

**FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO**

# **Chatbot for Domotics**

**Ricardo Jorge da Rocha Loureiro**



Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Ademar Manuel Teixeira de Aguiar

July 21, 2017



# **Chatbot for Domotics**

**Ricardo Jorge da Rocha Loureiro**

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Doctor Pedro Souto

External Examiner: Doctor Luís Teófilo

Supervisor: Doctor Ademar Aguiar

July 21, 2017



# Abstract

Home automation is an area that is seeing a lot of development and resources dedicated to in the past few years. Many devices nowadays are constantly connected to the Internet (IoT) and this is a huge opportunity to create a software application that can manage them all.

There were a lot of experiments, many of them with recognizable success, in creating the right software to manage these devices but none that succeeds completely. This is due to the overwhelming complexity and the difficulty of use. The purpose of having an IoT environment at home is to aim for comfort and ease of management, not to being constantly challenged or having the necessity of monitoring them all the time.

The main focus is to create a chat interface that allows the users to interact with their systems at home in their most used applications (reaching the users where they already are, like Facebook), creating something that not only allows you to control and manage your home automation systems but also something that gives you the right information at the right time by pattern detection or some behavior deviations. To test the developed product it will be used a domotics system already created by WIT Software and the MVP will be presented to the market afterwards.

The number of connected devices is expected to grow from 9 billion in 2011 to 24 billion in 2020. The opportunities to take on IoT are proliferating and this is the perfect time for the perfect technology, the market is aching for it.



# Resumo

Domótica é uma área que está a ver bastantes recursos e estudos serem dedicados nos últimos anos. Muitos dos dispositivos de hoje em dia estão ligados à Internet (IoT) e é a oportunidade perfeita para criar uma aplicação que os permita gerir.

Houveram várias tentativas nos últimos anos de criar a tecnologia perfeita, algumas com sucesso reconhecido, mas nenhuma conseguiu ser completamente bem sucedida neste paradigma. Isto deve-se ao elevado número de dispositivos e complexidade destes. O propósito de ter um ambiente IoT é ter algo de fácil gestão que provoca uma sensação de conforto e não a necessidade de estar constantemente a controlar todos os dispositivos para ver se estão a ter o comportamento esperado, esse é o real desafio.

O objetivo principal é desenvolver uma interface de conversação que permite aos utilizadores interagir com a aplicação usando linguagem através das aplicações que já existem nos seus telemóveis (chegar aos utilizadores onde eles já estão, exemplo Facebook), criando algo que não só permite a gestão de todos os dispositivos ligados à Internet bem como a interação destes com o objetivo comum de proporcionar a informação correta no momento exato ao utilizador. Este produto será testado num sistema de domótica desenvolvido pela WIT Software e numa fase final apresentado ao mercado.

Espera-se que o número de dispositivos aumente de 9 mil milhões em 2011 para 24 mil milhões em 2020. O potencial de cativar parte de um mercado destas dimensões é bastante interessante. O mercado está a procurar a tecnologia certa e este é o momento perfeito para a desenvolver.





# Acknowledgements

I would like to thank everyone that were present during not only the completion of this dissertation but also my master thesis.

First I would like to thank my family and friends. Their support has been extraordinary through the years and I'm sure that I can count on them for a lot more. I would also like to thank my girlfriend for being there for me since the beginning.

I would also like to thank all the professors at Faculdade de Engenharia da Universidade do Porto which gave me guidance and knowledge along these five years, with a special thanks to my supervisor Ademar Aguiar.

Also a word of appreciation to WIT Software who proposed this dissertation and provided me with all the tools necessary to accomplish it, more specifically a special thanks to Paulo Sousa and Pedro Andrade which provided continuous support since the beginning.

Ricardo Jorge da Rocha Loureiro



*“The first rule of any technology used in a business is that automation applied to an efficient operation will magnify the efficiency.  
The second is that automation applied to an inefficient operation will magnify the inefficiency.”*

Bill Gates



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Context . . . . .	1
1.2	Motivation and Objectives . . . . .	1
1.3	Document Structure . . . . .	2
<b>2</b>	<b>Literature Review</b>	<b>3</b>
2.1	Natural Language Processing . . . . .	3
2.2	Chatbots . . . . .	5
2.2.1	Introduction . . . . .	5
2.2.2	Chatbots as Information tools . . . . .	5
2.2.3	Chatbots Assistants . . . . .	6
2.2.4	Development Tools . . . . .	7
2.2.5	WIT Bot Platform . . . . .	9
2.2.6	Conclusions . . . . .	11
2.3	Internet of Things . . . . .	11
2.3.1	Introduction . . . . .	11
2.3.2	Technologies . . . . .	12
2.3.3	Domotics Platforms . . . . .	13
2.3.4	Current Issues . . . . .	13
2.3.5	Conclusion . . . . .	15
2.4	Pattern Recognition . . . . .	15
2.4.1	Introduction . . . . .	15
2.4.2	Scikit-Learn . . . . .	16
2.4.3	Conclusions . . . . .	17
<b>3</b>	<b>Implementation</b>	<b>19</b>
3.1	Chatbot . . . . .	19
3.1.1	WIT Bot Platform Flow Mechanism . . . . .	19
3.1.2	Amazon Lex and Alexa Integrations . . . . .	21
3.1.3	Chatbot Use Cases . . . . .	23
3.1.4	Conclusions and Important Remarks . . . . .	26
3.2	Abstraction Platform . . . . .	27
3.2.1	Server Overview . . . . .	28
3.2.2	Machine Learning . . . . .	32
3.3	Implementation Summary . . . . .	34

## CONTENTS

<b>4</b>	<b>Experiments and Results</b>	<b>37</b>
4.1	Questionnaire . . . . .	37
4.2	Usability Test . . . . .	39
4.2.1	Results and Conclusions . . . . .	39
<b>5</b>	<b>Conclusions and Future Work</b>	<b>43</b>
5.1	Conclusions . . . . .	43
5.2	Future Work . . . . .	43
	<b>References</b>	<b>45</b>
<b>A</b>	<b>Questionnaire</b>	<b>49</b>
<b>B</b>	<b>Usability Test</b>	<b>53</b>
B.1	Amazon Echo . . . . .	53
B.2	Facebook Messenger . . . . .	53
B.3	Amazon Echo and Facebook Messenger . . . . .	54

# List of Figures

2.1	Google Syntax Net Simple Example . . . . .	5
2.2	Google Syntax Net Complex Example . . . . .	6
2.3	YPA architecture . . . . .	7
2.4	Happy Architecture . . . . .	8
2.5	Sanelma . . . . .	9
2.6	Amazon Lex . . . . .	10
2.7	AWS . . . . .	11
2.8	WIT AI . . . . .	12
2.9	WIT AI . . . . .	13
2.10	IoT Technologies . . . . .	14
2.11	Internet of Things . . . . .	15
2.12	Pattern Recognition . . . . .	16
3.1	Bizagi Modeler - Engaging Pattern Example . . . . .	20
3.2	Bizagi Modeler - Different Task Elements . . . . .	21
3.3	Facebook Example Messages . . . . .	22
3.4	Bizagi Modeler - User Task Properties . . . . .	23
3.5	Bizagi Modeler - Example flow . . . . .	24
3.6	Implemented Flow - Turn Device State . . . . .	25
3.7	Implemented Use Case - Quick Replies . . . . .	26
3.8	Implemented Flow - Schedule a job . . . . .	27
3.9	Implemented Flow - Schedule a schema . . . . .	28
3.10	Implemented Flow - List and Notification . . . . .	29
3.11	Chatbot Use Cases Examples . . . . .	30
3.12	System Overview . . . . .	31
3.13	Scikit-Learn Polynomial Regression . . . . .	33
4.1	Questionnaire - Results Question 1 . . . . .	38
4.2	Questionnaire - Results Question 2 . . . . .	39
4.3	Questionnaire - Results Question 3 . . . . .	40
4.4	Questionnaire - Results Question 4 . . . . .	41
4.5	Questionnaire - Results Question 6 . . . . .	41
4.6	Questionnaire - Results Question 7 . . . . .	41
4.7	Questionnaire - Results Question 8 . . . . .	42
A.1	Questionnaire First Page . . . . .	50
A.2	Questionnaire Second Page . . . . .	51

## LIST OF FIGURES



# List of Tables

2.1	Home Device Systems . . . . .	13
2.2	Open Source Home Assistant Projects . . . . .	14
2.3	Classification and Regression Algorithms in Scikit-Learn . . . . .	17
2.4	Clustering algorithms . . . . .	17
3.1	Engaging Patterns on Turning Device State Use Case . . . . .	23
3.2	Engaging Patterns on Scheduling a Job Use Cases . . . . .	24
3.3	RESTful API Routes . . . . .	31
4.1	Usability Test, Facebook Messenger Results . . . . .	42
4.2	Usability Test, Amazon Echo Results . . . . .	42
4.3	Usability Test, System Results . . . . .	42

## LIST OF TABLES

# Abbreviations

IoT	Internet of Things
NLP	Natural Language Processing
NLU	Natural Language Understanding
YPA	Yellow Pages Assistant
FAQ	Frequently Asked Questions
XML	eXtensible Markup Language
ASR	Automatic Speech Recognition
AWS	Amazon Web Services
RFID	Radio Frequency IDentificaton
WSN	Wireless Sensor Networks
MVP	Minimum Valuable Project
SaaS	Software as a Service
WBP	WIT Bot Platform
BPMN	Business Process Management Notation
AIML	Artificial Intelligence Markup Language
JSON	JavaScript Object Notation



# Chapter 1

## Introduction

“We think you should message a business just the way you would message a friend” [Zuc16]. Artificial Intelligence Systems are the new era of technology. With the huge increase of devices connected to the Internet it is necessary to analyze their data and make it valuable to the users.

### 1.1 Context

Since the first introduction of smart home devices, the way people interact with their everyday objects has been changing. This technology has been growing at a respectful rate and it is expected that “24 billion things” will be connected to the Internet by 2020. [MMGA<sup>+</sup>15].

There is also another technology that has been prevalent everywhere, Bots. The market has seen huge funds being dedicated to research on this topic, from Facebook to Google, it is an evolving technology. Currently it is possible to create a chatbot that can be deployed to Facebook Messenger which holds up to 900 Million users [MBRR07], this is a huge opportunity to reach this market.

In the past years, there has been huge improvements in these areas and a considerable number of applications with potential exist and have a significant market share. However none of those is quite similar and has the end objective of this dissertation, having something that connects to our IoT devices and through pattern detection and machine learning suggests new stories to the user.

### 1.2 Motivation and Objectives

This dissertation comes together in the context of combining a chatbot associated with artificial intelligence integrating an home automation system. Although there are several questions that come up to mind that might be worrying and needed to be tackled accordingly. Bots are a new technology and are now only coming up to the market, it has still a lot to grow to reach a mature state to impact the market in a meaningful matter. There is also the fact that there are numerous IoT applications currently on the market, our vision must add some value to the end-user or it will

be left in the dark.

One of the main goals is to create a proof of concept that will be trying to take advantage of the best of both worlds, something that will manage all your IoT devices in a simple chatting interface being complemented with artificial intelligence with the objective of detecting patterns and suggest the best information to the users as possible.

To accomplish this goals the project has been divided in a few different phases. The first one is a research of the similar approaches from the market in these areas, both IoT and Chatbots. In this phase was also studied which technologies should be adopted during the development process, taking to consideration both the clients, ease of development and scalability. The second phase is the development phase; it was used Scrum with 2 weeks sprints so we could have a constant knowledge where the product stands and test its growth. There was a testing phase afterwards that aimed to wrap up the project giving the opportunity to test and further evaluate the product quality and, the most important, its traction to the market.

### **1.3 Document Structure**

Beside this Introduction chapter, this dissertation also contains 5 other chapters. In the chapter 2, is described the current state of art and presented some technologies already existing in the area. In the third chapter 3 is presented the implementation of a chatbot and all the technologies used to achieve the final product. All the results from the questionnaires and tests done to validate this project can be seen in the chapter 4. In the last chapter, 5, are some conclusions about this dissertation and the work in mind for the future.

## Chapter 2

# Literature Review

Chatbots as a service is growing rapidly, among developers and clients. The focus of this dissertation is the creation of a chatbot that will serve the users as a home assistant associated with artificial intelligence detecting behavioral patterns and giving meaningful opinions.

The creation of this chatbot can be done using different applications already well established in the market, and without doubt this is the way to go. In this section, it will be presented a different set of options and study their key points and where they fall short. It will also be presented some of the best approaches to the pattern recognition problem.

### 2.1 Natural Language Processing

#### Background

Natural Language Processing is a mixed discipline and is still influenced by different areas. The key areas that affect directly the development of NLP are: Linguistics, Computer Science, Psychology and Artificial Intelligence.

#### Definition

Natural Language Processing (NLP) is the computerized approach to analyze text. The input can be oral or written, the only real requirement is that they be in a language used by humans to communicate to one another. It is important to note that this text should not be constructed with the purpose of analysis but rather from an actual usage. This is usually mentioned as “naturally occurring texts” [Lid01].

The goal of NLP is to reach a state of human-like language processing, Natural Language Understanding. It is rather complex to be able to know if we’re in fact upon a system that understands language, all we can do is test if it performs its task successfully. A full understanding system would be able to:

- Paraphrase a source
- Translate the input to any other known language

- Answer and draw inferences from a source

Despite the progresses made in this area the state of full NLU is yet to be accomplished [Spy96].

### Levels of Natural Language Processing

The most practical way of representing what happens within a NLP system is by representing the language in different levels. The system is as dynamic as the number of levels interacting with each other. The principal difficulty in NLP is the pervasive ambiguity found at these levels. This ambiguity can be found at:

- Lexical: the same word can be either a verb, a noun and so on;
- Structural or syntactic: it can be hard to associate per example an object to a person or an activity (e.g. "I was seen with a telescope");
- Semantic: one word may have different meanings;
- Pragmatic or Referential: Depends a lot of the contexts where they are placed, it is sensible to interpretation.

These forms of ambiguity interact producing an extremely complex interpretation process.

Current NLP systems tend to implement the lower levels for several reasons. The application may not require all the complexity of higher levels, also lower levels are more easily regulated and have been widely researched.

### Google Syntax Net Approach

Google Syntax Net is an "open source neural network framework implemented in TensorFlow that provides a foundation for Natural Language Understanding" [Pet16a] systems. This framework provides both models that will be trained with our data and a parser, Parsey McParseFace. This parser allows efficient tools to analyze english text. It is referred as the most accurate model in the world to parse and analyze text, this is due to the machine learning algorithms that its built upon.

#### How does it work

It all starts with the text to analyze. Given this input the syntactic parser will start tagging each word with their respective syntactic function, creating a dependency parsing tree with the relationship between them.

This is a rather 2.1 simple example to display the actual processing and creation of the parsing tree by the parser. Here it is shown that the tree's root is the verb "saw" and Alice its subject while Bob is the direct object of the relation. This parser does not stop here, several other examples could be included as seen in 2.2 . This structure of storing data allows to recover each of the relationships within a sentence in a practical way. Questions as "Whom did Alice see?" and "Who saw Bob?" are easily answered by the system.



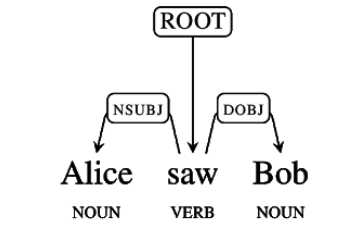


Figure 2.1: Parsing Syntax Tree [Pet16a]

### Parsey McParseface

This syntactic analyzer [Pet16b] has been referred before as the most proficient solving this problems, so how accurate it actually is? The first test environment was done retrieving 20 sentences from the Penn Treebank [MKM<sup>+</sup>94], a well-formed text basis. The parser recovered individual dependencies between words with an accuracy of over 94%. This reaches almost the human level of understanding, which is around the 95-96% [Pet16a], however this is only when the text is well-formed. The second test was in an web environment, the text is considerably harder to understand computer-wise decreasing the accuracy to just over 90%. This is far from being the perfect NLU system, but the high accuracy results prove that it is in fact a tool with extreme potential to developer use in many applications.

## 2.2 Chatbots

### 2.2.1 Introduction

A chatbot system is a software program that interacts with users using natural language. People naturally communicate with each other using language, the next step is talking to computers in the very same way. This idea was the driver behind the development of chatbots. Humans tend to “express their interests, wishes, or queries directly and naturally, by speaking, typing, and pointing” [ZBC<sup>+</sup>00].

The purpose of a chatbot system is to interact and stimulate a conversation with a person without compromises. This systems are composed by NLP capabilities and language models to simulate this conversations. In the first years of development there were a few implementations using simple keyword matching. With the constant improvement of of data-mining and machine learning techniques chatbots have become more practical, with manny comercial potential.

### 2.2.2 Chatbots as Information tools

Chatbots can be useful in a wide set of fields as an information retrieval tool. The authors of [SA07] developed a chatbot, Sophia. This messaging as a service system was created with the intent of allowing users to ask anything they might have doubts about while also interacting with other mathematical agents such as Pari and Mathematica to help in solving Algebra Problems.

Sophia is not only able to solve this tasks but also provides some knowledge about movies and common jokes to make it more user-friendly and trigger the sense of excitement in its users. It was used at the Harvard Mathematics Department with results that prove that teachers can use it to look for problems and students usually take it as a solving tool. But this goes way beyond education, in the very same paper was also created an FAQ chatbot system, where the bot took an attempt to answer the most asked questions. The results prompted to the user seem to be similar to the ones displayed by Google. The feedback given by the users were once again positive, what could be more natural to interact with a system that asking it something in natural language? Other really good example of chatbots as an information retrieval tool is the Yellow Pages Assitant [DRKN<sup>+</sup>98]. The YPA is composed by a NLP system and a wide database retrieved from British's telecom yellow pages. We can see its architecture in the figure. The interaction starts with an input from the user. If the chatbot succeeds to retrieve the information requested by the user during the chatting experience it engages a conversation flow with meaningful information aiming to suite the user needs. Still mentioning the same paper as before a study was done to prove this system credibility. It was prepared 75 questions, such as "I need a plumber with an emergency service?" and "Which restaurants are there in Colchester high school?", and saved the answers before-hand to compare with the results form the chatbot system, it succeeded 74% of the times.

“We think you should message a business just the way you would message a friend” [Zuc16].

Happy is a system developed to help “users access e-commerce sites to find relevant information about products and services” [CLZ<sup>+</sup>00]. The chatbot process the user information parsing it to a well structured XML logic form. Then it tests the structured data against it knowledge domain and if necessary it interacts with the user again asking for further information or prompting the

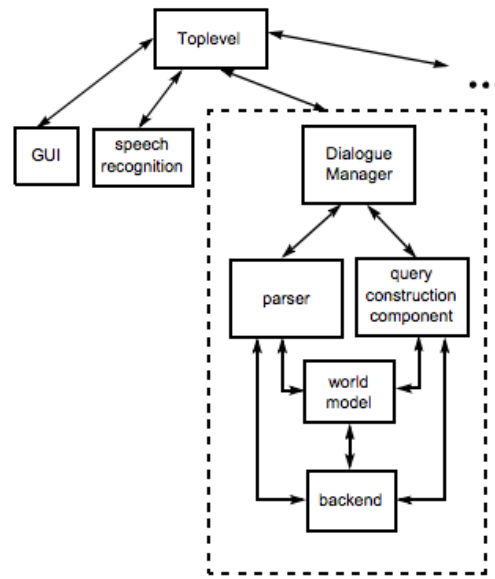


Figure 2.3: Yellow Pages Assistant architecture [DRKN<sup>+</sup>98]

results immediately. In all chatbots is important to pass a certain sense of feedback to the user in how the results appear, happy also takes care of this paraphrasing user requests in the results prompted. An example of interaction can be seen in the figure.

Another successful example is Sanelma. It is a fictional person that has knowledge of certain pieces of art and can be found on some Finnish museums providing additional information about these objects. “Sanelma is a 26 year-old woman from Helsinki of the 30’s” [MUM]. Through this chatting interface the users can immerse in a conversation from the epoch while learning, which was their main goal.

## 2.2.4 Development Tools

### 2.2.4.1 Amazon Lex

Amazon Lex is a service for creating chat interfaces with voice and text [Ama16b]. Lex allows Automatic Speech Recognition (ASR), converting speech to text and therefore taking advantages of the NLP to recognize the intents of the input. The goal is to be transparent to the user the fact that is interacting with a Bot (Turing Test) creating a high level of engagement and interactions that adds value to the user.

#### How does it work

Lex has a different set of tools as mentioned before such as ASR and NLP allowing developers to create more rapidly a chatbot. Its integration is done with the AWS Lambda or Amazon’s sdk which facilitates the process of back-end data treatment. After this creation, it can be easily deployed to any chat platform on mobile devices or even IoT devices. It also provides a data of

```
U: I am looking for a notebook for my consulting business
S: Please describe your financial constraints.
In this stage a list of most popular ones is displayed
U: not important thing performance is essential
S: are you looking for something that is top of the line?
U: Yes, absolutely
S: I have found something for you.
The target notebook is displayed for the user. And beneath it a summary
of the users request displayed to explain why this product is displayed.
```

Figure 2.4: Happy Assistance System Architecture [CLZ<sup>+</sup>00]

metrics that are fundamental to the optimization of the bot overtime and that will be the focus of this dissertation. It is always important to look at the bigger picture while developing a project and amazon got it covered by providing a solution that is scalable, secure, easy to deploy and monitor bots.

### Conversational interfaces

There are two types of engagements by the user: confirmation, which allows that an action is done before any type of interaction and the error management that usually tries to take a different approach to understand what is said by the user with greater clarity. The confirmation solicitations are required to detect the intent of the user before executing any type of logic in the back-end (e.g. “Turn off all the lights”). The management solicitations are to secure that the intent detected is in fact what the user meant. It is worse to understand wrongly what was the users objective than to ask again. This usually results from a deficient audio source provided by the user or if it could not be mapped to a specific known intent, Lex allows to define the number of repetitions that it will ask to the user before it gives up.

### Integrations

The integration with the AWS Lambda allows the developer to easily recover or manage the data from an external service. Amazon Lex also has several connectors with SaaS applications such as Salesforce, Microsoft Dynamics, Marketo, Zendesk, QuickBooks and Hubspot. It also has integration with chat applications such as facebook messenger and in the near future, they intend to launch integration with twillio and slack.

Amazon Lex can be purely used for its NLP and ASR algorithms, working as a standalone system that feeds the data and provides results throught API usage.

#### 2.2.4.2 Amazon Alexa

Alexa [Ama16a] is a service that allows users to interact with their devices in a more intuitive way, using voice. Alexa has a set of skills, such as playing music and answering general questions, that can be easily integrated with any application providing added value to it. This device is built over the Amazon Lex previously described.

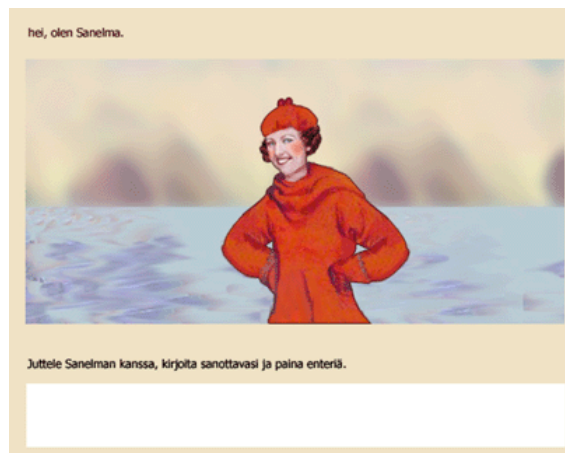


Figure 2.5: Chatting Interface [MUM]

Alexa allows the developers to create any set of skills that take advantage of their capabilities, hands-free voice experiences. Apart from this set of skills, Alexa also has a voice service that enables the possibility of having a conversation with the application.

### 2.2.4.3 WIT.AI

Wit AI is a service that allows the creation of the logic behind a chatbot using NLP to process the data [Fac16]. It is as simple as sending the input requested by the user and they tell you what is the next step. The developer can create a story for each type of conversation that the user might have and that the bot should know of. Whenever an input text is sent to the Wit.AI they will respond with a percentage of trust what is the intent of the user and the information requested. Again, such as Lex, WIT.AI can be used as an NLP motor and consumed by API requests. The conversations interfaces are similar in every aspect to the Amazon Lex previously described.

### Integrations

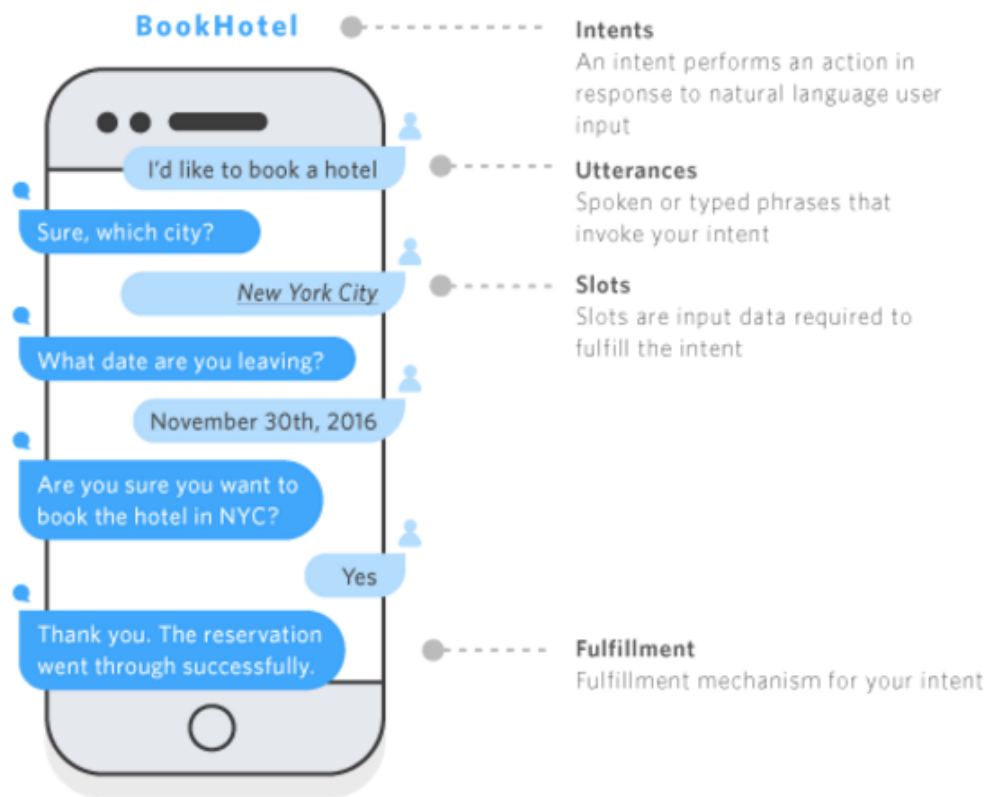
There are several well-documented open source libraries and SDKs for iOS, Ruby, Node.js, Python and so on that ease the process.

### 2.2.5 WIT Bot Platform

Along this section has been described the various advantages that chatbots bring to the real world business. Aiming exactly at that WIT Software developed a Messaging as a Service platform that allows end-to-end communication using chatbots, named WIT Bot Platform [Sof17].

#### 2.2.5.1 The Gateway

The WBP Gateway is the module responsible for both enabling bots into messaging channels, as well as maintaining their users satisfied while informing accordingly the respective businesses. It

Figure 2.6: Conversational Interface [[Ama16b](#)]

offers the possibility for users associate their account to the different businesses and setting up their preferred channel. This type of flexibility is key while building a Messaging as a Service platform. Their clients can choose from Skype, Facebook Messenger, RCS, SMS and so on. It also offers the possibility of integrating with third-party bot API services, to seamlessly integrate any chatbot that is currently in production to this services.

In any business it is important to have an analytical department that evaluates the different metrics of a certain system. This gateway provides insightful information such as: engaged and lost users, sessions and conversations per session. It also provides real-time sentimental analysis of each interaction allowing at any time a human take control of the conversation flow allowing the bot to learn.

#### 2.2.5.2 The Engine

The WIT Bot Engine is a platform that enables developers to create a chatbot in a comprehensive and easy to manage manner.

This engine has a set of built-in templates that are useful in customer service, enterprise productivity and retail. It also allows a convenient and seamless way to create new flows based on

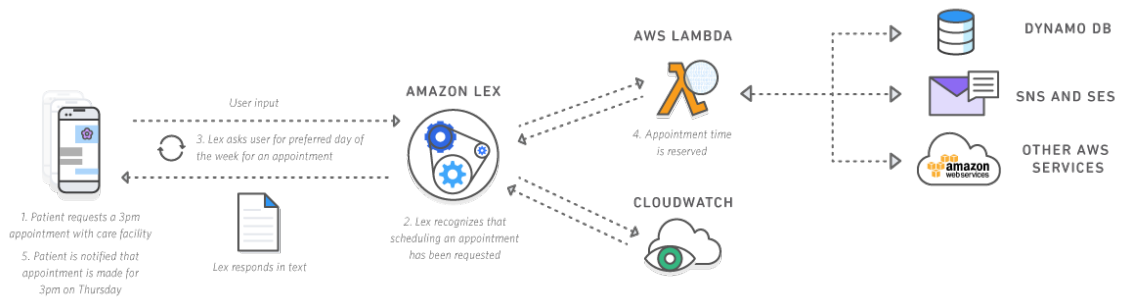


Figure 2.7: AWS platforms workflow [Ama16b]

Business Process Management Notation as seen in Figure 2.9. The code is afterwards generated with most of the basic logic.

It is also empowered with NLP integrations, such as WIT.ai and API.ai. This is extremely useful for understanding what the user is trying to do and in that way not being entirely dependent of AIML files. Adding to this feature set they close everything with Machine Learning capabilities, such as TensorFlow.

### 2.2.6 Conclusions

As shown in this section chatbots are really giving added value to users and creators assisting their everyday lives and replacing some of the most uncommon methodologies.

Chatbots however will never replace everything existing in the world, instead it should be seen as a mechanism of helping people, facilitating their work, and their interaction with computers using the most natural mean of communication, language.

## 2.3 Internet of Things

### 2.3.1 Introduction

The Internet of Things paradigm is gaining a lot of ground in the scenario of modern telecommunications. The basic idea behind this concept is that everything will be connected within a network from one way or another cooperating to reach a common goal. This network will be conceived by Radio Frequency IDentificators (RFIDs) tags, sensors, actuators, mobile phones and so on [AIM10].

Without a doubt one of the key points of IoT is the possibility of impacting numerous aspects of our everyday lives. This can be seen from two perspectives, private user or enterprise-wise. Taking in consideration the end-goal of this dissertation it is obvious that domotics will be one of the major aspects that will influence many private users but it doesn't end there. Fields such as assisted living, e-health, enhanced learning are only a few of which that can evolve immensely from it [GBMP13].

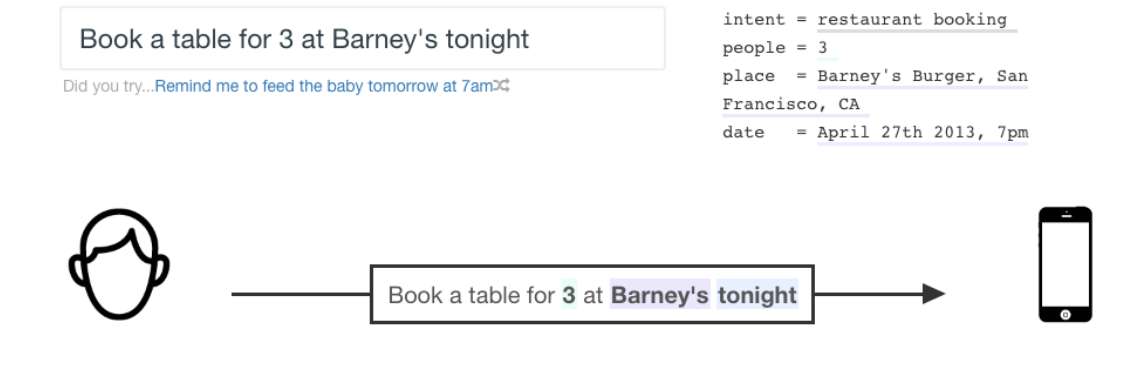


Figure 2.8: Conversational Interface Mechanism [Fac16]

The Internet revolution led the communication between people at an unprecedented scale and pace, everything points that the next revolution is creating a smart environment interconnecting all devices. The National Informatics Center predicts that “by 2025 Internet nodes may reside in everyday things – food packages, furniture, paper documents, and more” [AIM10].

### 2.3.2 Technologies

As described in the previous topic there are a few technologies that enable the communication between devices, it will be presented some of those technologies in a superficial manner.

#### 2.3.2.1 RFID systems

This systems are composed from one or more RFID tags. These are characterized by a unique identifier and are associated to a certain object. To retrieve data from this tags a signal needs to be generated which represents a query for all tags within a certain range. RFID systems can be used to monitor objects in real-time without the need to being directly in line of sight. This enables an incredible flexibility and it can be applied to any kind of scenario from logistics to e-health. The dimensions of this tags are usually very low, allowing once again flexibility, It can be as low as 0.4mm \* 0.4mm \* 0.15mm [Hor08] as developed by Hitachi.

#### 2.3.2.2 Wireless sensor networks

This networks consist of a large number of intelligent sensors that enable collection, processing analysis and dissemination of valuable information that are collected in a variety of environments. The data is shared among all the sensors and sent to distributed or centralized system for its treatment. This networks are composed by WSN hardware, communications stack, middleware and Secure Data Aggregation that are far too complex to be deepened in this research.



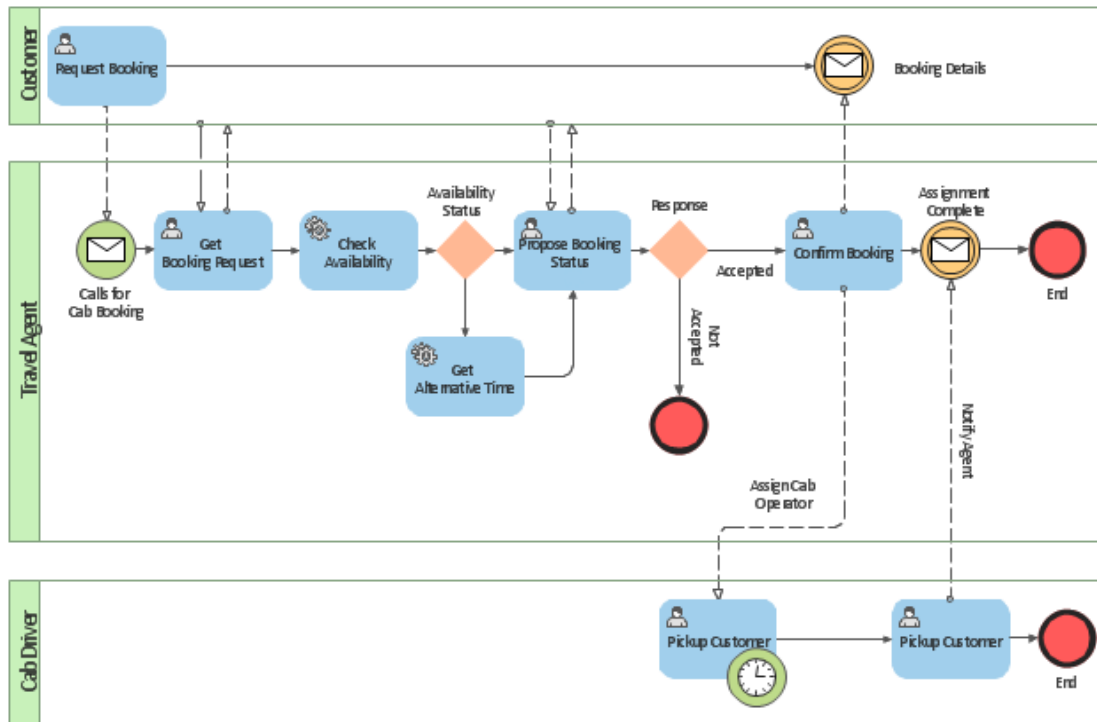


Figure 2.9: Example BPMN flow

### 2.3.2.3 Quick Overview of both technologies

### 2.3.3 Domotics Platforms

There are several home automation hubs that are now established in the market, in the table is presented some of the most common and pros and cons of this systems. [pcm15]

With the rise of IoT there is also a lot of open source projects that can help with the creation of a home automation system, on the table its displayed some of them and their key points [Rui15].

### 2.3.4 Current Issues

There are a few issues that need to be tackled before this market can scale at a even larger scale. Issues such as security, privacy and network are huge pain points of this paradigm. Lets take a better look at them.

#### Privacy

Table 2.1: Home Device Systems

	Pros	Cons	Bottom Line
Samsung SmartThings Hub	Wide wireless Protocols, Ease of use	Requires wired Ethernet, Lag	One of the most versatile
Philips Hue Bridge 2.0	Voice Control, Ease of use	Expensive	Expensive
Logitech Harmony Home Hub	Extensive device compatibility	Deep learning curve	Not so versatile

	Processing capabilities	Sensors	Communication	Range	Power	Lifetime	Size
RFID	No	No	Assymetric	10	Harvested	Indefinitive	Very Small
WSN	Yes	Yes	Peer-to-peer	100	Battery	<3 years	small

Figure 2.10: Overview of the two most used technologies [AIM10] chapter 3

Privacy has been a hot topic recently, with business like Facebook and Google being accused of huge privacy breaches [Hal15]. While people are feeling insecure about which information is actually being recorded and used the IoT environment will always face this as real problem. According to the paper [AIM10] Benetton is planning to tag an entire line of clothing and track its information. This means a number around 15 Million RFIDs, which is causing a lot of mistrust to what exactly this data will be used for.

This concerns are justified. It is totally impractical to know how many information we're actually seeding by using this devices, there are millions of messages being constantly sent and all this data can be used with the wrong intentions. Privacy should be protected by ensuring that each and every user select exactly which data wants to see shared and collected, by who, and when this happens.

With the goal to ensure data collection only when authorized a few solutions were created. The main concept is that there is a proxy that interacts with this systems and that is managed by the user, giving him the option to not only control each data is collected but also when this data is used. This suffers however of a scalability problem, again, by the huge number of data that passes this networks.

As the data storage costs reduce the number of collected data increases, becoming another issue. It is costless to store all the information that exists. All the solution in this area are now advised to implement forgettable capabilities. What this means is, the information that is collected and is not used has an expiration date, and after that date it should be deleted from the system to prevent data breaches from happening.

## Security

Table 2.2: Open Source Home Assistant Projects

	Key Points
OpenHAB	Easy of use, Several Plugins, Automatic data graph generation
PiDome	Runs on wide set of platforms
Domoticz	Set up rules, Quick and easy installation, Wide variety of device and services integrations
Home Assistant	Several protocols Support, Log event to files or database, Notifications, Timed commands
Fhem	GPS-Tracking, Web-based, Easy installation, Several protocols support
MajorDoMo	Very Lightweight, Time Based Operations, notifications, alarms and graphical reports
MyController	Automation by rules, Extensible by various plugins, Web-based
Node-RED	Visual tool for IoT wiring, Browser-based flow, Event-processing engine

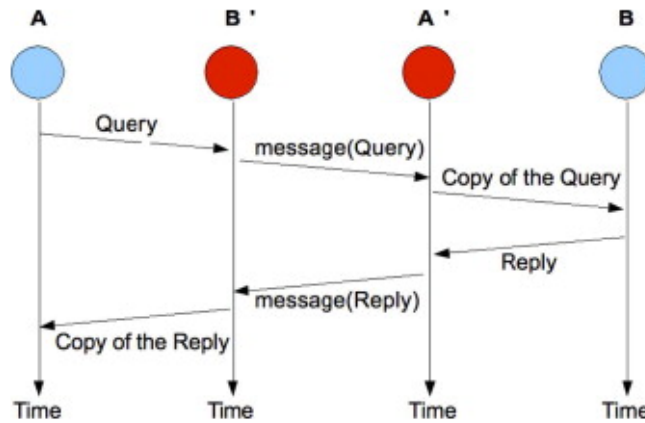


Figure 2.11: Man in The Middle Security Problem [PSW04]

The devices that support this paradigm, as mentioned in the section before, are characteristically extremely simple and with low resources, this causes the protocols of data treatment and transfer to be highly capable of being attacked, more specifically problems of authentication and data integrity. There were a lot of development theories and solutions proposed as seen in the paper [PSW04] however none of those protect against the man-in-the-middle attack.

These protocols aim to secure that the data generated by the nodes is the same data as reaches the user, but we can't ensure that. Most of this attacks can be easily prevented with cryptographic methodologies, however this devices can't support that type of algorithms as they lack the computation capabilities and are highly energy restrained.

More recently new solutions have been arising to prevent this type of data breaches “however (...) key management schemes are still at an early stage (especially in the case of RFID) and require large research efforts” [AIM10].

### 2.3.5 Conclusion

The possibility of enabling communications with and among smart objects lead to a vision of “anytime, anywhere, anymedia, anything” [WW11] communication, it really is the future, however there are some issues that need to be tackled before the implementation of this paradigms.

## 2.4 Pattern Recognition

### 2.4.1 Introduction

“The classification requirements are subjective since different types of classifications occur under different situations”[Fu68]. One of the main problems of pattern recognition is defining a model from the data available for each object. Usually objects that fall within similar patterns have among them some common properties, however not all of them are useful when considering the creation of a model.

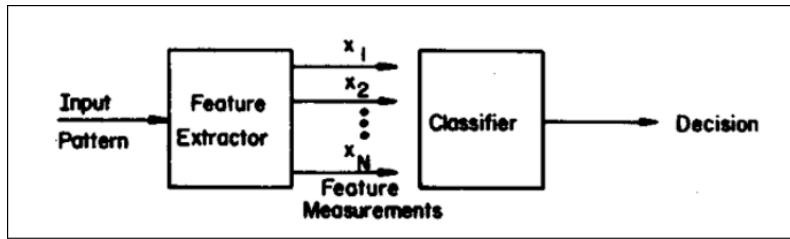


Figure 2.12: Pattern Recognition System [Fu68]

Another hard task that can surge as an obstacle is the problem of classification. This is making a decision to where a certain set of inputs are mapped to the respective class models. This type of classes are called features and the machine that classifies the objects is therefore a feature extractor. The device that performs the task of classification is called a classifier, the diagram presented in 2.12 represents a simplified pattern recognition system.

Machine Learning has been for a while a widely studied topic. With this came several platforms that help developers quickly setup good classifiers that answer their problems. Examples are TensorFlow and Scikit-Learn, both technologies developed under Google Laboratories and that can with achieve great results if used properly. In this next section will be presented with more detail Scikit-Learn, a framework for data analysis.

## 2.4.2 Scikit-Learn

The most common learning problems consist in a set of samples that are used to predict future and unknown values. However there is not one solution for all the problems, each and everyone should be evaluated and tackled accordingly. The most common set of problems can fall within two super categories, supervised and unsupervised learning.

### 2.4.2.1 Supervised Learning

Supervised Learning is the task of predicting additional attributes from labeled data taking in consideration previously trained models. These algorithms do that by feeding from data and afterwards inferring a function capable of doing so. The ultimate goal is that the algorithm becomes autonomous enough to generalize and predict unseen data.

Depending of the data type that we are trying to predict there are different approaches that need to be considered when choosing an algorithm. In general, supervised learning algorithms can be divided into two smaller groups, classification and regression problems.

Classification is often referred [Ras06] as a discrete form of supervised learning, generally there is a limited number of classes or categories and its job is to find the most appropriate correspondence to the samples provided.

Opposing to classification, regression solutions are usually used when the desired output consists in continuous variables.

Table 2.3: Classification and Regression Algorithms in Scikit-Learn

<b>Algorithm</b>	<b>Group</b>
Naive Bayes	Classification
Linear SVC	Classification
SGD Classifier	Classification
KNeighbors Classifier	Classification
SGD Regressor	Regression
Ridge Regression	Regression
Lasso	Regression

In the table can be seen the algorithms that are available in Scikit-Learn.

#### 2.4.2.2 Unsupervised Learning

Unlike supervised learning, this technique does not have any corresponding target values in their samples. It is also impossible to determine the model's accuracy, making this evaluation a lot more subjective.

Scikit-Learn looks at this type of learning by separating in two groups. Clustering, which aim to group similar samples and Density Estimation techniques that determine the distribution of input data.

A brief list of algorithms available in Scikit-Learn is presented in the table.

#### 2.4.3 Conclusions

Pattern recognition and machine learning are hard tasks to accomplish. Scikit-Learn provides a set of tools and documentation that helps integrating in every system. Both supervised and unsupervised learning has algorithms that are interesting and that could make sense to this dissertation goals. Either being using clustering algorithms to group devices and suggest them to the user for their similarity or knowing if a device is consuming as expected using regression algorithms with the power consumption as data.

Table 2.4: Clustering algorithms

<b>Algorithm</b>	<b>Group</b>
KMeans	Clustering
Mini Batch K-Means	Similar to KMeans focused on reducing computation time
Density estimation Histograms	Density Estimation
Kernel Density Estimation	Density Estimation

## Literature Review

## Chapter 3

# Implementation

Previously was described the set of tools and methodologies that were required to complete the product. In this chapter will be presented a more detailed and insightful information about the implementation.

First is a description of how was created a chatbot using WIT Bot Platform, providing code samples with the respective explanation that are essential to get a better understanding of the work done and its purpose, flow diagrams showing the chatbot use-cases and the set of integrations done to the existing project. Following-up will be the created abstraction platform. Here will be described its purpose, requirements and diagrams that help to get a better grasp of the platform as a whole. To finish the implementation section will be given information about the Machine Learning module. Further explanation of the algorithm used will be given to consolidate the research done in previous chapters and some examples to clearly see what it is doing.

### 3.1 Chatbot

As mentioned in the chapter [2.2.5.2](#), creating a chatbot within the WBP starts with the flow design that will represent the possibles engagements with the platform. Taking this into consideration lets break the design tool in a way that is easy to get a grip of what sustains everything. “Bizagi Modeler is a BPMN process modeling software that facilitates the creation and implementation of flowcharts and work flows ” [\[biz17\]](#). This application is used by WIT Software to create and model BPMN files.

#### 3.1.1 WIT Bot Platform Flow Mechanism

Different user intents require distinct flows. It would be possible to merge every flow into a super flow however this would not be scalable. The approach taken was to create separated flows that will act as one set of flows when the chatbot is created. This task is achieved by the creation of a BPMN flow parser that takes as input the BPM and XPDL files from each designed flow and outputs a set of Javascript and AIML files. These resources will be further presented and explained.

## Implementation



Figure 3.1: Engaging Pattern Example

### 3.1.1.1 Engaging Patterns

Each flow can be activated by specific keywords or sentences predefined by the modeler designer by simply creating an Engaging Pattern as seen in 3.1. At this point it would be possible to interact with the chatbot and this flow would be triggered upon the user's input.

As mentioned before there are two different outputs for each flow. A javascript file that encapsulates each element and has different templates according the defined rules. A small example can be found below.

The other output is an AIML file. This is used to map the user input to the different existing flows. However this would not be a scalable technique when using engaging patterns and a different approach was taken to tackle this problem, taking advantage of NLP that will be presented further.

```
exports.Start_Example_Engaging_Pattern = function(data,done) {  
    log("Start_Example_Engaging_Pattern");  
    // add any type of logic necessary to engage the pattern  
    done(data);  
}
```

### 3.1.1.2 Task Elements

Another important part of the BPMN modeler environment are the task elements. They can exist under the form of User, Message or Logic Tasks 3.2. Their purpose is to wrap into different section the flows. These task types will result in considerably different AIML and JS files generated.

Message tasks can be highly customized depending of the data type that they request from the user. Platforms like Facebook allow different type of messages such as carousel, quick-replies and text 3.3 that are mapped directly to this type of task in its modeler properties 3.4. Generally before this predefined messages are sent, there is a more generic text message that tries to regulate the user response by providing additional information from what is being asked. In this case AIML is capable of problem solving in a successful manner as it only results in a set of messages that are going to be sent to the user according different standards.

However it is important to know the person state of mind when interacting with the chatbot in order to choose which type of messages should be sent. WBP solves this problem by having



## Implementation



Figure 3.2: Different Task Elements

algorithms of sentimental analysis and different types of predefined messages depending of their result. A sample of an AIML file can be seen below.

```
<category>
  <pattern>WHICHSINGLEDEVICE_NORMAL</pattern>
  <template>Hey! Haven't seen you in a while. Which device from
    your list you want to schedule a job for?</template>
</category>
<category>
  <pattern>WHICHSINGLEDEVICE_ANGRY</pattern>
  <template>Which device you want to schedule a job for?</template>
</category>
```

User messages are exactly what their name indicates. Logic tasks should be reserved if there is something that does not fall under user or send messages, they can be entirely avoided but they are highly recommended as it facilitates code modularity.

### 3.1.1.3 Gateways

Until now it would be impossible to alter or conditionate the flow's direction, this is where gateways come useful. It offers the opportunity to validate user data and change the flow if needed. Gateways can also be used to advance several states of a BPMN flow, this is extremely useful if taken into consideration that an user can at any time answer more than it is asked and the chatbot should work as autonomous and smart as possible, simulating a normal conversation. A simple example of all this elements can be seen in the figure [3.5](#)

### 3.1.2 Amazon Lex and Alexa Integrations

Natural Language Processing is a must for every chatbot. This comes from the need to understand exactly what the user is saying, easing that problem. As stated in the chapter [2.2.5](#), WIT Bot Platform only had integrations with WIT.AI and API.AI, and interest was shown to extend this NLP connectors list and LEX seemed appropriate.

Amazon Lex can be used to create a standalone chatbot and deploy among several platforms, but the WBP already does this, instead was decided to use this as NLP connector and add to the list. Every time that an user uses the chatbot, input is sent to LEX which decodes its meaning.

## Implementation

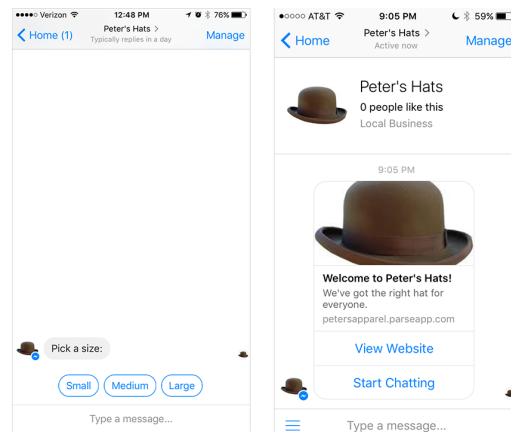


Figure 3.3: Facebook Example Messages

This is the solution referred in 3.1.1.1, using NLP mechanism we can substitute the basic AIML string matching and get actually a mechanism that acts smartly. It does need some configuration, there is a list of utterances and intents that needs to be filled so that the output can be controlled and within the designed flow. This is done by configuring the intents with the same name that the engaging pattern previously described.

Alexa also uses LEX, however its integration and usage is completely different. It has some constraints. First of all Alexa skills only permit one message at a time, this is a problem when chatbots have several set of messages which provide context at different times. Conversation flows are also affected by this as all messages need to be sent at once and punctuation becomes even more fundamental. At this time there is not a notification system which harm some capabilities based on patterns suggestion. Although Amazon already announced a developer kit which will allow developers to create notifications and send them to their Alexa skills.

A voice interface is a complete different paradigm than text messaging a chatbot. This became progressively evident as tests were done. The flows needed to reflect this, as the user is even more unpredictable when interacting using voice. This is evident by the recent launch of Amazon Echo with a monitor, helping in this way the normal usage of the device. Several solutions came to mind and the chosen approach was to create a gateway at each state that could leap into the every following state and adding utterances accordingly. This is important because at any time an user can not only respond to the question but also provide further information, for example, when an user is prompted to choose an hour to schedule a task he might answer with the hour and day. It would not bring a sense of accomplishment if the next question of the chatbot was regarding which day he would like to schedule the task for. This solves exactly that.

Both platforms were implemented and added to WIT Bot Platform only requiring the developers to provide the amazon credentials to access the Amazon Lex Chatbot.

## Implementation

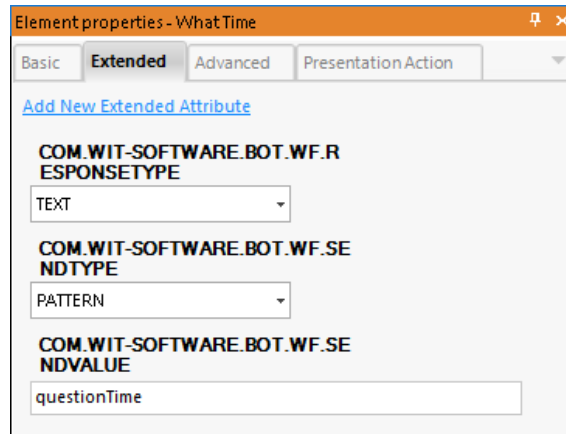


Figure 3.4: User Task Properties

### 3.1.3 Chatbot Use Cases

In the previous chapter [3.1.1](#), was given a detailed view on how to create a flow. In this chapter will be presented both the flows that were implemented during the course of this project and some code samples that are essential to their understatement.

#### 3.1.3.1 Turning Device State

This use case by itself is not extremely innovative however it would be a deal-breaker if not implemented. With this in mind was created a flow that can be seen in [3.6](#).

This flow contains four engaging patterns. This is due to the fact that it is important to support any type of input from the user. Different sentences may not indicate different flows. The table [3.2](#) explains some utterances that might trigger the engaging patterns of this use case.

Depending of the triggered pattern there are a few differences in the chatbot's responses. When an user first interacts with the platform there is a request to the abstraction platform that retrieves all the information about the user devices. This gives the possibility to help the user achieving self realization, for example when an user is trying to engage a conversation by saying "Turn off the light" the chatbot does not know the location, so it can not act unless he specifies the light name or location, however it does know that it will be a light that is currently on. So a quick-replies list with only the devices that are turned on is given to the user as can be seen in [3.7](#).

Table 3.1: Engaging Patterns on Turning Device State Use Case

Engaging Pattern	Utterances
Turn Light With Status	Turn <i>status</i> the light
Turn Light With Location and Status	Turn <i>status</i> the <i>location</i> light
Turn Light With Location	Turn the <i>location</i> light
Turn Light	Turn the light

## Implementation

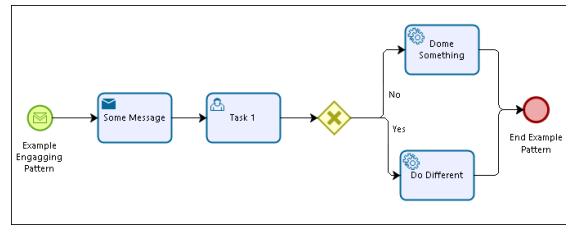


Figure 3.5: Example with all elements from BPMN

This use case also obligated the creation of two types of data structures among Alexa and LEX platforms, status and location. The status is self-explanatory, is a set of words that an user can say to whether turn on or off each device. Regarding the Location there is another problem, it is unlikely that all users will have exactly the same name in their lights as it is on this set. For example, nothing restricts the user to have a random name mapping to a light, and we want to allow them to do so. The strategy taken to tackle this problem was not only using NLP to try to map to one of our listed locations but also using string match to the name coming from the abstraction platform with the current devices. This strategy proved to be successfully.

### 3.1.3.2 Scheduling Events

This set of cases that use scheduling events are extremely useful and can help the user feeling real advantages regarding other systems. There are two different flows that can be classified as a scheduling event. The first one 3.8, is a simple job where an user can set a device to change its state at a certain time. The second, can be seen in 3.9, is similar in every way however it creates a schema, and an user can add more than one device at a certain time with periodicity.

These flows are again highly flexible as both of them have a few engaging patterns, to again tackle the problem of unpredictability of the user input. In the table, is all the engaging patterns combined with some example utterances that trigger them.

In addition to the set of data types referred in the previous flow were added an AMAZON.Time and AMAZON.DayOfTheWeek data types. This one, as the name indicates, is a native type provided by amazon that handles all dates parameters, allowing to easily process the data of examples such as “Turn on the kitchen light tomorrow at 4.30pm”.

This type of scheduling events can be achieved by calling the abstraction platform created and will be detailed in further sections.

Table 3.2: Engaging Patterns on Scheduling a Job Use Cases

Engaging Pattern	Utterances
Start Scheduling Event For Device	Schedule a Job
Start Scheduling Event For Device Without Day	Turn <i>status</i> the <i>location</i> light at <i>time</i>
Start Scheduling a schema	Schedule a schema

## Implementation

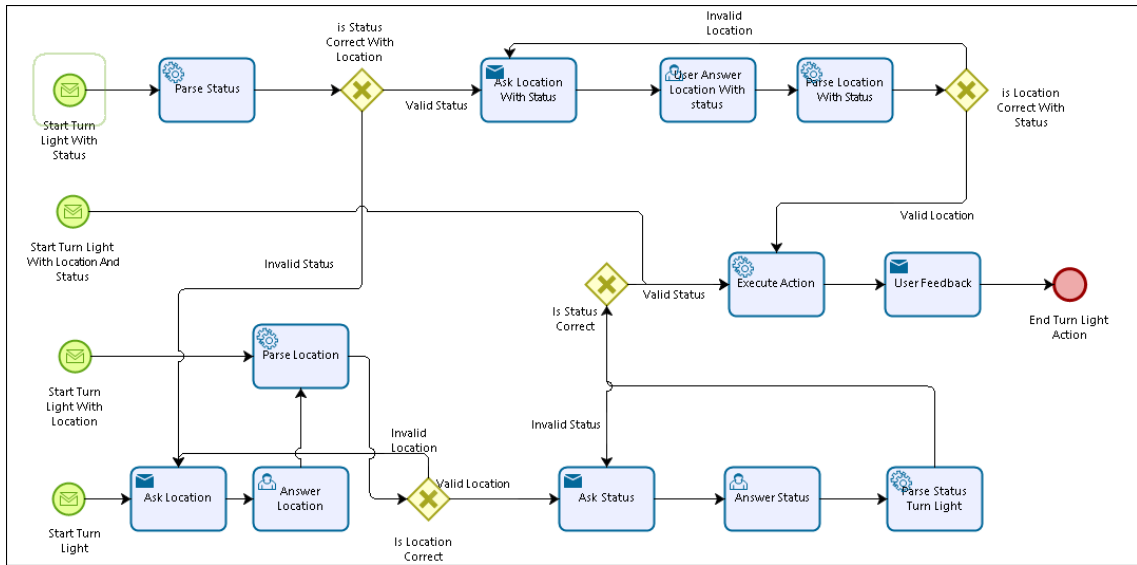


Figure 3.6: Turning Device State

### 3.1.3.3 Listing and Notification Events

The set of flows that fall under this section were created with the intent of helping user both achieving their intents and also make them feel like are using a true home assistant. Something that can learn from usage and act upon it. This is achieved by machine learning algorithms that will be detailed in further chapters.

Each engaging pattern in 3.10 is a independent flow. This flows are triggered both by utterances provided by the user and by tasks that run periodically in the abstraction layer for each user.

First in the list is “Start Unusual Consumption”, this flow uses machine learning algorithms that analyze data from devices usage and notify the user if something unexpected happens. This can be useful for example when highly sensitive devices stop or start consuming energy all of a sudden without the user noticing them. The user can define which devices are sensitive and that this analyzes will be focused on, an example of the chatbot notification can be seen in 3.11. When receiving this messages an user can decide whether he wants to act upon it directly or just ignore it.

The next engaging pattern is similar in every way regarding how the flow is triggered. The difference is that instead of a unusual consumption suggestion it gives a pattern detected on each devices, grouping those if within a small time frame. An example can be seen in the middle screen shot in 3.11, after the notification is sent an user can again opt to either make the suggested pattern a schema or ignore it.

The other engaging patterns present in this section are more informational than the previously described. They try to frame the user intents while providing useful information.

## Implementation

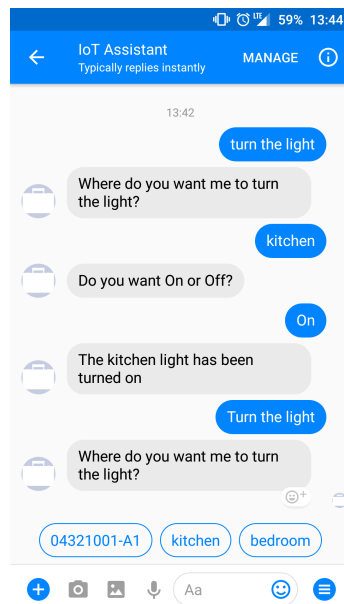


Figure 3.7: Quick replies mechanism example

### 3.1.4 Conclusions and Important Remarks

During the chatbot development were encountered some difficulties that needed to be tackled smartly. One of them, and probably the biggest, was the fact that it is extremely different to interact with a chatbot using voice or text. At first this did not seem that obvious but when using some of the first use cases with Alexa this became obvious. No one likes to use something that is highly unresponsive and slow to achieve the final goal. The problem was solved by giving the chatbot the capability to jump within states and a special attention to the messages sent to the user. This was a solution that increased a lot the code complexity but the final result was worth it and much needed.

The second major problem was never let the user without knowing what the chatbot capabilities are, or what to do next. This was solved in a few different ways. First of all there is needed a lot of thought put into each sentence given to the user. This messages are used in association with carousels or quick-replies providing as much as possible the current context. It was also implemented a persistent menu for the messaging interfaces that makes the usage of the chatbot way easier, at any time an user can see exactly what are the capabilities of the tool in question.

Another problem, as stated in previous sections, is the unpredictability of an user's input. At any time, during a flow, nothing holds the user back to ask something entirely different and every question needs an answer. At this point the solution consists in two distinct paradigms. If an user asks something that is completely random, an AIML database will come in rescue and answer the user, after this message is sent back to the user a steer back mechanism will come in play giving promptly feedback to where he was before diverging from the flow. The other paradigm results from an user input that is going to trigger a different engaging pattern, at this point an user must

## Implementation

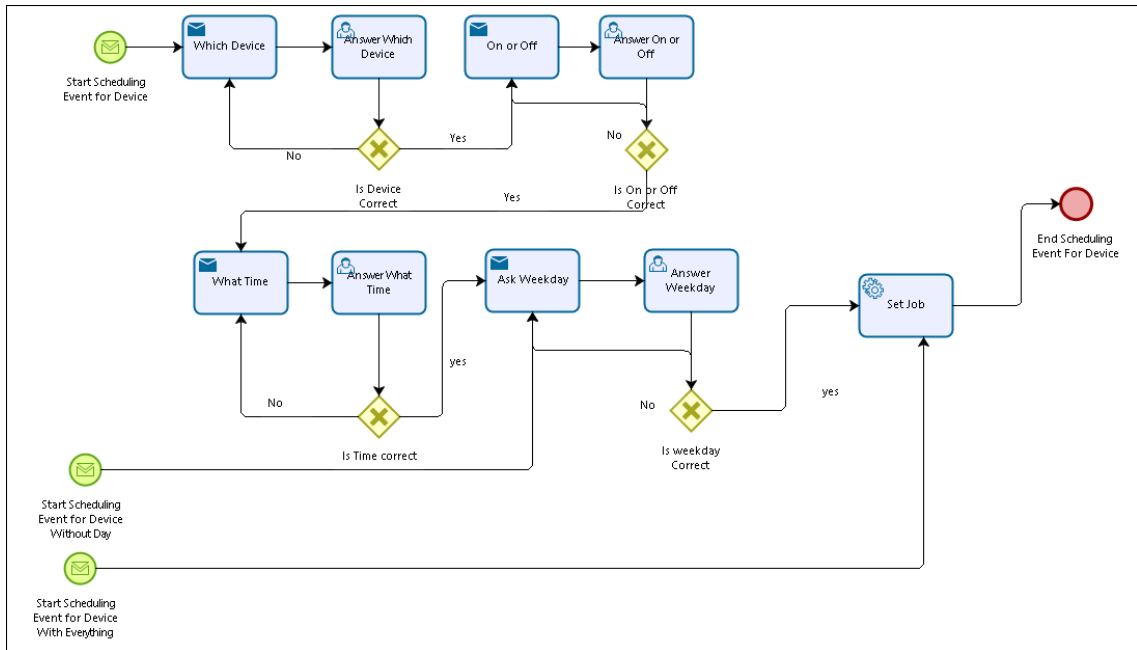


Figure 3.8: Schedule a Job

decide if he wants to cancel the current transaction and engage on the new one, or simply ignoring the previous message and continue the current flow, again if this was the option he will be given feedback to where exactly he was before.

Chatbots are a great way of working, but they are far more complex than they appear.

## 3.2 Abstraction Platform

In previous sections of this chapter there were some references to an abstraction platform. At first was not meant to be created an additional server but during the development process it was obvious that an abstraction layer was needed to separate the chatbot itself from all its domotics and machine learning integrations. This is exactly that, a django server that implements both integrations to Scikit-Learn and WIT Software's domotics platform while providing a RESTful API to the chatbot.

In this section will be presented the data models that were created to achieve this requisites, a further explanation of how the domotics abstraction works and how can other systems be integrated seamlessly, a deep look into the machine learning module and how it works. Will be also provided a small documentation of the created routes to help to get a better understanding how will the chatbot consume this platform's services.

## Implementation

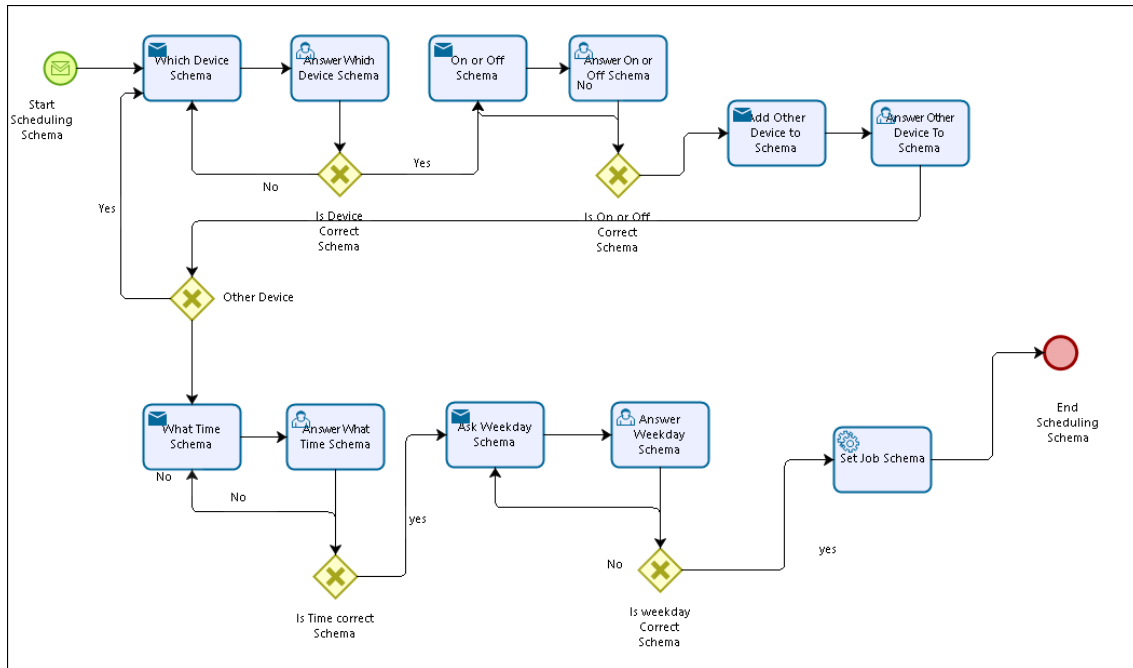


Figure 3.9: Schedule a Schema

### 3.2.1 Server Overview

This platform works as an centralized server to all the chatbot needs. With this in mind was required the design and implementation of a database schema that could handle with ease all the users information. In the image 3.12, can be seen the implemented system. Users table is as expected the main model in the system. It has a one-to-many relationship with the Actions Models, which holds all the information about state changes of every device connected to the domotics system. The Domotics System model is an abstract class that the system interacts through to execute the required actions, each user at the moment is only allowed to have one domotics system associated. This is the class that offers flexibility and from which other Domotics Systems should be integration from. At this moment there is only one domotics system implemented, the one used to create this chatbot, which is EFAPEL Domotics System. EFAPEL's model holds all the information necessary to authenticate the user and other domotics systems should follow the same rules. There is also a one-to-many relationship between users and schemas table, this is either the detected patterns or the rules that the user decided to create to optimize the home assistant usage.

#### 3.2.1.1 Domotics System Integration

It would be pointless to create a proof of concept without a working demonstration of the product. WIT Software allowed me to have the opportunity of integrating their developed system for EFAPEL with their hardware. In this subsection will be described the process of integrating this system and how it had effect in the final product.



## Implementation

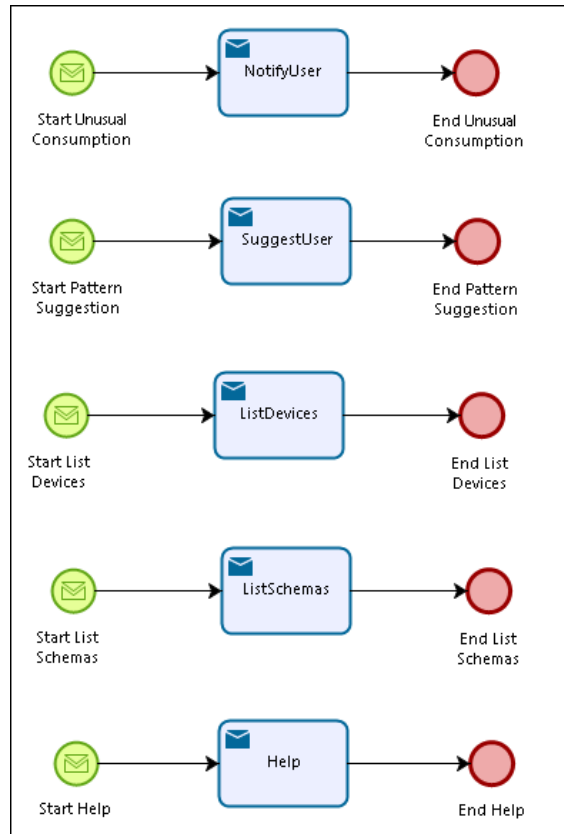


Figure 3.10: List and Notification flows

The first step was authenticating with the domotics system, there is information specific to each domotics system that need to be provided. At this point two solutions to gather this information were possible. Implementing a flow for each specific domotics system or creating a web application that would be prompted to the user after the first engagement with the chatbot. The second seemed more appropriate and was the solution taken. There is a lot of useful information that can be taken from the list of endpoints provided by this platform, however there is not one that provides the state information of each device over time, another step had to be made to achieve this type of data, as its use is crucial to suggest the user based on previous behaviors.

For the sake of this dissertation there was found a solution that work as a proof of concept but it is completely unscalable and impossible to have working in a production application. It was created a python script that connects by a machine-to-machine connectivity protocol, denominated MQTT, that listens to state changes of each device and posts the information to the abstraction platform API afterwards. This events are passed as proto buffers, which is an easy way of serializing structured data, an example of the proto class can be seen below.

```
message DeviceStateEvent {  
    required MessageInfo info = 1;  
    required int64 deviceId = 2;
```

## Implementation

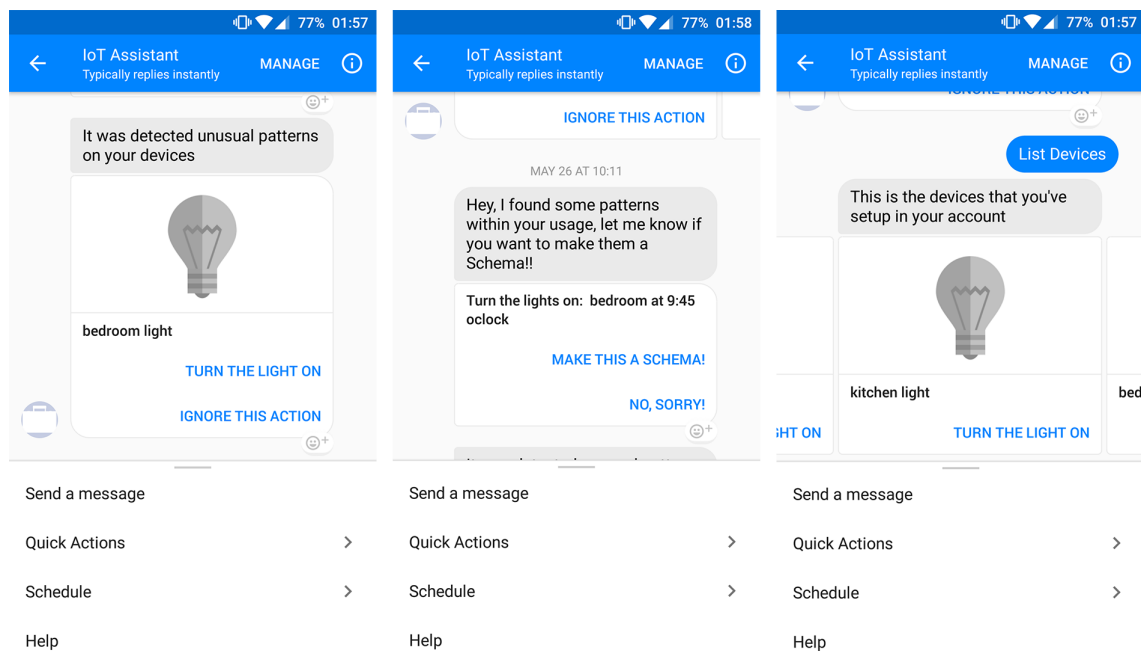


Figure 3.11: Chatbot Use Cases Examples

```
// Button specific
optional DeviceEndpoint endpoint = 100 [default = NoValue];
optional ButtonState buttonState = 101;

// Energy specific
optional RegulatorState energyChangeState = 200;
optional uint32 energyLevel = 201;

// Movement specific
optional uint32 timeActive = 301;
}
```

This solution enabled access to the devices state change, not only by the usage within the application but also by toggling physically, which is key.

EFAPEL's API can be used to schedule jobs and schemas, that would be an huge help, however it is important to create something that can be flexible enough. In this platform there is a mechanism that creates both CRON and AT jobs depending of if it is a schema or a job respectively. This was created with the intent of helping future domotics systems integrations that do not have this capability.

## Implementation

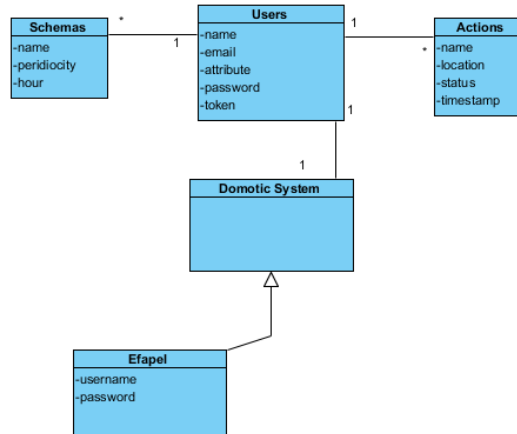


Figure 3.12: UML Class Diagram of the implemented models

### 3.2.1.2 RESTful API

In the table 3.3 can be seen the most important routes created to feed the chatbot. All this receive and return JSON objects. To access the abstraction platform's API is required to pass through an authentication mechanism, which uses OAUTH2, that ensures the user executing this actions is in fact who it is. This authentication protocol is also used to authenticate the user among the several channels, such as Amazon Alexa and Facebook.

The first set of routes, `/users/user_id/actions`, are used to handle all the device actions. Also if an user changes the connected items to the domotics system it updates their values accordingly. The second route group is the `/domotics/` one. When an user first interacts to the chatbot he has to give some information that will make him able to use the chatbot. The `/social/` is the route list that handles all that for the different channels. Every usage of the devices fall under `/devices/`. This routes are used to change a device state or get all the devices for the current user. Finally there is a route that handles the scheduling events for each user, it has different parameters that will decide the periodicity of the job and which devices will have their state changed.

Table 3.3: RESTful API Routes

HTTP Verb	Route
GET	<code>/users/</code>
GET/POST/PUT	<code>/users/user_id/actions/</code>
POST/GET	<code>/users/user_/schemas/</code>
GET/POST	<code>/domotics/</code>
POST	<code>/domotics/efapel/</code>
GET	<code>/domotics/devices/</code>
PUT	<code>/domotics/devices/device_id/</code>
GET/POST	<code>/domotics/social/</code>

### 3.2.2 Machine Learning

Providing artificial intelligence capabilities to the chatbot was an important key point during the development of this dissertation. Previously, [2.4.2](#), was described a tool to reach this type of objectives, in this chapter will be provided the full scope of the implementation and data management to reach the current model.

#### 3.2.2.1 Data Model

The first step to choose a strategy to handle this problem was knowing exactly the type of data that was possible to be gathered for each user with the current application. As mentioned before there is a few different endpoints that can feed this data to the machine learning model, however this data is not uniform as we can have data from several devices and it is important to tackle this accordingly. Another point that needed to be taken into consideration was how quickly the algorithm can make trustable suggestions, it would not make any sense to have a super complex model that would only be ready to use after a couple of years. With this in mind, and after several changes, was decided to create a model with only the most important features, them being:

- **Weekday:** This field takes zero or one as values depending if it is a working-day or not. The workdays can be defined by the user when first engaging the application as they may fluctuate over users.
- **Device:** There are different devices that take distinct measures, it is important to separate them and analyze taking this into consideration, this field does exactly that.
- **Time:** It is important to know when a certain event occurred, however it is even more important to group events that occurred under similar circumstances. This feature aims exactly at that, each day was divided into fifteen minutes groups that go from 0 to 96. When an certain event happens it passes through a classification algorithm that labels it not only by the exact time of occurrence but also by the proximity of data near that point. This is useful to the suggestion use-cases previously described.
- **Value:** This feature is highly attached to the device in question. It can be either temperatures, lights, energies and so on. Is one of the most important features in the model as expected and is the column that the model will attempt to infer after learning the user's pattern.

This model was defined with the intent of solving and providing data to the uses cases of pattern recognition described in [3.1.3.3](#).

#### 3.2.2.2 Possible Approaches and Solution

At this point the model is well defined, however there is not any reference to the algorithm used. In the process of finding the solution which provided the best results there were two studied approaches. The first was a classification algorithm that used linear regression to infer results, the

## Implementation

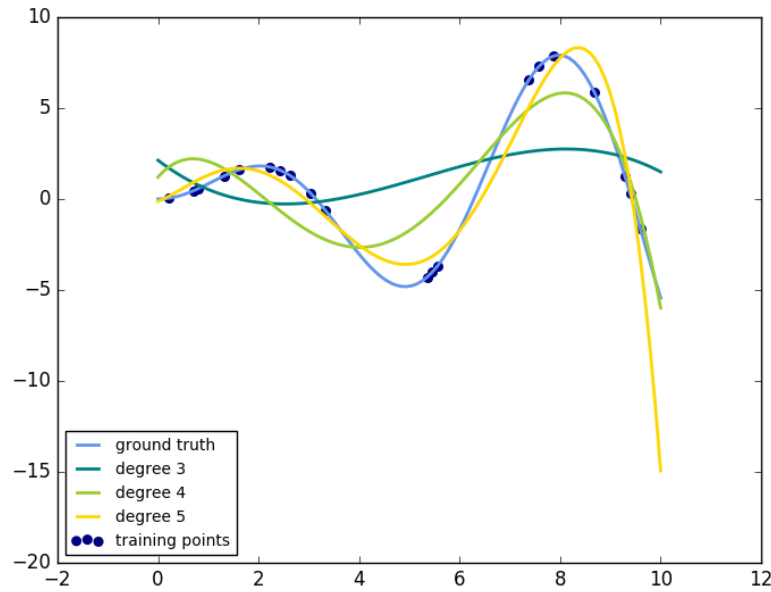


Figure 3.13: Polynomial regression approach example

second was an clustering algorithm that, as the name suggests, groups the samples that contain similarities.

This last approach was though as being hard to implement as the results can be highly subjective, and this was proven right. In this approach was used K-means, an algorithm that tries to minimize the sum of squares of the samples within the same clusters, using Skicit-Learn. After a while it was obvious that this was not the way to go, at least with the data samples available at this point. The results were not the expected and it was impossible to generalize them enough to serve as a pattern recognition tool, even though the devices were being grouped had a relation within them would not be reasonable to proceed with this approach.

The chosen solution was to use a classification algorithm, allowing to both analyze the power consumptions of each device and suggest patterns from those analyzes. It is a linear regression algorithm that tries to minimize the residual sum of squares between the different samples in the dataset. The data generated by the usage of chat is however nonlinear. This required the implementation of Ridge Algorithm with Polynomial Regression, keeping the fast performance of linear methods while fitting the data as required. An example of how this polynomial algorithm modeling non linear data can be seen in [3.13](#).

### 3.2.2.3 Data Treatment

Before feeding the data model is important to work the dataset taking into consideration a number of things. First there is a treatment that aims to remove the samples from the dataset which their values are out of scope. For each device is calculated a few metrics such as mean and median that

## Implementation

will afterwards be used to discard the outsiders. As stated in the previous section, the data that makes part of the model is the events triggered by the user. However there is a lot of missing data that we actually know the values and it is important to specify it clearly so the model knows it as well, for example, when there is two samples that are turning a device on and off, during the span of time that there is not any information about the state device, we know for a fact that the state is equal to the state before the last toggle. This flow of changes to the data is the mechanism that transforms it from an PostgreSQL database to the dataset that will be used by the model.

### 3.2.2.4 Model Refinement and Results

At first it was intended to have one model per user, having all devices within the same model and infer the current state of the device from there. However the results were not good enough for such a small dataset. The solution was to create a model for each device following the model previously described. When the user prompts a power consumption analyzes or a CRON job is triggered it creates separate models and afterwards the data is merged into one. This allowed to have better suggestion in fewer time, which is also important.

Since the beginning of the machine learning model implementation it was fundamental to test it under a larger and more complete dataset. With this concern in mind it was necessary to find a raw dataset where the solution implemented could be validated. In [FB16] was found exactly that, a raw domotic's dataset with over 800k samples distributed over several devices. This data set required again a lot of data treatment in order to be used with the current model as it is currently hosted under a mongo database. Different tests were done with the intent of modeling the power consumption and the likelihood of a device's state. The model proved to be working as expected.

## 3.3 Implementation Summary

During the course of this dissertation, the implementation path was not always as smooth as expected. There were a considerable amount of challenges in the way that required some versatility and work.

The first big encounter was learning and implement a bot using the WIT BOT Platform. It is a system that has been in development over a long time and it is quite complex. The integrations done proved to be within the standards and are now apart of the project. Following this first step it was necessary to create a Django abstraction layer. At this point having no experience in python hardened the process but quickly was overcome. The other big stone in the path was the integration between all systems and the domotics platform already developed by WIT Software, this was done seamlessly due to the documentation and help provided. This was the end of the first milestone of the project, a small prototype was implemented and the first feedback started to come rapidly, mostly by meetings within the company. The next big step was creating the machine learning module. During the course of its implementation there was some changes that resulted in the previously described model. After all the system integration the proof of concept

## Implementation

was complete, and an usability test was done to evaluate the final project. The results from this test will be described in the next chapter.

## Implementation



## Chapter 4

# Experiments and Results

Chatbots are an innovative way of creating an application as seen in previous chapters. However they suffer from user traction problems, it is common to a chatbot lose a lot of users after the first couple of engagements as they usually lack content. To validate the concept of the application was done a questionnaire to FEUP students and WIT Software collaborators, which aimed to get information about the user possible adoption and traction, based on the problem-solution fit technique. The next step to validate the project consisted in a set of usability tests that were done under WIT laboratories to their collaborators.

### 4.1 Questionnaire

The questionnaire contains eight questions and had the participation of ninety one inquiries, for each question will be explained their purpose, results and conclusions.

- **Question 1 - Age group**

This question was both useful to evaluate the inquiries age distribution and to check if their age had influence in the remaining of the questions. The results can be seen in [4.1](#). Analyzing this graphic it is possible to see the disparity between age groups. This is due to the channel where the questionnaire was proposed on, however, even considering the small sample, there were not differences in the rest of the questions depending of the age.

- **Question 2 - Do you have knowledge of any application to manage domotics?**

- **Question 3 - Which Use Cases would make a good domotics application**

- **Question 4/5 - Do you use or would be interested in an application that implemented those use cases? Justify the answer above.**

This group of questions were designed to know whether the users identify with this types of applications, which use cases they would prefer to see implemented that added value to the application and to know if this is indeed a problem that needs to be addressed.

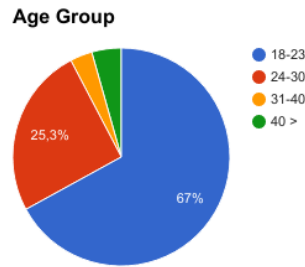


Figure 4.1: Age group, question 1

Analyzing the results 4.2 from the second question it is evident that the domotics paradigm is still not quite yet established in the market. It has been evolving in the past years and is only now gaining roots. Domotics is undeniably a topic that has come to stay and will gain traction over the next years, and this responses suggest that now is when a smart solution needs to appear. The following questions were to see what an application like this should contain to be appealing and if the implemented solution is going in the same direction as the users want. The results from these questions can be seen in 4.3 and 4.4. A lot of the inquiries suggested use-cases previously described in the implementation section, this is surely a good sign. They also answered affirmatively when asked if were willing to use an application that would implement them. However this is much an “Ice-cream question”, who would say no if an application filled all their requisites? The results can not be taken as assured clients but more like interested users, which not being the same is again a good marker. When asked to justify their answer to the fourth question, answers such as commodity, time-saving and household energy consumptions reduction were often referred. There were as well several references to the amount of money that this system usually require to have access to the entire house, which is a big problem that is holding the domotics paradigm for now.

- **Question 6 - Suggestion based features require the application to store sensible information. Would you feel comfortable knowing that the application does it?**
- **Question 7 - Would you still use the application without the suggestions module?**

In previous chapters was mentioned the security issues of handling such sensitive data, this group of questions were designed to know exactly what the users think of that topic. The results from 4.5 are clear, not all the users are completely available to allow an application handle and store this type of data, however there are features that require it in order to work. This is mostly the suggestions use cases that use machine learning algorithms that feed from this data, their storage is required. Since the beginning of the implementation the system was implemented in order to allow the user to choose each modules he would like to use.

## Experiments and Results

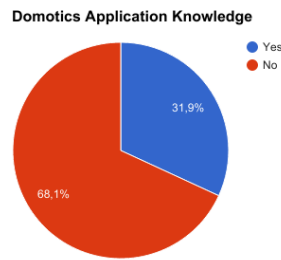


Figure 4.2: Domotics applications knowledge, question 2

The next question was exactly with this module system in mind, even though it adds value to the user having this suggestions and warnings they are not completely necessary as the results in 4.6 show.

- **Question 8 - Which channel would seem appropriate to this type of applications?**

There are numerous ways of using a domotics application, this question was to know which was the preferred option from the possible users, the results can be seen in 4.7. Even though this application is not available in iOS/Android it was obvious that the answer would be a smart phone application. This results prove that the market is not entirely confident in messaging applications as services, however this results do not go against everything in this solution, if something it helps showing possible growths of the product and its possibles channels of use.

## 4.2 Usability Test

The usability test aimed to validate the solution implemented. There were selected ten collaborators from WIT Software which had experience using chatbots with the objective of receiving informed opinions. These collaborators where divided into two separate groups at first. Each group was presented with a different way of interacting with the chatbot, voice or text and afterwards repeated the same exact tests with the system as a whole. This was to test both interfaces separately and improve them accordingly. Before the task's execution a quick presentation where the global scope and use cases were briefly introduced.

### 4.2.1 Results and Conclusions

The first set of results can be found in 4.1. The overall result test prove that it was possible to almost everyone complete the tasks proposed. This table presents the average result for the difficulty on each task, where 1 is easy and 5 is hard. Scheduling a schema is something that requires a lot of user interactions if the correct utterance is not given. The fact that the difficulty level was 3 proves that this flow needs still some refinement and that is far from being satisfactory.

## Experiments and Results

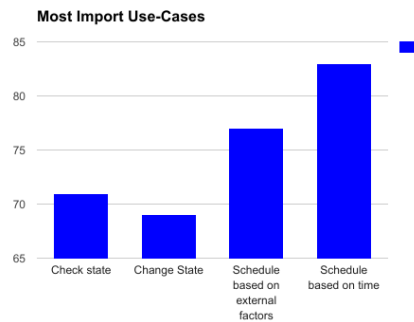


Figure 4.3: Use cases, question 3

The second group of results [4.2](#) show exactly what in previous chapter was described. There is way harder to interact with the chatbot through voice. In these results some tasks proved to be very confusing making the users not available to end it at all. Other tasks, such as asking for the unusual power consumptions, failed to detect some utterances that were obvious examples but were missing in the NLP mechanism.

At this point the majority of the users had already successfully executed the task so instead the results [4.3](#) try to map the users preference and commodity while using the system as a whole. As expected there is a preference on using different interfaces to perform distinct actions. This would be even more obvious under certain circumstances, for example: a loud environment may not be the most appropriate for a voice interaction. These results prove that the system works best as a whole, users were more willing to use the voice interaction to accomplish small and simpler tasks as turning a device state but whenever there was a more complex interaction the text was always the chosen one. All the feedback resulted in the current state of the chatbot flows described in [3.1.3](#).

## Experiments and Results

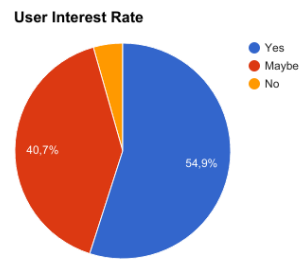


Figure 4.4: User interest rate, question 4

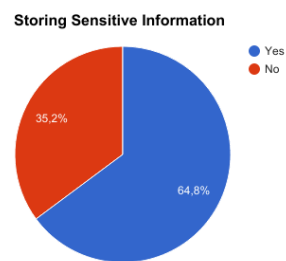


Figure 4.5: Allow storage of sensitive information, question 6

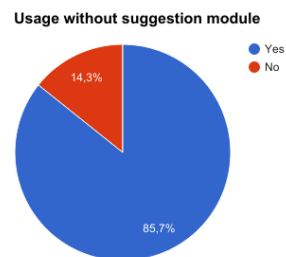


Figure 4.6: Application interest without storing information, question 7

## Experiments and Results

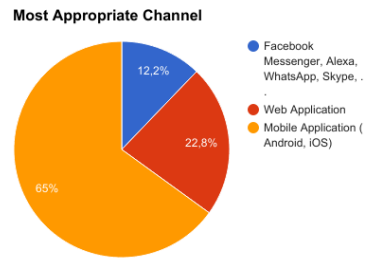


Figure 4.7: Different preferred channels to use the application, question 8

Table 4.1: Usability Test, Facebook Messenger Results

Task Number	Effectiveness Rate %	Difficulty (1-5)
1	100	1
2	100	1
3	100	1
4	80	2
5	80	3
6	100	2
7	80	3

Table 4.2: Usability Test, Amazon Echo Results

Task Number	Effectiveness Rate %	Difficulty (1-5)
1	100	1
2	100	1
3	60	4
4	80	3
5	60	4

Table 4.3: Usability Test, System Results

Task Number	Interface Used	Difficulty (1-5)
1	Text	1
2	Text	1
3	Voice	1
4	Voice	2
5	Text	3
6	Text	2
7	Text	3

## Chapter 5

# Conclusions and Future Work

### 5.1 Conclusions

The main focus of this dissertation was to create a chatbot to manage domotics with artificial intelligence. These are three separate things that need to be addressed as such. The creation of a chatbot is an harder task than though at the beginning, specially if taken into consideration the possibility of using voice as mean of communication. This step was overtaken as described in the chapter [3.1.3](#). In this solution there was some focus in creating use-cases for lights as it was the only hardware available, however the proof of concept can easily scale for other devices without hassle. Applications that manage domotics systems are quite hard to implement due to the broad set of devices and technologies available in the market. The solution used to to tackle this problem is described in [3.2](#). Last but not least is the artificial intelligence module. It is more important than ever to treat the data all over the Internet. The solution find to solve the suggestions problems can be found in the chapter [3.2.2](#).

Chatbots are an innovative and exciting way of creating applications. It can easily reach a large user-base in their everyday applications. However, developing a chatbot requires a lot of work. During this dissertation was evident the problems of using a voice interface to control an application, the lack of a screen to complement the information displayed to the user was a nightmare. More recently, and to address this problem, amazon launched a new product which is an Amazon Echo with a screen. This is without a doubt a step forward as it facilitates hugely the user interaction and therefore the developers work. Regarding the text channels it is an easier process, as for now Facebook implements a rich feature set into their messaging applications that make it almost a standalone application. This paradigm of applications are gaining their traction in the market however they are a few steps from being accepted as such as seen in [4.7](#).

### 5.2 Future Work

The abstraction platform [3.2](#) was created with the goal of supporting several domotics systems. One of the most obvious additions to the work already done is their integration.

## Conclusions and Future Work

Other necessary implementation to have this application in production would be the integration of several other use cases contemplating the different intelligent devices that we usually find in our homes. Toggling a device state can work as a general use case however there is a need to know exactly which device are we changing states to have a more natural conversation with the user while providing a tailored experience.

Chatbot's main problem is the lack of context given to the user upon the first engagement. Usually the user are left without any type of feedback when interacting with the application, to solve this problem there are a few implemented mechanism however it is important to refine them and implement even more contextual feedback mechanisms.

After the questionnaire there was one obvious future requirement, mobile application. This was though since the beginning as well. There would be a module where the chatbot would fit. Allowing like this both normal application usage or voice interactions under the several channels or within the application.

Last but not least there are some performance issues that need to be addressed in order to scale. At this point both the chatbot does not have any type of information regarding the user devices which adds a huge amount of delay when first engagement. This is because the user can update their devices and the platform should be able to address this accordingly. Many solutions come to mind in order to solve this problem, being a simple refresh button which updates the information on chatbots database or a simple CRON job which updates this information.



# References

- [AIM10] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.
- [Ama16a] Amazon. Amazon alexa. Available at <https://developer.amazon.com/alexa>, December 2016.
- [Ama16b] Amazon. Amazon lex. Available at <https://aws.amazon.com/pt/lex/>, December 2016.
- [biz17] bizagi. Bizagi modeler bpmn. Available at <http://www.bizagi.com/en/how-we-help/process-modeling>, May 2017.
- [CLZ<sup>+</sup>00] Joyce Chai, Jimmy Lin, Wlodek Zadrozny, Yiming Ye, Margo Budzikowska, Veronika Horvath, Nanda Kambhatla, and Catherine Wolf. Comparative evaluation of a natural language dialog based system and a menu driven system for information access: a case study. In *Content-Based Multimedia Information Access-Volume 2*, pages 1590–1600. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D’INFORMATIQUE DOCUMENTAIRE, 2000.
- [DRKN<sup>+</sup>98] Anne De Roeck, Udo Kruschwitz, Philip Neal, Paul Scott, Sam Steel, Ray Turner, and Nick Webb. Ypa—an intelligent directory enquiry assistant. *BT Technology Journal*, 16(3):145–154, 1998.
- [Dun16] Jeff Dunn. It’s been a good year for the amazon echo. Available at <http://www.businessinsider.com/amazon-echo-sales-figures-stats-chart-2016-12>, December 2016.
- [Fac16] Facebook. Wit ai. Available at <https://wit.ai/>, 2016.
- [FB16] M. Fernandez-Carmona and N. Bellotto. On-line inference comparison with markov logic network engines for activity recognition in aal environments. In *IEEE Int. Conf. on Intelligent Environments (IE)*, pages 136–143, 2016.
- [Fu68] KC Fu. *Sequential methods in pattern recognition and machine learning*, volume 52. Academic press, 1968.
- [GBMP13] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

## REFERENCES

- [Hal15] James Hall. Google accused of invading privacy with pictures of house numbers. Available at <http://www.telegraph.co.uk/technology/google/9205486/Google-accused-of-invading-privacy-with-pictures-of-house-numbers.html>, April 2015.
- [Hor08] Tim Hornyak. Rfid powder. *Scientific American*, 298(2):68–71, 2008.
- [Lid01] Elizabeth D Liddy. Natural language processing. 2001.
- [Man16] Stefan Mancevski. How we automated in-house content sharing using slack chatbots. Available at <https://chatbotsmagazine.com/how-we-automated-in-house-content-sharing-using-slack-chatbots-da438c0bda5iap859ba>, September 2016.
- [MBRR07] Daniel Mack, Anne Behler, Beth Roberts, and Emily Rimland. Reaching students with facebook: data and best practices. *Electronic journal of academic and special librarianship*, 8(2):4, 2007.
- [Mer16] Ian Mercer. Home automation. Available at <http://nlp.abodit.com/Applications/HomeAutomation>, 2016.
- [MKM<sup>+</sup>94] Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The penn treebank: annotating predicate argument structure. In *Proceedings of the workshop on Human Language Technology*, pages 114–119. Association for Computational Linguistics, 1994.
- [MMGA<sup>+</sup>15] Javier Miranda, Niko Mäkitalo, Jose Garcia-Alonso, Javier Berrocal, Tommi Mikkonen, Carlos Canal, and Juan M Murillo. From the internet of things to the internet of people. *IEEE Internet Computing*, 19(2):40–47, 2015.
- [MUM] MUMMII. A concept for chatbot: Sanelma, building engaging relationship between the work of art and the exhibition visitor. Available at <http://mlab.uiah.fi/mummi/sanelma/>.
- [pcm15] pcmag. The best smart home automation hubs of 2015. Available at <http://www.pcmag.com/article2/0,2817,2468647,00.asp>, May 2015.
- [Pet16a] Slav Petrov. Announcing syntaxnet: The world’s most accurate parser goes open source. Available at <https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html>, May 2016.
- [Pet16b] Slav Petrov. Building resources to syntactically parse the web. Available at <https://research.googleblog.com/2011/03/building-resources-to-syntactically.html>, May 2016.
- [PSW04] Adrian Perrig, John Stankovic, and David Wagner. Security in wireless sensor networks. *Communications of the ACM*, 47(6):53–57, 2004.
- [Ras06] Carl Edward Rasmussen. Gaussian processes for machine learning. 2006.

## REFERENCES

- [Rui15] Rui. 9 home automation open-source platforms for your projects. Available at <https://randomnerdtutorials.com/9-home-automation-open-source-platforms-for-your-projects/>, May 2015.
- [SA07] Bayan Abu Shawar and Eric Atwell. Chatbots: are they really useful? In *LDV Forum*, volume 22, pages 29–49, 2007.
- [Sof17] WIT Software. Wit bots platform ai-powered conversational interfaces. Available at <https://www.wit-software.com/products/wit-bot/>, May 2017.
- [Spy96] P Spyns. Natural language processing. *Methods of information in medicine*, 35(4):285–301, 1996.
- [WW11] Nihong Wang and Wenjing Wu. The architecture analysis of internet of things. In *International Conference on Computer and Computing Technologies in Agriculture*, pages 193–198. Springer, 2011.
- [ZBC<sup>+</sup>00] Wlodek Zadrozny, M Budzikowska, J Chai, N Kambhatla, S Levesque, and N Nicolov. Natural. *Communications of the ACM*, 43(8):117, 2000.
- [Zuc16] Mark Zuckerberg. Keynote speech delivered at the f8 conference at the san francisco design centre, 2016.

## REFERENCES

## **Appendix A**

# **Questionnaire**

# Questionnaire

6/6/2017

Assistente Pessoal para Domótica

## Assistente Pessoal para Domótica

O objetivo deste inquérito é avaliar o possível interesse em usar uma interface conversacional para controlar os dispositivos inteligentes nas nossas casas. Não só algo que execute as ações pretendidas mas também aprenda os nossos comportamentos e faça sugestões baseadas neles.

**\*Obrigatório**

### 1. Idade \*

*Marcar apenas uma oval.*

- ☐ 18 -  
☐ 18-23  
☐ 24-30  
☐ 31-40  
☐ 40+

### 2. Conheces alguma aplicação capaz de gerir os dispositivos de domótica? \*

*Marcar apenas uma oval.*

- ☐ Sim  
☐ Não

### 3. Quais seriam os casos de uso necessários para uma aplicação deste género ser bem sucedida? \*

*Marcar tudo o que for aplicável.*

- ☐ Ver estado de qualquer dispositivo  
☐ Mudar estado de qualquer dispositivo  
☐ Agendar mudanças de estado baseadas em fatores externos, por exemplo condições climatéricas  
☐ Agendar mudanças de estados baseadas em tempo, por exemplo "Ligar a rega às 17.00"  
☐ Outra: \_\_\_\_\_

### 4. Usas ou estarias disposto a usar uma aplicação similar no dia-a-dia? \*

*Marcar apenas uma oval.*

- ☐ Sim  
☐ Não  
☐ Talvez

### 5. Pequena justificação da resposta anterior

---

---

---

---

---

<https://docs.google.com/forms/d/1V4L0eNtEk6j7DJ5HBvpdJAunG1FZHNxGKdHwGnAySXs/edit>

1/2

Figure A.1: Questionnaire First Page

## Questionnaire

6/6/2017

Assistente Pessoal para Domótica

6. Para obter sugestões baseadas em comportamentos passados é necessário que a aplicação os guarde. Sente-se seguro sabendo que a aplicação guarda este tipo de informação? \*

Marcar apenas uma oval.

- ☐ Sim  
☐ Não

7. Continuará a usar a aplicação sem o módulo de sugestões sabendo que assim esta não guardaria qualquer informação sobre o seu histórico de uso? \*

Marcar apenas uma oval.

- ☐ Sim  
☐ Não

8. Qual seria o canal mais indicado para uma aplicação deste género? \*

Marcar tudo o que for aplicável.

- ☐ Facebook Messenger, WhatsApp, Skype, etc..  
☐ Aplicação Web  
☐ Aplicação Mobile (Android, iOS)  
☐ Outra: \_\_\_\_\_

---


Com tecnologia  
 Google Forms

Figure A.2: Questionnaire Second Page

## Questionnaire



## **Appendix B**

# **Usability Test**

The usability test consists in a set of tasks that represent the different use cases and aim to evaluate if the user can accomplish them successfully.

### **B.1 Amazon Echo**

Group of users only using voice interface interaction

- 1 - Start the conversation with the chatbot
- 2 - Change the state from your devices
- 3 - Check your device unusual consumptions
- 4 - Schedule a task for 5pm
- 5 - Schedule a task to run periodically

### **B.2 Facebook Messenger**

Group of users only using chat interface interaction

- 1 - Start the conversation with the chatbot
- 2 - List all the devices
- 3 - Turn the light on the living room
- 4 - Schedule a task for 5pm
- 5 - Schedule a task to run periodically
- 6 - List current scheduled tasks
- 7 - Disable currently scheduled tasks

### **B.3 Amazon Echo and Facebook Messenger**

Group of users that have exactly the same tasks as before but can decide whether voice or chat interfaces to solve them.