about to send. This decision was taken because Dialogflow and Facebook Messenger (the chat application) communicate themselves between JSON's messages, but do not handle the parameters in the same way. Facebook Messenger is able to store images, but Dialogflow is not.

The only JSON message that the developer can use is the one of Dialogflow tool. This one will have several fields: lang, id, result (with all the contexts), etc. and one of them will be the original request of the user, the message that he sent to Facebook Messenger originally and then Dialogflow interpreted. As explained previously, all the information that has to be stored is introduced in Dialogflow's contexts. The issue is that contexts are only handled by Dialogflow and not by Facebook Messenger, and just the last original user query will be available. This is not a big impediment, except when we talk about images.

As said previously, Dialogflow is not able to store images, it only detects if the user has sent one (using events). This means that if the program wants to store the image, this one should be sent in the last message, in order to be able to recover the image and send it to the webhook. If the image is sent before, Dialogflow will be unable to process it and it will be lost by the next message.

Also, as Dialogflow is not able to store the sent images, the picture will be stored on Facebook servers and the bot will receive a link where it can retrieve it. The retrieving process will be done in the webhook, which will be discussed later.

Another important design decision was the language of the bot. A lot of developers think that an assistant should have a defined personality. It is important to decide if the bot is going to have a friendly, neutral or a serious attitude. Depending what the bot wants to achieve, a personality or another should be defined. In this case, the idea is to help the citizens to do administrative task and show them the platform. It could be argued that the bot should have a friendly attitude, because this will have a positive impact on the citizens. But it should be considered that this system is designed to receive complaints and/or confidential information, and if the defined character is too friendly, it might be annoying. So, the defined will be a neutral one, with some friendly touches. These could be some answers with an emoji or a casual expression.

It was also decided that one of the first questions that the chatbot should ask when the user would want to do a complaint, it would be the complaint itself. If the user had to be forced to add all the information before explaining his dissatisfaction, he might be bored in the process and abandon the conversation. Putting the two questions related to this theme in second and third position, could motive the user to finish the form.

As said previously, it was also decided to create 2 dictionaries for the names and surnames of the users. This was done knowing the possible errors and malfunctioning that it could generate. Nonetheless, it could be a method to validate user information and it could also help to study user behaviour.

Finally, some answers were adapted to the interface of Facebook Messenger, but this will be discussed in the next paragraphs.

6.2 Facebook Messenger

6.2.1 Integration

The Dialogflow integration into Facebook Messenger was easier than expected. Thanks to the NLP tool one-click integration, the chatbot was deployed in only a few clicks.

First, a Facebook page had to be created. The chatbot was going to be hosted in this page. Then, a Facebook application was created in the Facebook Developer Console. Once the application was created (a straightforward process where an app name and a contact email is asked), a token had to be demanded in order to use the API. Obtaining this token was also an easy process. This token was needed to introduce it in Dialogflow and start the integration. Finally, it was necessary to configure the webhooks between Dialogflow and Facebook Messenger, introducing some URL's at Facebook Developer Console that Dialogflow provided. Summing up, all this process was following some easy steps and was done in 10 minutes.

6.2.2 Graphic Interface

Even if the chatbot is already working in an application, there are some elements that we have to implement in order to improve user experience and to have a better graphical interface.

Facebook Messenger offers us diverse ways of displaying messages. By default only text is shown, but it is also possible to attach images, cards or buttons, increasing the user experience. In this application, cards and buttons have been implemented.

The first are composed of an image, a title, a text, and a button that redirects to a web page. In this case, the card is used to welcome the user into the system. It shows him the logo of the commonwealth to help him understand with who is talking, and the button redirects him to the webpage of the entity, helping him (if he desires) to increase his knowledge about this politic body.

The buttons are used to help the user answer diverse direct questions. For example, if the bot asks the citizen if he wants to be notified, the suggested answers will be “yes” or “no”, and the citizen could click on one of these buttons to answer. These buttons will appear on selected questions with an easy and short response and on some menus, in order to show the user what he can do.

6.2.3 Tests

Chatbots are applications that must be revised and approved by Facebook. The company allows the developers to deploy the app into the platform to see the result of the implementation and to test it before sending it into approval. It also offers the possibility of selecting testers to give some application feedback.

As it is not legal to publish in name of an organization without his consent, all the testing and feedback had to be done selecting test users. This would have been a problem easily solvable using the different testing methods that Facebook offered, but during the final part of the development, Facebook was accused of several security problems (Cambridge Analytica) and it had to remove several of these tools. At the end, there was only one way to test the chatbot.

The system forced developer and testers to become friends on Facebook. Then, the developer had to assign all the auditors as collaborators of the Facebook page where the bot was published and as testers of the chatbot application per se. Finally, the auditors had to register themselves as Facebook developers, introducing their phone number or credit card. As this method was bothersome, it was decided to prioritize interviews, giving the opportunity to test the system in the developer’s devices.

6.3 AWS Lambda and Client SOAP Calls

6.3.1 Lambda Function

To execute the business logic, a function has to be programmed. As previously explained the function was going to be developed in Node.JS language and stored in Amazon Web Services (AWS), concretely in their product AWS Lambda.

In order to achieve this task, an account was created in AWS and a function in the Lambda tool. Even if the AWS website gave the option to code directly in a web editor, if the developer wanted to use some Node.JS modules (like the soap module used in order to make soap calls), he had to pack all the code into a zip and upload it.

The function is a long one, but not a complex one. The first thing that is done is recover all the information from JSON sent, and store them in variables. Then, with the help of the contexts, the decision of the user is detected and the info modified in order to fulfil the SOAP message that is going to be sent. Before issuing the message, the image that the user sent has to be downloaded and stored into a blob variable.

There are 4 soap calls to make: i) **createDraft**, ii) **attachDocument**, iii) **validateDraft** and iv) **registerRequest**. These 4 soap calls require different pieces of information. The first one, **createDraft**, is where all the citizen information is asked (name, complaint, email, etc.) and a draft code is given as output of the call. The second, **attachDocument**, is where the image is uploaded (in case there is one). This is done passing the blob variable and the draft number to the call. Finally, the last two methods, **validateDraft** and **registerRequest**, realise internal processes and the only parameters asked is the draft number.

The initial idea was to make all the soap calls, and then, send a callback to announce to the bot that the fulfilment has been finished. But this has been impossible because of the time that the service takes to make these calls. The soap system is not fast, and the chatbot needs a nearly instant answer. The decision taken was to realise only the first call, make the callback, and then realise the rest of the soap calls.

6.3.2 SOAP Service

The implementation of this part was one of the slowest, because the soap calls were badly documented and the service was not well deployed. The connection was made using node-soap module and it was necessary to realise 4 calls to complete the procedure.

During the implementation development there were several problems. The first and most important one was the lack of documentation and knowledge of this service. It was part of a bigger product of the company and it was not been used in this office. This, with a lacklustre documentation, made this task a complicated one. Several variables had to be guessed and other has to be searched on large databases, as most of them were totally undocumented.

Also, the service I was going to use, worked behind a VPN (Virtual Private Network). It was necessary to deploy a machine with this utility outside the VPN and configure it correctly. This did not affect a lot to the development time, as I could program the function on a local environment.

Finally, once the system was deployed in a public machine, the code was copied to AWS with all the corresponding Node.JS modules. It worked correctly, but it was slow.

To sum up, this part took more time than expected because of the enterprise system. It used old technology, it was slow and mostly undocumented.

6.3.3 Endpoint API Gateway

Once you have the function on AWS Lambda and a chatbot who generates JSON and send it via POST's messages, it is necessary to communicate both pieces and make work them together.

For this, AWS API Gateway tool was selected. It was an easy procedure. First, an API was created, giving it a name and a description. Then, an endpoint had to be built, so a POST method was generated and configured in a way that each time that this method was called, the Lambda function started. The configuration was easy; only a straightforward web form had to be fulfilled.

Finally, the URL of the API call had to be introduced in the Fulfilment tab and select what intents were going to call this API. In our case, these were the last intents, where the user was asked if he wanted to send a photo.

6.4 Forms, Interviews and Testing

6.4.1 Forms

In order to collect the opinion of the testers, two forms were made. The first, to know how to contact the tester and explain him how to test the bot, and the second to recover all the testers feedback.

The first form is a consequence of Facebook scandal. As most of their testing methods were stopped or unavailable because they were modifying their systems, it was impossible to give an easy way to evaluate the bot. How this was resolved will be discussed below. In this form it was asked the full name and the email of the tester and if he had a Facebook account. In the case he had one, if he was friend of the developer or if he wanted to be. With this information, it could be decided how this tester was going to test the application.

The second form is the form where the tester will express their opinion about how the chatbot works, if it is useful, their point of view regarding administrative portals, open government and if the chatbot has simplified all this process. There were 7 sections in the form, where different questions were asked:

1. Personal data: Sex, professional sector, age
2. Bot functionality: How the bot worked
3. Personal opinion - Bot: What do they think about the idea
4. Bot improvements: What should the bot do in the future
5. Personal opinion – Administrative portals: What do they think about administrative portals
6. Use of administrative portals before bot usage: How much did they used administrative portals before using the bot
7. Use of administrative portals after bot usage: How much are they going to use the portals now and if they would continue to use the chatbot.

6.4.2 Testing

As it was said previously, Facebook changed their tools and policies because of the Cambridge Analytica scandal. This means that the testing phase was complicated.

If a tester wants to try the bot, he has to be friend with the developer of the application. The developer has to make him administrator of the Facebook page where the bot is stored and then invite him to test the application. Then, the auditor has to register as a Facebook Developer. Finally, he will be able to test the application.

This is a long a complex process for an everyday Facebook user without a high knowledge in technology. That is why it has been given the possibility to the testers to try it on the developer mobile phone, doing an interview.

6.4.3 Interviews

The interviews will be very simple. The developer will give the mobile phone (or website) with the chatbot showing the welcome message. After that, the auditor will have to learn how to use it help and interact with it without the developer's help. If the tester does not have the idea to start a complaint, the developer will encourage him to do one, but without explaining how to do it.

Then, the tester will be asked to fulfil the questionnaire. If he needs help with the meaning of a question, the developer will be able to answer his questions

Finally, the developer will ask the tester if he has anything more to say, like ideas, how to fix it or other opinions that he could not detail on the form.

CHAPTER VII: RESULTS

In this chapter results will be analysed in four different categories. Evidence will be interpreted and discussed, seeking the understanding of this data.

All the information has been retrieved using questionnaires given to these testers. Then, it has been transformed into pie and bars charts, that will be the ones analysed.

7.1 Demographics

There has been a total of 20 participants in the testing phase. The number of participants has not been as big as wanted, but the problems that were encountered during the development have complicated the task.

According to [Image 7], three-quarters of the participants are male. This is because most of the testers are author's computer science classmates, and the vast majority of computer science students are men (only 20% of students are woman, and the trend is that is going to be worse in the next 5-10 years according to [22]). Also, looking at [Image 8] it can be seen that a 25% are ICT workers, where there is also a dominance of male gender.

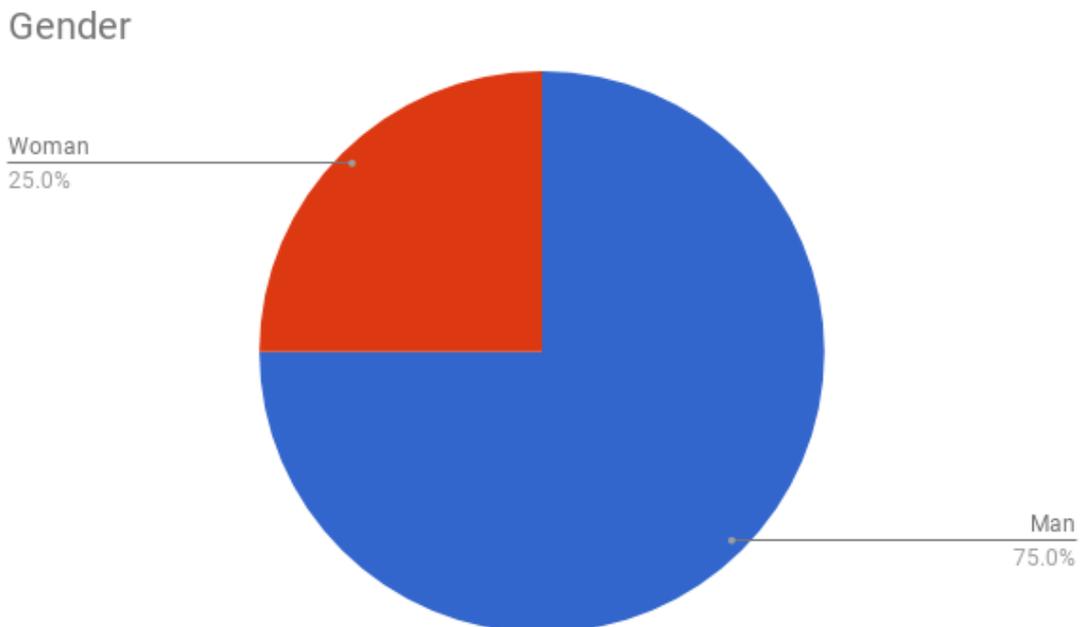


Image 7: Testers Gender.

Professional Sector

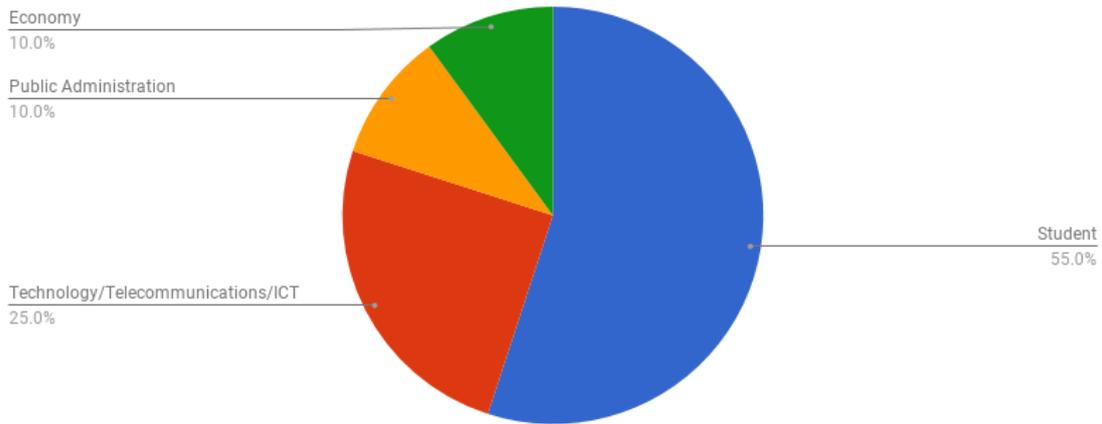


Image 8: Tester's Professional Sector

[Image 8] shows that a 55% of the surveyed testers are currently studying. Then, it is not a surprise that looking at [Image 9], 80% of our testers have between 16 and 25 years old.

Age

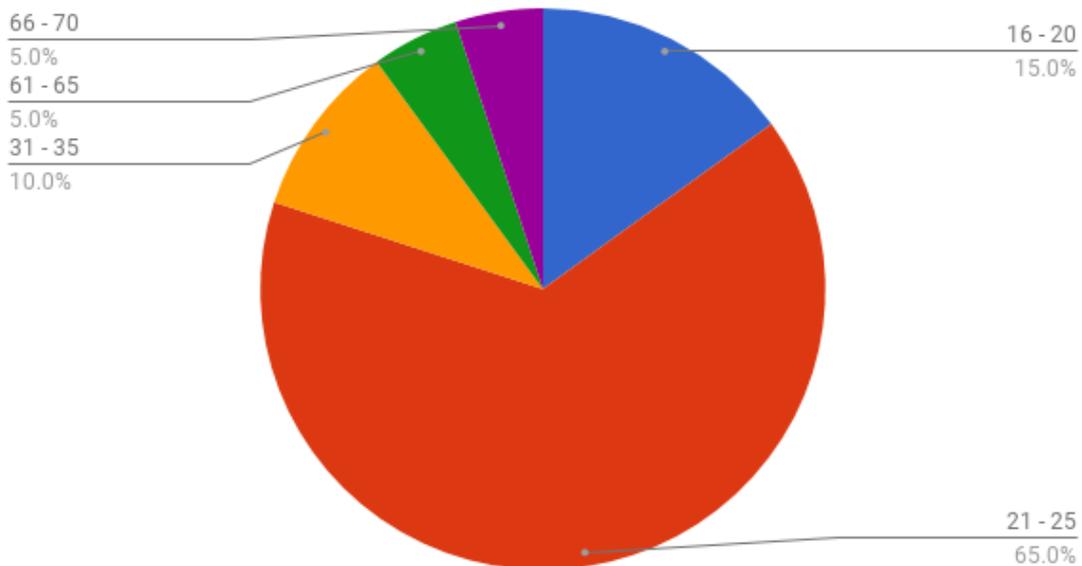


Image 9: Age of the testers

It could be said that this data is biased, as most of the testers are young and use technology in a day to day basis. This is true, as it does not represent the general public, but the obstacles found during the development of this investigation have complicated the task. Even if a lot of efforts were put in tutorial creation and simplification of tasks to get the maximum number of participants, the numbers of steps needed to test the app on a personal smartphone were too big. Most of the testers lost interest in the application when they saw what they had to do. Interviews were effective but they took a lot of time.

Nonetheless, it continues being valuable data. In order to know if this tool is going to work in general public, it has to be tested in an “expert” community. Young people, students and ICT workers are usually the first to embrace a new application or technology, and then it spreads to the rest of the people [23]. If it is wanted that this chatbot is used by the vast majority of the commonwealth citizens, it has to be appealing to the aforementioned community. Then, these will be the responsible to spread this product between their relatives and friends.

7.2 Chatbot

Even if the chatbot development was only a demonstration, the functionality of it had to be measured. This is because, in case the tested people did not like the tool, it had to be analysed if it was because of a bad quality demonstration or as a result of a bad idea. It is important to note that all the participants knew that it was a demonstration and that not all the chatbot functionalities had been fully developed; the text recognition was limited and a lot of tester queries might not be understood.

The final result of the chatbot integration into Facebook Messenger can be seen in [Image 10].

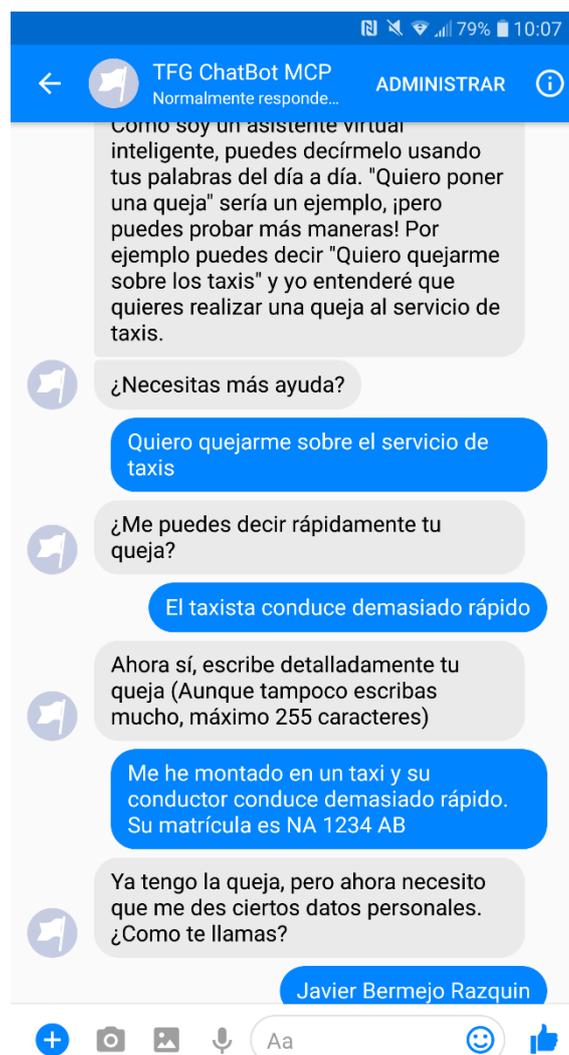


Image 10: Chatbot app

As it can be seen in [Image 11], 70% of the messages were mainly recognised by the chatbot and only a 30% of the messages had to be modified in order to be recognised. Even if merely a 35% of the surveyed said that the chatbot totally understood their messages, the important fact is that the 100% of the auditors arrived to fully test the application. This is also seen in [Image 12], where a 75% of the testers have arrived to submit without any problem one or more complaints. It can be said that the demonstration worked and the testers could interact with the chatbot to try this new tool.

Has the bot understood all your messages?

- The vast majority of my messages have been understood
- Yes, he has understood everything
- I had to modify some messages in order to be understood

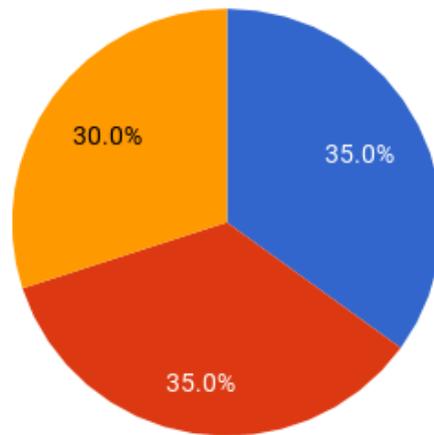


Image 11: Percentage of messages understood by the chatbot

Did the chatbot process all the complaints?

- Yes, he did process all the complaints
- The vast majority of complains have been processed
- I had some problems that I could solve doing some changes in my request

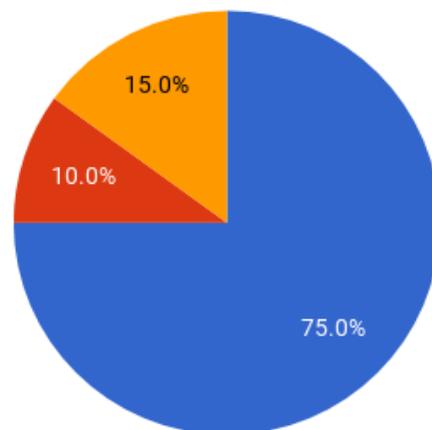


Image 12: Complaint feature

According to [Image 12], third-quarts of the testers have arrived to fully send a complaint with any kind of problem. Only 10% have had little problems and a 15% had to reformulate their complaints in order to finish them.

The people that had to rewrite their complaints have said that their majority of problems came from the name and surname verification. Previously is explained that this was introduced for several reasons, but finally this has created more errors than solutions.

There have been a reduced number of testers that said that the chatbot was not very intelligent and they had to reformulate phrases, transforming them to ones that look more like the standard one. This is true, but it was explained that it is a demonstration and that all of the features are not fully developed. Nonetheless, this should be noted for future tool upgrades, as the people do expect a fully intelligent agent.

It is not a surprise that most of the people have understood the limitations of the tool, as a 90% of the testers think that the help and indications given were enough. This can be seen in [Image 13], where only 10% of the testers did not fully understand the help given by the agent. Nonetheless, the 100% of them arrived to fully test all the features.

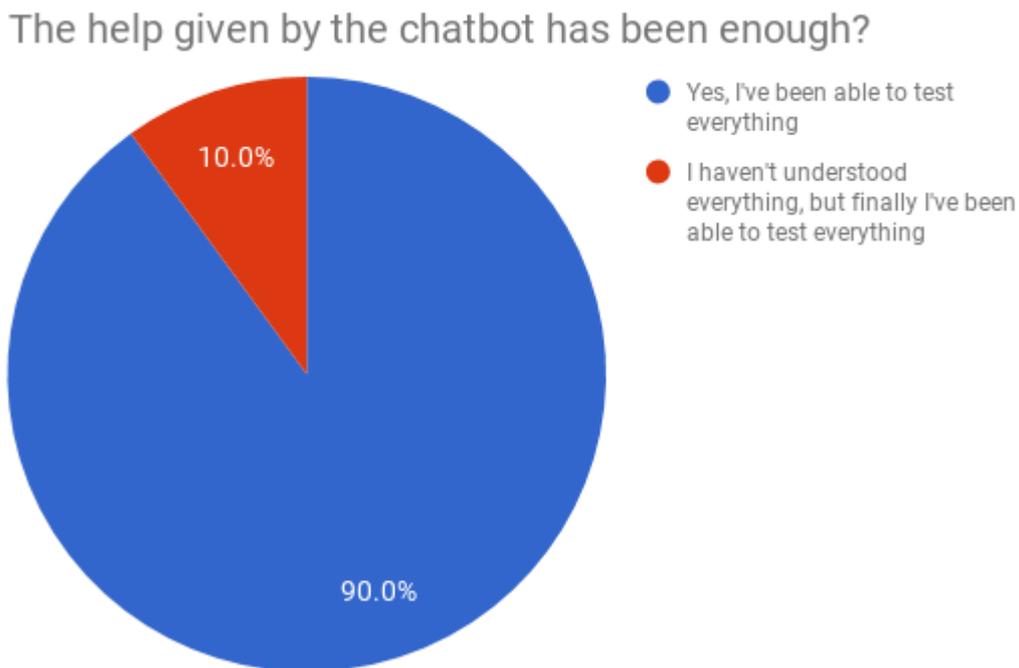


Image 13: Chatbot help

Even if we cannot see it in the charts, a lot of people were initially confused with the messaging platform, as it was the first time they used it. In Pamplona, city where this tool was tested, and to the best of this author's knowledge, most of the people use Whatsapp Messenger as messaging application. They are not habituated to use this application, which has more features but it is far less simple than Whatsapp. Once they started understanding the messaging program, they felt more at ease talking to the chatbot and could interact easily with it, ameliorating the comprehension of the information given by the agent.

Another important fact that it is not directly seen at the charts, but it was easily seen at interviews, is that most of the people did not read what the bot says. When a user starts the conversation, an initial (and long) message is sent to him, in order to explain this new tool. It was seen that most of the testers did not read all of it, and then felt a little lost during the testing.

The only proof of this is [Image 14]. On it, it can be seen that the testers did not know to what services they could complaint using the chatbot. In the questionnaire they were asked what the available services were. Out of 20 testers, only 12 knew that they could process complaints about the taxi service. And this was the most recognised service. In [Image 15] it can be seen how a 55% of the testers only recognised one service of the 5 available, and only a 20% selected all of them.

This might be the result of two reasons. First, the auditor had to ask to the chatbot what services were available in the complaint feature, as they were not prompt initially. Secondly, as this is a known government entity, the testers might think (wrongly) that they know all of the services.

What were the available services in the complaint feature?

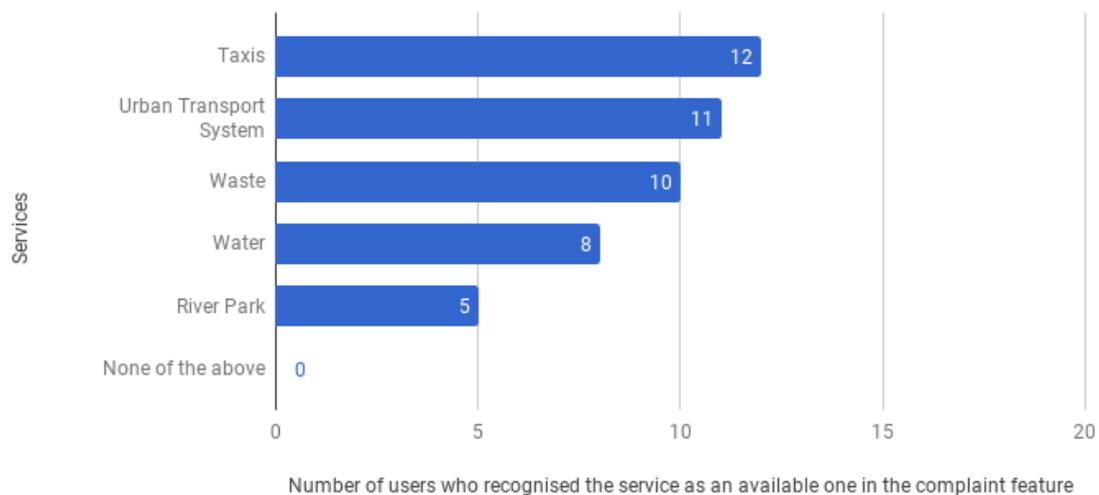


Image 14: Available services

Total number of available services recognised by the testers

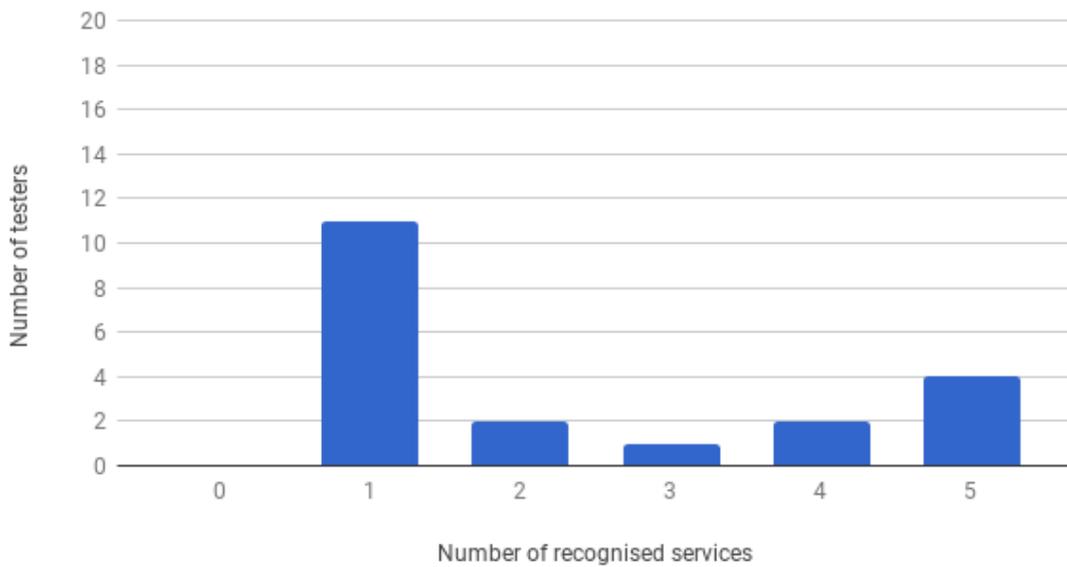


Image 15: Number of services recognised by the testers

As it was said previously, the chatbot needed a personality. This can be shown in the vocabulary and language used by it. The intention was to develop a personality where the user was found at ease and where he understood all of the messages sent by the bot, in order to explain correctly to the user what he had to do to fulfil his desired action.

According to the data recollected and looking at [Image 16], 90% of the testers think that the chatbot uses a correct language. 5% of them find this vocabulary too informal and another 5% of them find that the chatbot should be more specific when asking the questions. The tester satisfaction with the language is also found at [Image 17], as 80% of the testers say that they have understood everything the chatbot said. Nonetheless, is important to note that 20% of them had problems with some terms. It can be said that most of the testers are satisfied with the language used, as a big majority understood correctly the bot and they agree with the vocabulary of it.

What do you think about the language that the chatbot uses?

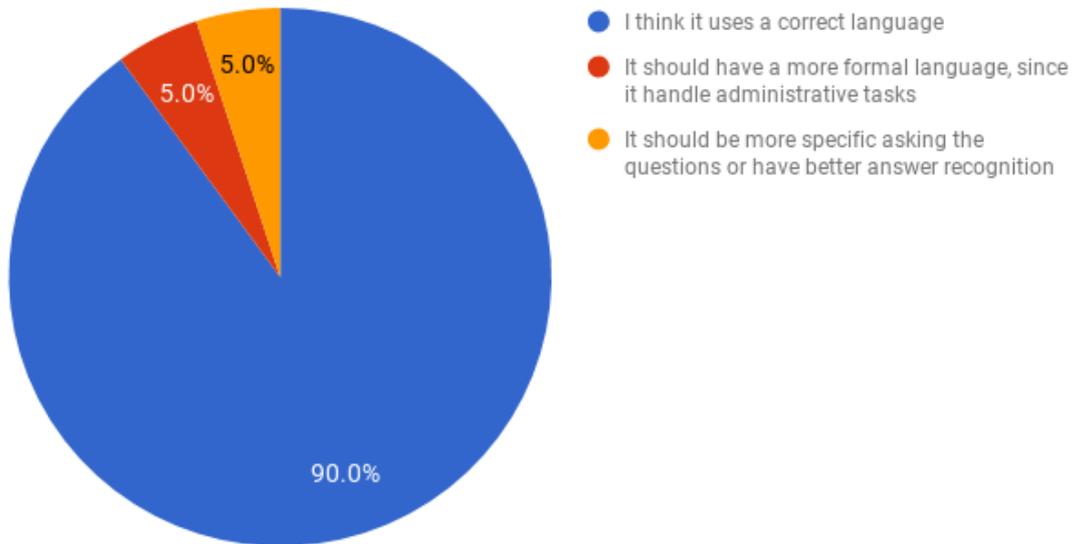


Image 16: Chatbot language

Have you understood the chatbot? Did you find the vocabulary too technical?

- I have understood everything
- I have had some problems with some terms

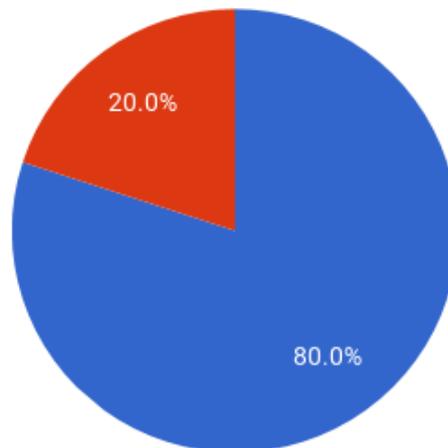


Image 17: Chatbot vocabulary

Recapitulating, it can be said that the developed demonstration has already fulfilled part of the initial objectives. The application works and it has been produced in less than 4 months. It has served to study the developing complexity of an application of this type and to test it on a selected group of people, to study if the proposed solution works for this case.

7.3 Idea and Future Improvements

Once proved the chatbot worked well enough to demonstrate the proposed solution, information has to be analysed in order to see what testers think about the idea.

The testers were asked about what they thought about the idea and the tool. The results can be seen in [Image 18]. 85% of the testers think that it is a good idea to use this type of tool for administrative tasks. The remaining 15% of them think the same, but they state that there are several things to polish. This is also seen in [Image 19], where 20% of the testers think that more functionalities should be added to this particular chatbot in order to find it totally useful. The general consensus about this idea is a positive one, even if there are certain details to correct.

Do you think it's a good idea using this tool for administrative tasks?

● Yes, it's a good idea ● It's a good idea, but there are things to polish

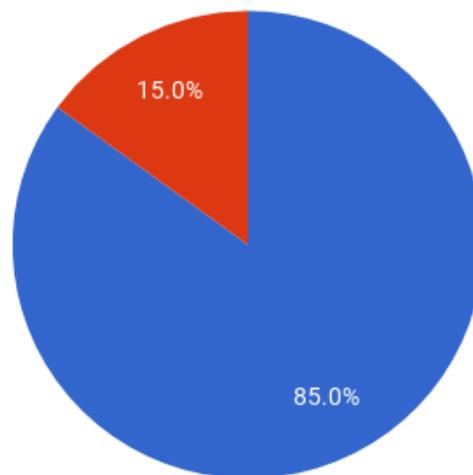


Image 18: Chatbots for administrative tasks

You find useful processing complaints to the commonwealth via a chatbot?

● Yes, it's very useful ● Yes, but it should have more functionalities

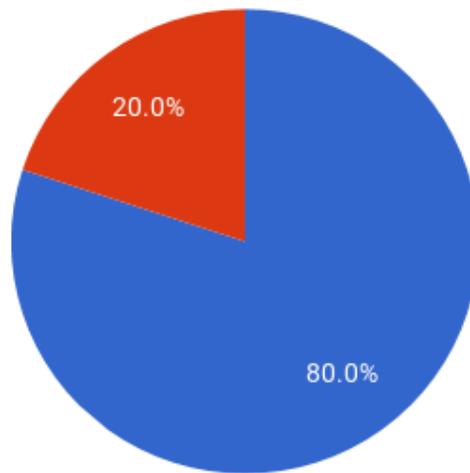


Image 19: Chatbot as a tool for the Commonwealth of Pamplona

As this application is a demonstration and its capabilities are limited, it can be supposed that most of the auditors have found this tool useful by the reduced time needed in order to make a complaint. This can be seen in [Image 20], where a 60% of the testers gave 4 points out of 5 when rating the time needed to complete this process. The average score was 4.05 out of 5.

How fast have you found the complaint process? (1 Slow - 5 Fast)

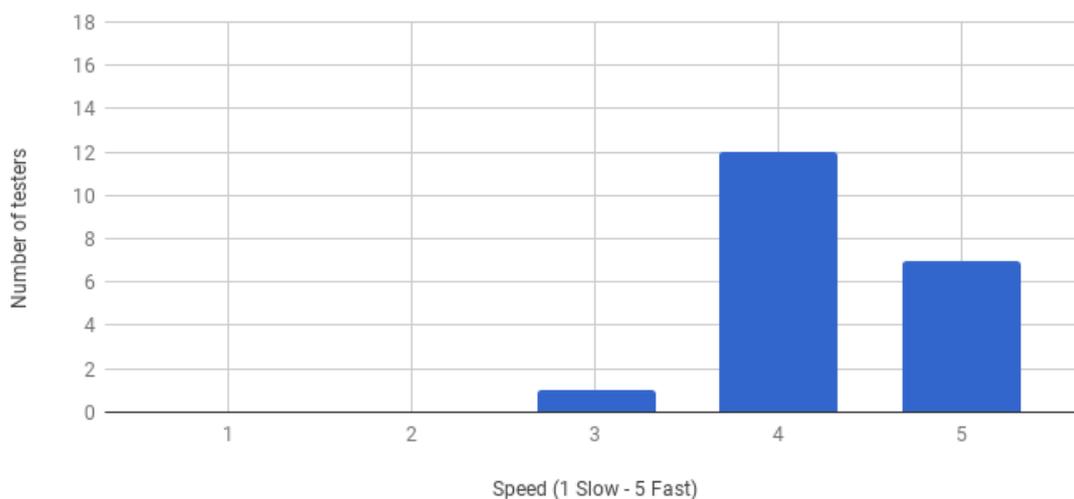


Image 20: Time needed in order to process a complaint

Also, the testers had to answer questions about future updates and new features. In [Image 21] results are shown. It is interesting to see that the most voted future features are the ones with less administrative problems. Bus subscription, their arrival times or taxi information are problems easily solvable, but most of the testers would like to be

helped by the assistant when processing these requests. Other less known but more complex administrative procedures were voted, but with huge difference with the aforementioned ones. Managing the bus subscription was a feature asked by 70% of the testers, but managing administrative payments was only asked by 30% of them.

What new features would you like to see in future updates?

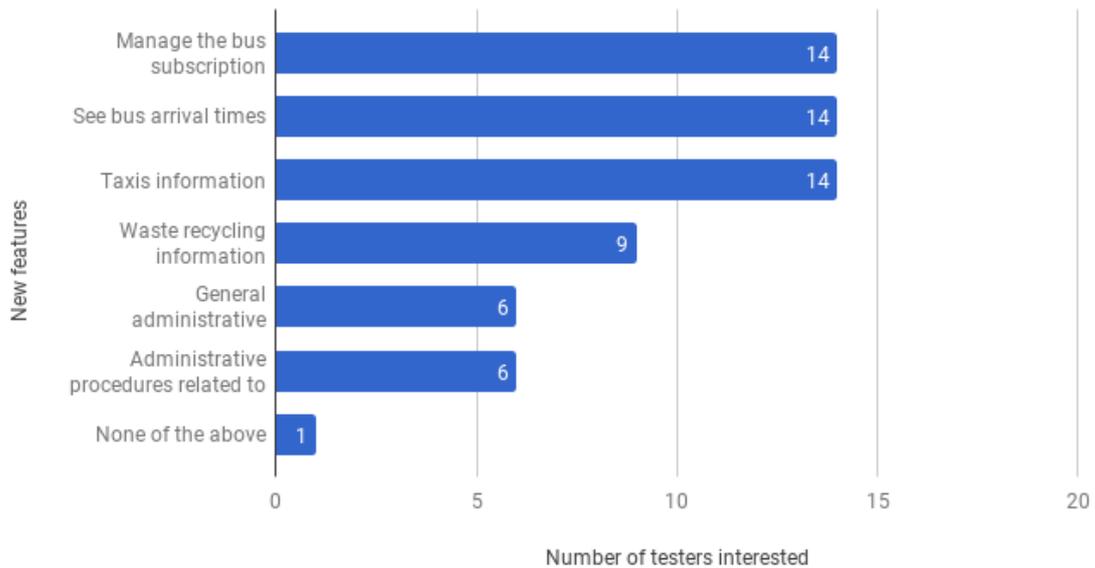


Image 21: New features

Finally, it was asked where the testers wanted to use the chatbot. This can be seen in [Image 22] and, expectedly, nearly all of them (a 90%) wanted to be able to access it in Whatsapp. This makes sense as it is the most used messaging application in Spain [24]. Not a lot of testers wanted this bot added to Google Assistant or Siri (only 20% and 25% respectively). This is surprising, as these are similar technologies and would work perfectly together. The only difference between them is that virtual assistants work with voice commands and chatbots with text messages.

In what platforms would you like to use the bot?

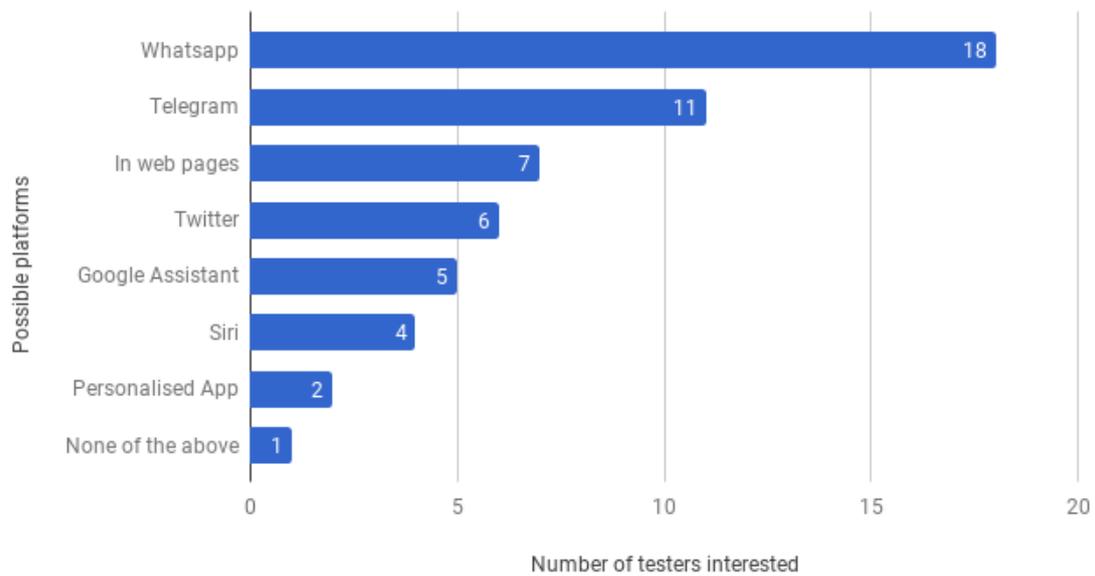


Image 22: Platforms preferred

Summing up, testers think that this tool is a good idea in order to realise administrative process. They find it useful and they are open to try new features and use it in other platforms.

7.4 Administrative Portals

Testers were asked several questions about administrative portals, looking to study if there was a usage change thanks to the chatbot. The first half of the questions was focused on their knowledge and opinion on these portals. The second half was about how much did they used them and if the chatbot contributed to increase the use of them.

The results, and concretely [Image 23], show that 60% of the testers did not know the existence of these websites until now, discovering them at the moment they started using the chatbot. Looking at [Image 24], it can be seen that an 80% though that they were useful websites, and only 20% considered that these were too complex to use. Testers say that the processes are too complicated, ask too much information and sometimes not all needs are met in order to finish a process and the citizen has to finish it on the city hall.

It is important to consider that this information may not be fully reliable, as in [Image 23] a lot of the auditors do not know what these websites are, but then they react positively without having used them [Image 24]. Surveyed might think these are a good idea, but without having tested them they cannot give a final opinion.

Did you know about the existence of these administrative portals?

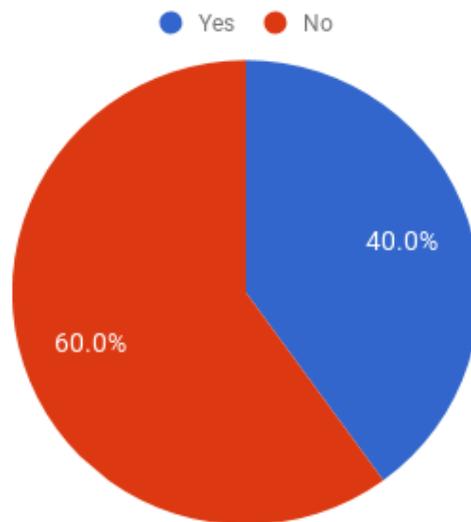


Image 23: Existence of Administrative Portals

Do you find useful these administrative websites?

● Yes, they solve a lot of problems. ● I find them useful, but they are too complex.

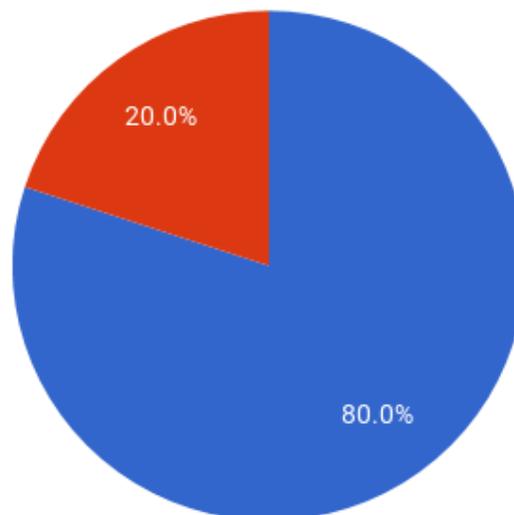


Image 24: Usefulness of Administrative Websites

Another question asked was if when doing a procedure how they prefer doing it. [Image 25] shows that a 55% of the testers always try to do it online and only a 10% prefer doing it in person. This is surprising, as the initial problem is that public administrations find that too many people choose to do this type of paperwork in their offices. From this data it can be extracted that people are motivated to use these online systems, but the complexity of them stops them from using these tools.

When you do an administrative process, you prefer doing it online or in person?

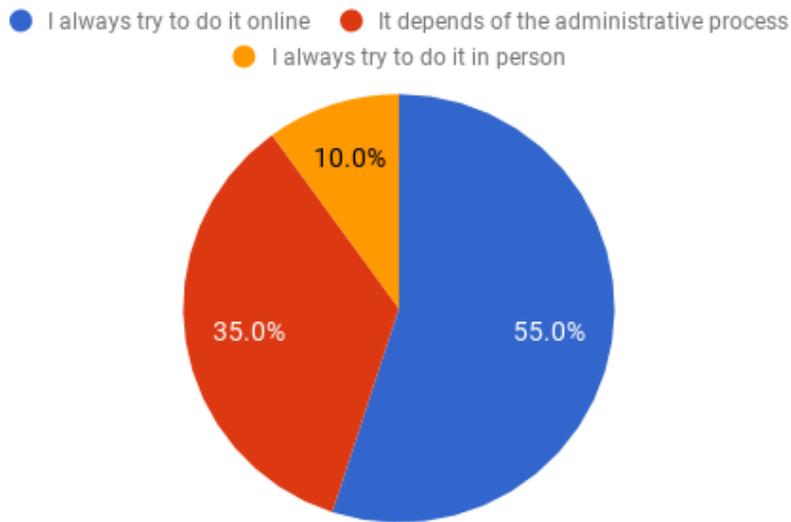


Image 25: Tester's preferences when doing administrative processes

To the best of this author's knowledge, certificates and/or identity card reading systems are common requisites in order to access to all the functions of administrative portals. This is why testers were asked if they owned one of these certificates or reading systems, and the answers can be seen in [Image 26]. Only a 15% of the surveyed own one of these systems and 35% of them finds these systems too complex to install. These results are very poor if public administrations want to impulse open government systems.

Do you have a digital certificate or a system to read your electronic ID?

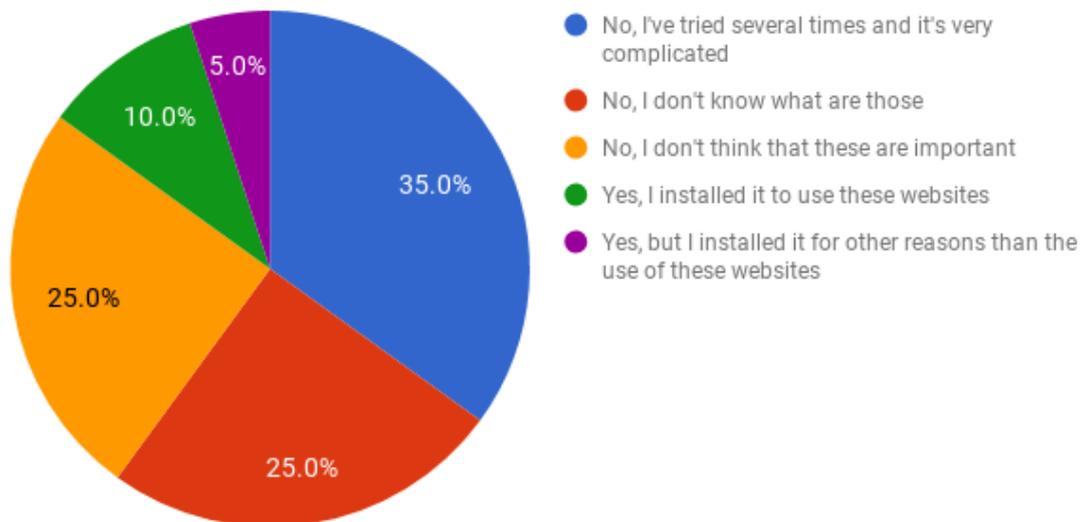


Image 26: Certificate ownership

It was also asked to the surveyed if they had ever used one of these portals. [Image 27] shows that only a 45% of them did. It has to be noted that in [Image 23] a 40% of the surveyed say that they do know these websites, but in [Image 27] a 45% of them have used these, which is impossible (In order to use something, you have to know what it is). This might be the result of a tester who did not understand the survey, or has misinterpreted the question.

Do you use or have you used administrative portals/websites?

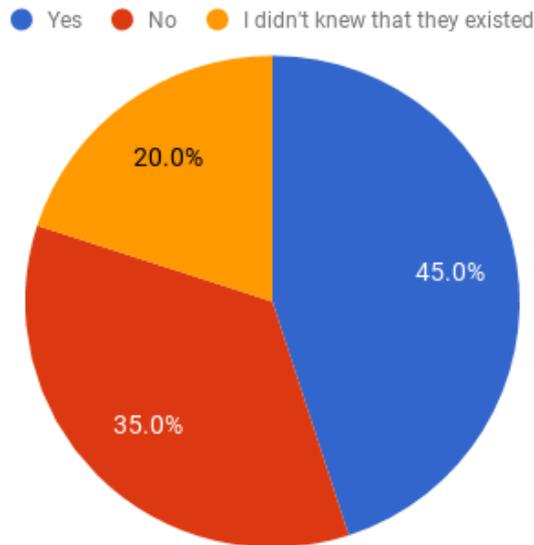


Image 27: Usage of administrative portals

Following the last question, surveyed were asked how much did they use these portals. In [Image 28] it can be seen that there is a low user adoption. It is normal that these websites are not used every day, but nonetheless the median rate is 1.8/5 points, which is very low.

How much do you use these administrative portals? (1 Nothing - 5 A lot)

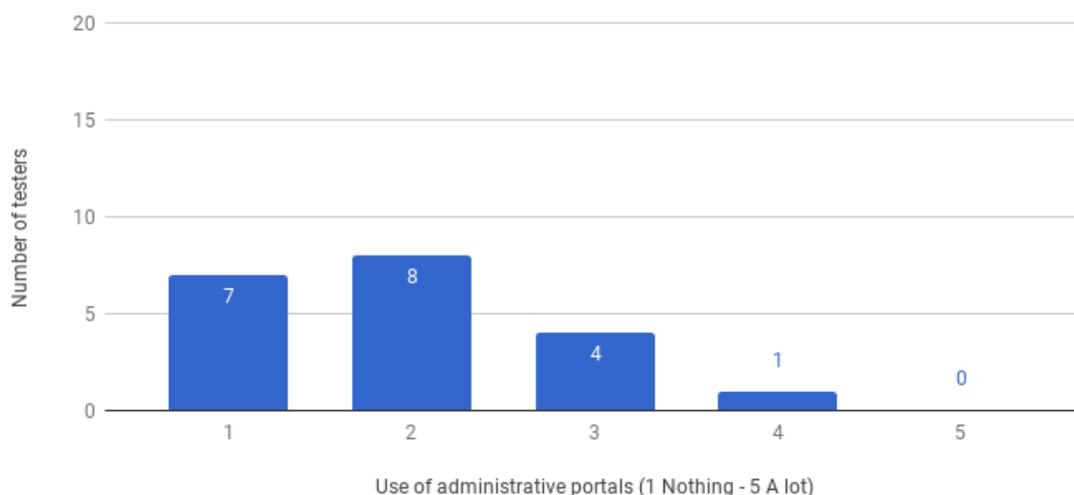


Image 28: Periodic usage of administrative portals

Now that it is known how much these portals are used, it can be seen how much the bot has modified tester's vision on administrative portals.

The first question asked was if that now that they have tested the chatbot, they would continue using it. [Image 29] shows that a 50% of the testers would continue employing it and a 40% of them would only do it if more features are added. A 10% would not use it, as they are not interested in these portals. These results are positive, as a 90% of the surveyed are motivated to continue using this tool.

Would you continue using the bot?

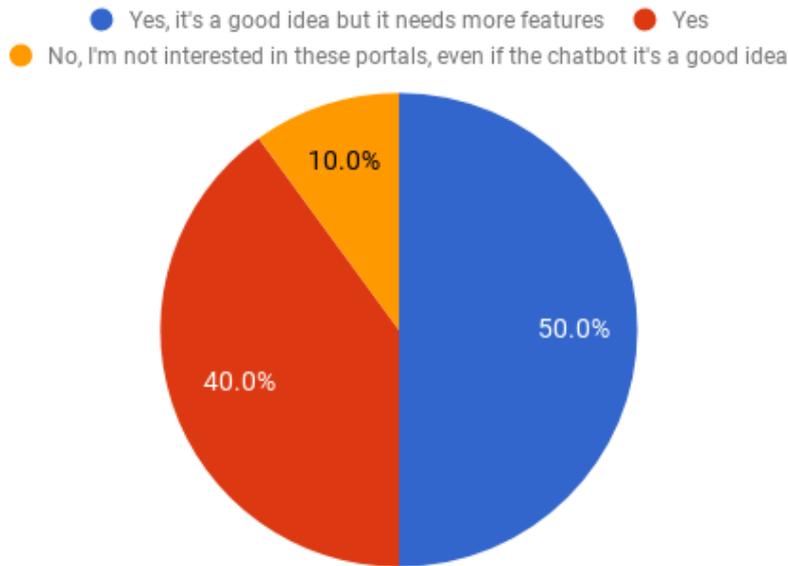


Image 29: Bot usage retention

Testers were also asked if they would do more administrative processes online if they could do them via the chatbot, and the results can be seen in [Image 30]. Only 5% answered that they would not be interested, as they do not like this tool. The other 95% would do administrative processes using the chatbot; but only a 40% would do all of them with the chatbot. This may be because people may be afraid of doing these processes on a messaging application. The important fact is that the big majority of them are open to start managing these processes on their mobile devices.

Would you do more administrative processes online if you could do them via the chatbot?

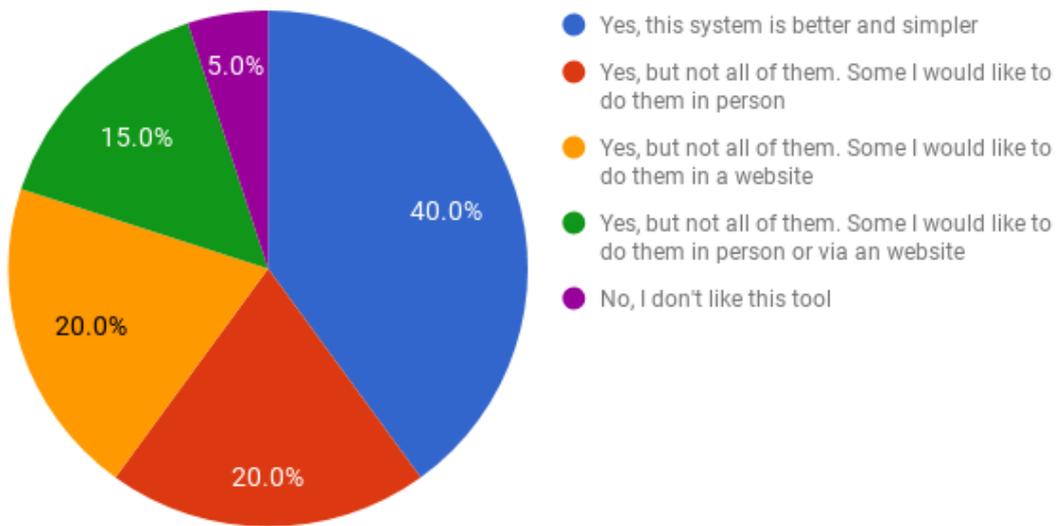


Image 30: Realise administrative processes via chatbot

Finally, testers were asked if after trying the bot, they would use administrative portals. According to the data collected (that can be seen in [Image 31]), 90% of the surveyed would use these portals, as they used them before or because it is something new that they do like. 10% of the surveyed are not interested in administrative portals, even after using this new tool.

After trying the bot, are you going to use the administrative portals?

- Yes, It's something new I didn't knew
- Yes, I used them before
- No, since I am not interested in them

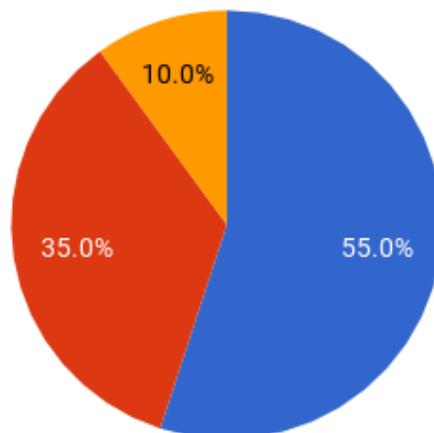


Image 31: Usage of administrative portals after trying the chatbot

Analysing all the data as a whole, an important conclusion can be extracted: People are interested in these platforms but the vast majority do not know them; and the ones that do find them too hard to use. The idea of managing administrative processes easily is appealing to the vast majority, but the results show that the testers do not like how the traditional tools or the system works. Also, [Image 29] and [Image 31] show that, not only the testers are interested in continuing using the chatbot tool, but that this tool has helped to increment the awareness of these portals and websites.

CHAPTER VIII: CONCLUSIONS

Now that the results have been analysed, some conclusions will be drawn. Some of these conclusions will be obtained from the results, but others will be extracted from the author experience when developing and testing this tool. This chapter will be divided into two sections, one focusing on the chatbot development and the other centred in the proposed idea as a solution and administrative portals as a whole.

8.1 Chatbot Development and Design

The first conclusion from this research is that a chatbot has to be prepared to be easily accessible, as it is completely new tool. It has to be available in one or several famous messaging applications, in order to facilitate the task to the user. Downloading an app prolongs the process and may discourage new users to test this tool. It is also important that the agent introduces itself, explain what its functionalities are and, step by step, show his capacities to the user. During the interview, it was clearly seen how the testers were initially confused by the application; they did not know if they had to talk to it, send it a message or wait. Only once they did read the initial information, the testers knew what to do. This is why the agent must present itself. And it is very important that this is done progressively, because if the chatbot sends a big message it is highly probable that the user does not read it.

And this is the second conclusion that can be extracted from this research: People do not read. When a tester started the chatbot, the application sent a long message to inform the tester of how he could use this new tool. In the interviews people stopped reading immediately and started asking questions, and the interviewer had to encourage the testers to continue reading. Proofs showing that most of the testers did not read in depth the instructions are [Image 14] and [Image 15], where the surveyed were asked about what services they could complaint, and only a small number of them detect correctly all the available services. If the user is faced with several messages and complex rules, he is not going to use the tool.

And this is another important rule to follow when building a chatbot, it has to be simple. The testers were able to test the chatbot because the features were limited in number and complexity. The agent must keep things clean and fast, and if a lot of steps are needed, these should be managed trying to not saturate the user with too much information. The objective of this chatbot has to be clear: Help the user carrying out processes, not repeating the same steps as in the open government portals but in a different platform. But in order to do that, the assistant must understand what the user wants.

Even if the testers understood perfectly that this was a demonstration and the chatbot would not understand everything. Nonetheless, they expressed several times that if this was a public tool, they would expect a fully intelligent assistant. The reasoning is logic: If the user has to talk naturally, the answers have to be natural and the conversation fluid. If this is not achieved, the user will stop seeing this tool as an assistant, and will start seeing it as a robot. And this tool has to give the feeling of being human. Another way to achieve this feeling is giving the chatbot a personality.

Another important conclusion is that the robot has to have a settled personality that has to be constant. This has to be done to give the user the sensation that it is talking with a person. During the interviews, the testers realised with joy that the agent used emojis to express feelings, as a human would do in a message conversation. This personality has to be established since the beginning of the development, to translate it into the messages and into the agent conversation flow.

A complete integration of the agent with each app should not be forgotten. If establishing a personality is essential, translate it to the agent is equally – if not more – important. This can be easily done with the tools offered by each messaging system. For example, Facebook offers cards, buttons and persistent menus to guide the user, but also to extend this personality to the user experience. Nonetheless, these features should be add-ons and not the main features, as these usually work differently on each platform and the bot should be able to work on diverse platforms.

A bot should be designed taking into account future integrations in other applications. During this development, the only app supported was Facebook, but it could have been easily ported to other messaging services. The proposed solution it is about an assistant, and this one should be available everywhere. This argument is supported by several companies like Google [25] or Amazon [26], who are extending their virtual assistant everywhere (from mobile phones to cars). Core functions should be able to be executed in any messaging app.

A big mistake made during the chatbot design and development was the inclusion of a dictionary to validate the name and surname of the testers. The initial idea was that if the auditor wrote “Jaiver” instead of “Javier”, the system would recognise his error and ask him again for the name. The problem was the recollection of all the names. Even if thousands of names and surnames were introduced into the dictionary, it was not enough and some of them were not recognised. Also, the assistant could not learn new names by itself. This made that a lot of testers had to modify their names, introducing fictional ones, in order to continue testing the chatbot. From this, a conclusion can be extracted: All the information written by users will be unexpected, and the system must be prepared for it. For this, the developer must follow two rules: i) do not introduce a dictionary unless strictly necessary and ii) test the application with different users than yourself.

It is strictly necessary to test the chatbot before doing the public release. This testing has to be done with different people than the development/design group, as the idea is to retrieve phrases and situations that this group did not expect. As said previously, the robot is expected to be intelligent, and the maximum of all of the possible outcomes should be considered. Normally, these new cases found during the testing phase are not very important and can be easily fixed. If a phrase is not recognised, the user will be obligated to reformulate the petition, making his experience less satisfactory. During the testing of this application, a lot of phrases, words and situations were found that were not considered initially. Also, if the testing is done at a large scale, it could detect how the system reacts to several user petitions at the same time.

If the chatbot has to communicate with a database, web service or make an action in some system, a server will be needed. In this case, it is important to assure that the server always works and it is prepared for a certain influx of people. In this case, the

selected NLP tool demands that all external operations have to be done in less than 5 seconds.

Concluding, it is not extremely complicated to build a chatbot, but several design obstacles can be found during the development of it. The complexity of this tool is not the building process, but the design one. There is not a standard solution to everyone, as each bot will have a different personality and will need to solve a different problem. There are several aspects to take into account in all chatbot developments (as stated previously), but each case should be analysed individually. Nonetheless, this research has proved that is possible to build a functional chatbot in three months.

8.2 Administrative Portals and Idea Execution

Looking at all the data recollected, a conclusion can be extracted: Open government portals are tools not known between the citizens. More than half of the surveyed did not know about these websites, even if most of them were interested in doing administrative processes via online services. Also, the need of digital certificates or electronic ID reading systems restrain a lot of citizens from accessing totally to these systems. These portals have to be simplified, reducing the constraints to access them. Also, open government websites have to be much more advertised, to reach a larger number of citizens.

And this proposed idea can popularize these websites. According to the surveyed, more than half of them have discovered these portals and are going to use them in the future. With this simple feature, citizens can discover the digital administration and start doing more complex tasks on the already available services. Nonetheless, surveyed said that they would continue using the chatbot only if more features are installed.

The proposed solution has been accepted by the surveyed people, but some of them have asked for more features. Chatbots have to simplify tasks and guide the users to finish complex ones, but it has to keep everything clean. The future features should be simple, or at least, not a lot of information should be asked to the user.

Also, it is important to note that most of the surveyed did not want to realise complex administrative processes on the chatbot. This can be for two factors: i) the platform is not convenient for this type of formalities and ii) they are not interested in these processes. Administrative procedures should be added, but blending them with simple or informative tasks, in order to make the citizens use the chatbot.

While using it, people will learn the administrative language and will become familiar with the open government idea. Also, if the formalities can be processed quickly via the chatbot, public administrations will have a better image and citizens will stop seeing them as a slow and boring system. As consequence, the citizen possibly will, step by step, start interacting with the Administration and will be more compromise with it.

Concluding, a chatbot should be used as a tool to introduce the user to the Open Government idea. The people will discover administration portal and other systems thanks to this tool. The chatbot will be an assistant that can recognise the problem and guide the user to solve it, or even resolve very simple tasks. But in order to achieve this task, open government portals should become simpler, formalities less complex and the administration should process these faster. Also, chatbots should always remain

uncomplicated and clear, because these are not an administrative portal replacement; chatbots are the assistants that will introduce, teach and guide citizens across this new digital administration.

CHAPTER IX: FUTURE LINES

As said several times in this document, this development was a demo and a testing concept, and it can be perfectly continued and updated. This chatbot is capable of doing more, but it has to be polished and fixed in order to do so.

One of the first actions that should be done in order to continue developing this app is to refine its recognition. During the testing phase, several phrases were not recognised by the agent. As said in the conclusions, people expect a fully intelligent agent and they do not want to repeat or reformulate phrases. New phrases, keywords and situations should be taken into account, and to achieve this, a new and more widespread testing should be done.

This chatbot should be demonstrated to a more varied public. This would help the developer learn what the people are expecting from the bot and how auditors react to its instructions. It is very important to study user behaviour, as the objective is to reduce the time needed to do something. The users should not learn how to use the tool, but the tool should learn to understand the users. However, users will need to follow chatbot instructions in order to finish their desired tasks.

The chatbot should be able to communicate briefly but effectively with the user. The agent will give instructions and these should be followed. As said in the conclusions, users are not used to read long texts, so the chatbot should give concise and clear instructions.

These are the most important things to do in order to continue the development, but more things should be done in order to deliver a complete product. More functionalities should be added and more platforms supported. Also, the chatbot should be able to explain the user services or even guide him across different open government portals.

Other possible feature would be to register the user with his personal data on the chatbot. Like that, once a user wants to do a complaint or needs personal assistance, most of his information will be stored and the chatbot will not need to ask him again for all his personal details. Procedure times will be reduced and user satisfaction will rise.

In the conclusions it was said that the chatbot needed a personality. This personality could be enhanced introducing small-talk to the agent. The robot could be able to answer simple questions as "What's your name?" in order to increase user interaction. Also, the chatbot could retrieve user information continuing the conversation. If a user asks him "How old are you?" the robot could answer: "I am only one year old! And you?" and then store the answer. Like that, the chatbot could be able to store future information that could be needed in procedures, reducing the time needed to complete one.

Nonetheless, chatbot technology could be adapted to other areas, like medical assistance. A chatbot could help users detect their diseases and take an appointment for them. But this could be also applied to simpler areas, like restaurants or stores. The assistant could be able to help you find a t-shirt and then buy it online. Or you could ask the assistant for a sandwich and he would command it for you.

Summing up, several things can be done in order to continue this or other projects. Chatbot technology is a new trend that, to the best of this author's knowledge, is being highly pushed by IT companies. But in order to integrate totally this technology in the day-to-day of the mainstream public, the public has to be studied and their behaviour studied. As Steve Jobs said two decades ago: "You've got to start with the customer experience and work backwards to the technology" [27].

CHAPTER X: BIBLIOGRAPHY

1. *The Cultural Evolution of Civilizations*. **Flannery, Kent V.** 399-426, Ann Arbor, Michigan : Annual Review of Ecology and Systematics, 1972, Vol. 3.
2. **National Geographic**. Civilization - National Geographic Society. *National Geographic Society*. [Online] National Geographic, 26 May 2018. [Cited: 26 May 2018.] <https://www.nationalgeographic.org/encyclopedia/civilization/>.
3. **Oxford Dictionaries**. state | Definition of State in English by Oxford Dictionaries. *English Oxford Dictionaries*. [Online] Oxford, 24 May 2018. [Cited: 24 May 2018.] <https://en.oxforddictionaries.com/definition/state>.
4. —. government | Definition of Government in English by Oxford Dictionaries. *English Oxford Dictionaries*. [Online] Oxford, 24 May 2018. [Cited: 24 May 2018.] <https://en.oxforddictionaries.com/definition/government>.
5. **Open Government Partnership**. About OGP | Open Government Partnership. *Open Government Partnership*. [Online] Open Government Partnership, 26 May 2018. [Cited: 26 May 2018.] <https://www.opengovpartnership.org/about/about-ogp>.
6. **Techopedia**. Techopedia. *What Is an Intelligent Virtual Assistant*. [Online] Techopedia, 26 April 2018. [Cited: 26 April 2018.] <https://www.techopedia.com/definition/31383/intelligent-virtual-assistant>.
7. **Chan, Carylyne**. The Chatbot Landscape, 2017 Edition. *KeyReply Blog*. [Online] 26 June 2017. [Cited: 2018 April 26.] <https://blog.keyreply.com/the-chatbot-landscape-2017-edition-ff2e3d2a0bdb>.
8. **TechTerms**. Webhook Definition. *TechTerms*. [Online] TechTerms, 12 June 2017. [Cited: 6 April 2018.] <https://techterms.com/definition/webhook>.
9. *A (Very) Brief History of Artificial Intelligence*. **BUCHANAN, Bruce G.** 4, s.l. : AI Magazine, 22 December 2005, AI Magazine, Vol. 26, pp. 53-60.
10. *On Computable Numbers, with an Application to the Entscheidungsproblem*. **TURING, A. M.** 1, s.l. : London Mathematical Society, 1937, Proceedings of the London Mathematical Society, Vols. s2-43, pp. 544-546.
11. **CAMBRIDGE**. Turing test Meaning in the Cambridge English Dictionary. *Cambridge Dictionary*. [Online] Cambridge Dictionary, 3 May 2018. [Cited: 3 May 2018.] <https://dictionary.cambridge.org/dictionary/english/turing-test>.
12. **ENGE, ERIC**. 31 Must Know Mobile Voice Usage Trends for 2018. *Stonetemple*. [Online] Stonetemple, 20 January 2018. [Cited: 15 Mars 2018.] <https://www.stonetemple.com/voice-usage-trends/>.
13. **Wavestone**. The Rise of Intelligent Voice Assistants. *Wavestone*. [Online] September 2017. [Cited: 15 Mars 2018.] <https://www.wavestone.com/app/uploads/2017/09/Assistants-vocaux-ang-02-.pdf>.
14. **EASTWOOD, GARY**. How third-party developers are transforming the Alexa experience. *Network World*. [Online] IDG, 22 February 2017. [Cited: 15 Mars 2018.]

<https://www.networkworld.com/article/3172190/internet-of-things/how-third-party-developers-are-transforming-the-alexa-experience.html>.

15. **Google**. Learn more about the world around you with Google Lens and the Assistant. *Google Blog*. [Online] Google, 21 November 2017. [Cited: 15 Mars 2018.] <https://www.blog.google/products/assistant/learn-more-about-world-around-you-google-lens-and-assistant/>.

16. **LANGDORF, JESSICA**. 3 challenges facing virtual assistants. *VentureBeat*. [Online] VentureBeaat, 26 August 2016. [Cited: 23 Mars 2018.] <https://venturebeat.com/2016/08/26/3-challenges-facing-virtual-assistants/>.

17. **Elgan, Mike**. Why smartwatches failed. *Computerworld*. [Online] IDG, 26 August 2017. [Cited: 2 May 2018.] <https://www.computerworld.com/article/3219854/wearables/why-smartwatches-failed.html>.

18. **Huhtanen, Hannu**. Chatbots made easy with Dialogflow. *Reflections of a Goldfish*. [Online] 15 October 2017. [Cited: 22 May 2018.] <https://blog.huhtanen.eu/2017/10/15/chatbots-made-easy-dialog-flow.html>.

19. **Moon, Mariella**. Facebook Messenger launches its first airline bot. *Engadget*. [Online] 30 Mars 2016. [Cited: 28 May 2018.] <https://www.engadget.com/2016/03/30/facebook-messenger-first-airline-bot?guccounter=1>.

20. **JAMES**. Azure Functions vs AWS Lambda – Scaling Face Off. *Azure from the trenches*. [Online] 6 January 2018. [Cited: 23 Mars 2018.] <https://www.azurefromthetrenches.com/azure-functions-vs-aws-lambda-scaling-face-off/>.

21. **AWS**. AWS Lambda - Serverless Compute. *AWS*. [Online] Amazon, 3 May 2018. [Cited: 3 May 2018.] https://aws.amazon.com/lambda/?nc1=h_ls.

22. **Accenture**. Cracking the gender code: Get 3x more women in computing. *Accenture*. [Online] 20 10 2016. [Cited: 6 May 2018.] https://www.accenture.com/t20170417T030444Z__w_/us-en/_acnmedia/Accenture/next-gen-3/girls-who-code/Accenture-Cracking-The-Gender-Code-Report.pdf?la=en.

23. *Uses and Attitudes of Young People toward Technology and Mobile Telephony*. **Valor, Josep and Sieber, Sandra**. 66, s.l. : BLED 2003 Proceedings, 2003.

24. **Schwartz, Joseph and Shahar, Mandel**. SimilarWeb - The Market Intelligence Blog. *The Most Popular Messaging App in Every Country*. [Online] SimilarWeb, 24 May 2016. [Cited: 15 May 2018.] <https://www.similarweb.com/blog/worldwide-messaging-apps>.

25. **Lee, Nicole**. Engadget. *Google Assistant wants to be everywhere you are in 2018*. [Online] Engadget, 8 January 2018. [Cited: 17 May 2018.] <https://www.engadget.com/2018/01/08/google-assistant-everywhere-2018/>.

26. **Mannion, Patrick**. ElectronicDesign. *CES Teardown: Echo Connect Shows Why Alexa Will Be Everywhere in 2018*. [Online] Electronic Design, 3 January 2018. [Cited:

17 May 2018.] <http://www.electronicdesign.com/communications/ces-teardown-echo-connect-shows-why-alexa-will-be-everywhere-2018>.

27. **Jobs, Steve.** Steve Jobs Q&A. *WWDC*. s.l. : WWDC VHS, 13 May 1997. pp. 52:15-52:22. Can be found in: https://youtu.be/Jh_9Wwx43r4?t=52m15s.

Online reference

- Dialogflow: <https://dialogflow.com/docs/getting-started/basics>
- Facebook Messenger: <https://developers.facebook.com/docs/messenger-platform>
- AWS Lambda: <https://aws.amazon.com/documentation/lambda/>
- Google Docs Forms: <https://support.google.com/docs#topic=1360904>
- Good practices for chatbots – Mining Business Data: <https://miningbusinessdata.com>