

Final Master Thesis

**MASTER'S DEGREE IN AUTOMATIC
CONTROL AND ROBOTICS**

**Design and Implementation of an Automatic
Speech Recognition Interface for a
Multipurpose Assistant Robot (MASHI)**

MEMORY

Author: Juan Elwin Alvarado Vargas

Advisors: Dr. Cecilio Angulo, Dr. Raul Ramirez Lopez

Date: October, 2017



Escuela Técnica Superior
de Ingeniería Industrial de Barcelona



Abstract

This Master's thesis is focused in the research, design and implementation of an automatic speech recognition interface using online services. This interface is intended to be used in a multipurpose robotic platform called *MASHI*. This improvement is presented as one of the series of possible improvements for the robot in order to be used in social experimentation inside public building such as cultural centers.

The system is considered to be implemented in two Raspberry Pi's using the Master-Slave configuration. Through the design process, we will explore the different online services for being implemented following this architecture. We will keep an easy modifiable but efficient architecture in order to be tuned for other environment such as hospitals, orphanages and elderly care homes.

The main architecture is visualized as an embedded system integrated by several modules which interact using their cloud computing capacities. The main modules of the system will be tested and evaluated through experimentation. The use of natural language processing will be considered, however it won't be the center of our investigation.

As one of the main objectives, we will focus our design in the Human Robot Interaction (HRI) as a truly hands-free process. As an extra ability, the interaction and control of an Internet-of-Things (IoT) micro-controller is considered. Having the main advantage, the use of Application Programming Interface (API) scraping for data acquisition, we will present how this data could be retrieved and saved for a future machine learning and statistical analysis, however we would not explore yet these mentioned processes.

Finally, we will present a usability testing design with the vision of being implemented after the experimentation process. Once done, the information retrieved from this usability test would be used as a feedback in order to modify the programming of the modules and thus improve the HRI.

Contents

	Page
Abstract	1
Index of figures	6
Index of tables	9
Acronyms	11
1 Introduction	13
1.1 Motivation	14
1.2 Objectives	15
1.3 Scope	16
2 Background	17
2.1 Service Robots	17
2.1.1 Assistant Robots	18
2.2 Voice assistants	21
2.3 Human-Robot Communication	24
2.3.1 Interaction Model of Communication	25
2.4 Automatic Speech Recognition	27
2.4.1 Types of ASR software	28
2.4.2 Natural Language Processing	28
2.4.3 ASR Technologies	30
2.4.4 ASR Tuning	31
2.5 Online resources and services	32
2.5.1 WakeWord Engine	33
2.5.2 Application Programming Interfaces	33
2.5.3 Cloud Computing	34
2.6 Miscellaneous abilities	34
2.6.1 Data Scraping	35
2.6.2 Internet of Things	35
3 Design Methodology	39
3.1 Delimitation	39

3.2	Preliminary Slave-RasPi Structure	40
3.3	ASR technology analysis	41
3.4	Comparison	44
3.5	Proposed Slave-RasPi Structure	45
3.6	Preliminary Master Architecture	46
4	Implementation	49
4.1	Processing Phase	49
4.2	AVS module	49
4.2.1	Execution	50
4.3	WakeWord module	50
4.3.1	Training	51
4.3.2	Execution	52
4.4	Alexa Skills Kit	53
4.5	Particle Photon	54
4.5.1	Particle API	54
4.6	Master-RasPi Implementation	54
4.6.1	API Scraping	55
5	Evaluation	57
5.1	WakeWord engine	57
5.1.1	False Positives Analysis	58
5.1.2	False Negatives Analysis	58
5.2	Reactive IoT Module	60
5.2.1	ASK and AVS API Setting	61
5.2.2	Results	62
5.3	General Evaluation	63
5.3.1	Mechanical Design	64
5.3.2	3D Printing and Assembly	65
5.3.3	Programming	66
5.4	Future Work	69
6	Usability Testing Design	71
6.1	Presentation	71
6.2	Instructions	72
6.3	Evaluation	73
6.4	Questionnaire	74
7	Costs	75
7.1	Time	75

7.2	Budget Breakdown	76
7.2.1	Material Cost	76
7.2.2	Personal Cost	76
	Environmental impact	79
	Conclusions	81
	Acknowledgement	83
	Bibliography	85

List of Figures

2.1	Assistant robots for personal/domestic use	18
2.2	Zembo Robot	19
2.3	Tapia AI Companion	19
2.4	Pillo	20
2.5	Hospi type R	20
2.6	REEM-C	21
2.7	Care-O-bot 4	22
2.8	Seen benefits of a smart home assistant	23
2.9	Seen disadvantages of a smart home assistant	24
2.10	Schramm's interaction model	25
2.11	Natural Language Processing	29
2.12	Application Programming Interface	33
2.13	Internet of Things devices	36
3.1	Modified Schramm's Model	40
3.2	Robot communication	41
3.3	Proposed Slave-RasPi Structure	46
3.4	Master Scraping	47
4.1	AVS Running	51
4.2	WakeWord Running	53
5.1	Social Networking Samples	60
5.2	Photon Code	61
5.3	AVS API	61
5.4	breadboard	62
5.5	Particle API	62
5.6	Particle API scraping	63
5.7	Mechanical Arms	64
5.8	Palletizing Robot Arm	65
5.9	CAD Design	65
5.10	Arm Comparison	66

7.1 Gantt Diagram 75

List of Tables

3.1	ASR technologies comparison	45
5.1	WakeWord comparison	58
5.2	Correct Triggering in an Average Mall Environment	59
5.3	Correct Triggering in a Night Club Environment	59
7.1	Detailed material cost	76
7.2	Detailed personal cost	77

Acronyms

API Application Programming Interface

ASK Alexa Skills Kit

ASR Automatic Speech Recognition

AVS Alexa Voice Service

CAD Computer-Aided Design

CLIR Cross Language Information Retrieval

CRIS Custom Recognition Intelligent Service

CX Customer Experience

DOF Degree of Freedom

HRI Human-Robot Interaction

IFR International Federation of Robotics

IoT Internet of Things

JSON JavaScript Object Notation

LUIS Language Understanding Intelligent Service

NL Natural Language

NLG Natural Language Generation

NLP Natural Language Processing

NLU Natural Language Understanding

NRI National Robotics Initiative

PLA Poly(lactic acid)

REST Representational State Transfer

RRI Responsible Research and Innovation

ROS Robot Operating System

STT Speech to Text

TTS Text to Speech

UI User Interfaces

UPC Universitat Politècnica de Catalunya

URL Uniform Resource Locator

UX User Experience

VUI Voice User Interfaces

WoZ Wizard of Oz

XML Extensible Markup Language

Chapter 1

Introduction

Nowadays the use of human-robot collaboration has been an important matter in the production and the manufacturing areas. By introducing robots into the same environment as humans, we have to realize that robots won't totally replace humans, but they will augment human capabilities as they work and learn from each other. However, if a human is collaborating with a robot in the same working cell, it is important to have an intuitive way of communication between them.

Robotics in personal and domestic applications has experienced strong global growth with relatively few mass-market products: floor cleaning robots and robots for education and entertainment. Future product visions point to domestic robots of higher sophistication, capability and value, such as assistant robots for supporting the elderly, for helping out with household chores and for entertainment. A strong growing sector is public relation robots which will increase to more than 6,500 robots. These robots are increasingly used in supermarkets, at exhibitions, in museums etc. as guides or information providers [1].

For more than 30 years, the interaction between people and machines has been a rectangle of plastic keys (the keyboard), combined with a plastic ball and two buttons (the mouse), also in recent years a touch screen can be pressed to make words, requests and communicate our needs to the machines. On the other side, voice is the most natural form of communication between humans, the evolution of speech and the language itself has allowed us to form even more intricate communities, understand the mechanics of our surroundings and reach the modern state of our civilization as we know it today. Yet, we have not been totally powered our interactions with robots by voice or speech.

In order to handle this communication problem, speech recognition technologies have been heavily researched in several investigation areas, such as Human-Robot Interaction (HRI). Investigations in Voice User Interfaces (VUIs) have become in the last years one

of the most natural and intuitive methods of HRI. These recent developments in VUIs are demonstrating that voice interfaces may finally be ready for mass adoption.

1.1 Motivation

Up to now, we have been restricted by pushing our control through manual tools and interacting manually through the world, for instance, we literally have been using our hands in each moment we interact with the machines around us. Our motivation, in other hand, born from the pursuit of going into the next level of Human-Robot Interaction (HRI), the level in which we could use our voice in order to communicate and push our wills into our assistant robots. However, we are not interested in a world where our robots do all the work for us, we are more interested in the world where robots would enhance and improve our capabilities through collaboration and also from what we can learn from each other.

This motivation started 18 months ago when a couple of colleagues and I, had in mind a project focused in the Internet of Things (IoT). The idea was simple, small reactive modules embedded in a system that would learn from the user's behaviors. The implementation though, was not as simple as the defined idea, mostly because the limited technology at the time.

Six months later in a new team, we had the idea of a smart voice interface which could not also learn from the user's behavior, but also that would control these small reactive modules through a home environment. We experiment for over 5 months with a new Arduino-type IoT microcontroller and with the accessible Automatic Speech Recognition (ASR) technology that at the time was recently open-sourced by Amazon. Our results were good in the communication phase but still the ASR of Amazon had to be improved with the time, experience and with even more research on the field. Also, as we had a lack of knowledge about all the possible online services we can use, we couldn't start the learning phase since we had issues transferring the acquired data from the cloud services to our processing hardware, and, as a result, this problem became the Achilles' heel of our work. Similarly, as we were focusing our project in a domotics perspective, we had to figure out how to implement a huge wired microphone structure in a way that the system could heard the users' speech all time.

Because the issues exposed above, our project stayed in a stand-by phase until we met one robotic platform that could solve our dilemma, a mobile robot that was part of a social experiment in a cultural center in Barcelona, a multipurpose robotic platform called *MASHI* [2]. At that time *MASHI* was controlled in a Wizard-of-Oz (WoZ) atmosphere,

giving possibilities of teleoperation through WebRTC. Although it was a very impressive job, there was scope in which the platform could be improved.

Then, we realized that implementing what we had done so far in a mobile platform would be a restructure of the main architecture, however having a challenge in which we could use the most of the tools and knowledge we learned during the first 3 semesters of the Master's degree worth the extra effort, without having in count that we also would find the solution for our static-wired issue. Consequently, this challenge became the opportunity of what some people say "*to look at the bigger picture*".

1.2 Objectives

The general objective of this thesis Design and Implement an Automatic Speech Recognition system that will be used as an Interface for Human-Robot Interaction.

In order to have a better perspective of what can be reachable, we defined also specific goals through the process of developing the general objective and subdivide them in three main topics:

ACCESSIBILITY

- Design a general ASR system architecture that after fine-tuning could be used also in other public environments such as Hospitals, Schools and Elderly Care Homes.
- Develop a system that will be cheaper than the available robot assistants in order to have a robotic platform affordable for all the people.
- Develop a user-friendly interface for the public that will be interacting with the platform.

SAFETY

- To offer the necessity of an Off-line mode for the protection of users' information and thus avoiding hackers.
- Design a reliable system that can avoid the more possible faults between the Human-Robot Interaction.

INTERACTION

- Use cloud computing services in order to access to the information from anywhere and improve its scalability.
- Integrate the online services in a coherent mode in order to keep the Natural Language sense.

- Design a Master-Slave structure avoiding the elimination of the capacities of the previous platform.
- Use as many online services as necessary to implement a reliable system but not as many as the system gets slow with bad internet connections.

1.3 Scope

The work to be done during this thesis is based in a series of incremental improvements for the robotic platform *MASHI* in order to its better insertion at the museum environment, assuring its better integration with the previous teleoperated system. The principal work is the incorporation of a structured ASR system into *MASHI*'s structure in a master-slave environment.

The first step is to analyze the different services which can be implemented as a standalone module in a Raspberry Pi and which offer an ASR engine to start with. Next, we will design an architecture which can be used in a museum scenario, this design will be inspired by a one-to-one interaction where the communication process can be enhanced by the robot using an ASR engine as an automatic access of information. Likewise, as the platform is projected to be used for social interaction in other public buildings such as hospitals, schools and nursing homes for elderly people, we will keep an open architecture which can be modified based in the exploratory usability test in order to tune it easily for the other situations.

We will test each module individually in order to check its functionality and reliability. After all the interaction modules have been tested and interconnected following the designed architecture, we will design a usability test in order to have a feedback in following real tests.

It is important to point out that even the individual modules work perfectly this won't guarantee the reliability of the complete system. Also, the real testing of the complete system is not contemplated inside the scope since we have limited knowledge about the social factors that were studied previously [2]. However, it is interesting to contemplate this as a future collaboration with the authors of the cited research in order to observe the variances about the social human-robot interactions with the current semiautomatic guided-robot in the context of a cultural center.

Chapter 2

Background

2.1 Service Robots

Despite the great colonization of industrial robots over the last five decades, it is expected that industrial robotics will be soon surpassed by resulting markets from the so-called service robotics in the coming decades. In fact, there are currently more service robots than industrial ones, and the market value is projected to be increased by more than twice in future years. Service robots deals with robotic applications in, for instance, rehabilitation and health care, logistics, defense, agriculture and forestry, construction, search and rescue, transport, home-care, and education [1].

Several developed countries have already started national and multi-national plans, such as the National Robotics Initiative (NRI) in the US, the Cognitive Systems and Robotics projects in the EU and the International Federation of Robotics (IFR), for supporting basic and applied research that ensures their leadership in the future robotic industry. The potential of the social and economic relevance of these robots is evident.

Nowadays, the companies are faced with increasing labor costs and a shortage of workers, thus they invest in robotics. Robots never request more payment and are able to work 24/7. Robots can perform tasks that most humans could not possibly do, such as working in challenging circumstances and being able to perform amazing task with enormous precision. What makes an organization ready for the challenges of the future is not just technology but the management of Human Resources (HR).

Given the limit of current service robots, a simple natural option emerges to improve, accelerate and facilitate the incorporation of such systems in society at low cost. The standard in robotics is to develop robots with complex skills using a full collection of sensors, devices and programming in order to solve issues faced in the real environment.

2.1.1 Assistant Robots

As a fragment of service robots, Assistant Robots are making task and time management an automated procedure, providing benefits to independent family members, autonomous professionals and multitasking freelancers. Elderly and child caring robots such as robotic pets are also designed to monitor family members at any in-home situation. Service robot suppliers already estimated in 2010 a strong increase of sales of robot assistants. As can be seen in Figure 2.1, it is projected that between 2016 and 2019 about 42 million units of service robots for personal and domestic use to be sold [1].

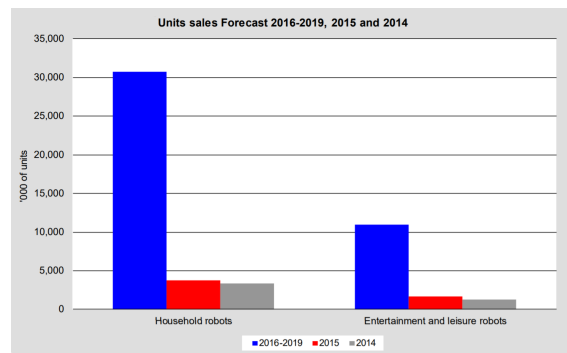


Figure 2.1: Assistant robots for personal/domestic use

However, up till now, there have been no significant sales of humanoids as human companions or social robots, giving all the importance to the typical and the multitasking robotic platforms in order to perform the everyday tasks inside the office or home environments.

Robotic assistants can be found in a number of specialized occupations as well, including haircut-suggesting robots and silicone airport assistants. These technological advancements suggest science fiction subjects may not be far off. Up to next we present the most outstanding and modern assistant robots are the most emblematic service robots that can be found today. Also at the end of this work we will provide a comparative evaluation of the VUIs used by these robotic assistants against the ASR Interface of *MASHI* after the upgrades, improvements and tests.

Zenbo

The robot has an endearing appearance, with a head complete with a touchscreen “face” and a small lower body that allows the robot to roll around. The home interactive robot can be used as a personal assistant that reminds homeowners of their appointments, schedules, medications and also can detect emergencies. When not being used as a screen for a variety of functions including taking pictures, revealing recipes and reading stories and songs to children, the robot has large eyes and dramatic facial expressions.



Figure 2.2: Zembo Robot

Tapia AI Companion

The Tapia AI doesn't have legs or wheels, but is instead egg-shaped and features on a pair of eyes that are capable of expressing emotion and interacting with humans. The Tapia AI robot learns about you and those around you by having conversations and is capable of making calls, ordering products online and let you know about information like the weather. The Tapia AI companion robot can also be set to be with users for designated periods of time. This enables it to be used as a friend for the elderly and let loved ones know if something is wrong.



Figure 2.3: Tapia AI Companion

Pillo

'Pillo' works by securely storing medications and dispensing them at the appropriate time for users to take. This functionality can also be utilized for supplements for those that are looking to increase their overall wellness. Users can ask 'Pillo' questions pertaining to their healthcare, can reorder medications from the pharmacy and even connect patients directly to their doctor. The 'Pillo' health assistants sync with an accompanying smartphone app

and wearable to keep users informed about how to manage their health even when they aren't home.



Figure 2.4: Pillo

Hospi type R

Back in 2010, Panasonic revealed a robot prototype for the medical industry and today the Hospi type R has been cleared to be used in Japan and abroad. Specifically sanctioned for personal care duties, this approachable robot will be able to perform such drudge work as toting around medicine, patients and performing tasks like hair washing. The Hospi type R is being touted by the company as the first robot in the world to meet this particular combination of safety standards. In order to work, it is equipped with a combination of cameras, Wi-Fi and preprogrammed maps to navigate around buildings. It can even take the elevator. Most importantly, the Hospi type R has an ID security card system so unauthorized people can't steal its contents.



Figure 2.5: Hospi type R

REEM-C

REEM-C is a full-size biped humanoid robotics research platform. It's flexible, reliable,



open, standard and up-gradable. An advanced robot to boost research areas like navigation, HRI, vision or AI. REEM-C is a biped humanoid robot as tall as a person and designed for Research and Development. It can be move easily with a crane and challenge several research areas, including walking. Can be used in multiple research areas, like navigation, Vision, HRI, AI, grasping, walking or speech recognition.



Figure 2.6: REEM-C

Care-O-bot 4

Care-O-bot 4 consists of 6 independent and configurable plug and play modules. Scale your Care-O-bot 4 solution according to the requirements of your specific application: from a simple transportation unit, a tele-presence and tele-manipulation system to the full-fledged two-armed gentlemen robot. Up to 31 degrees of freedom with two spherical joints provide an exceptional work-space that is completely covered by sensors. The spherical joints allow for roll-pitch-yaw movements of torso and head and such empower the robot to perform a wide range of body gestures. Care-O-bot 4 offers multi-modal user input: touchscreen, microphones, and speakers. Head cameras enable human robot interaction solutions based on a graphical user interface and gesture recognition. Care-O-bot's state can be signaled back to the user through LEDs, sounds, text-to-speech, laser pointer, and body gestures. The robot is shipped with Open Source drivers and is powered by the Open Source Robot Operating System (ROS). ROS allows Care-O-bot 4 to use a wide range of highly capable service robot components from the field of navigation, manipulation, and perception.

2.2 Voice assistants

HRI challenges AI in many regards: dynamic, partially unknown environments that were not originally designed for robots; a broad variety of situations with rich semantics to understand and interpret; physical interactions with humans that requires fine, low-latency



Figure 2.7: Care-O-bot 4

yet socially acceptable control strategies; natural and multi-modal communication which mandates common-sense knowledge and the representation of possibly divergent mental models. Robots that could understand arbitrary sentences in natural language would be incredibly useful-but real vocabularies are huge. American high school graduates have learned, on average, about 60,000 words [3].

In 1990 Dragon released its first speech-recognition product, Dragon Dictate, which promised to translate speech to text directly to a PC. The first version cost a whopping 9,000 USD. Dragon continuously improved the product over the following decades, but it required a training sequence in order to replicate a user's words accurately. In the early 2000s, voice recognition stalled at around 80% accuracy as companies focused on other technology priorities.

Voice assistant technology, in its latest form, has been around for several years now, with Apple's Siri, Google's Google Now service, and Microsoft's Cortana. And with that breakthrough comes a number of competences for robotic assistants in order to augment our capabilities of communication between humans and machines.

The new technologies of communication between a human and machines have more recently caught fire with the increased adoption of Amazon's Alexa-based devices such as the Amazon Echo and Amazon Dot and Google's Google Home device. Also, advances in AI are allowing them to accurately understand more information, while upgrades to

mobile networks are facilitating quick transfers of data to robust clouds, enabling fast response times. In addition, the swell of internet connected devices like smart thermostats and speakers is giving voice assistants more utility in a connected consumer's life.

In a new report [4], BI Intelligence explains what's driving the recent upsurge in adoption of digital voice assistants. It explores the recent technology advancements that have catalyzed this growth, while presenting the technological shortcomings preventing voice assistants from hitting their true potential.

The study by BI Intelligence comprised a survey of 900 millennials and business leaders in a U.S. panel who make strategic decision within their organizations. A certain number (18%) of people don't see a reason to have a home voice assistant, but a large majority (83%) do. Here is what are viewed as the main benefits of a smart home assistant:

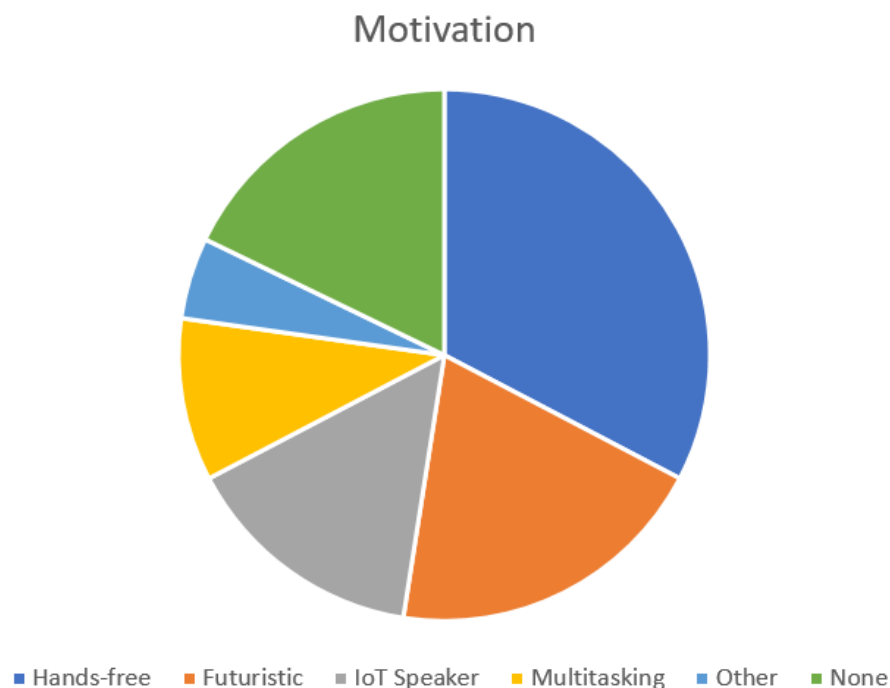


Figure 2.8: Seen benefits of a smart home assistant

However, there are a number of potential pitfalls for voice assistants. For example, many consumers worry about their security. After all, in order to be voice-enabled, today's generation of devices need to be "always listening". Similarly, part of that concern is likely driven by outward security measures, such as a lack of requirement to put in a password or verify identity. In addition, some (19%) consumers see a drawback as an anticipated negative experience due to technology issues. A small number (11%) of consumers don't see any drawbacks, but here's the breakdown of the main perceived drawbacks of a smart home voice assistant:

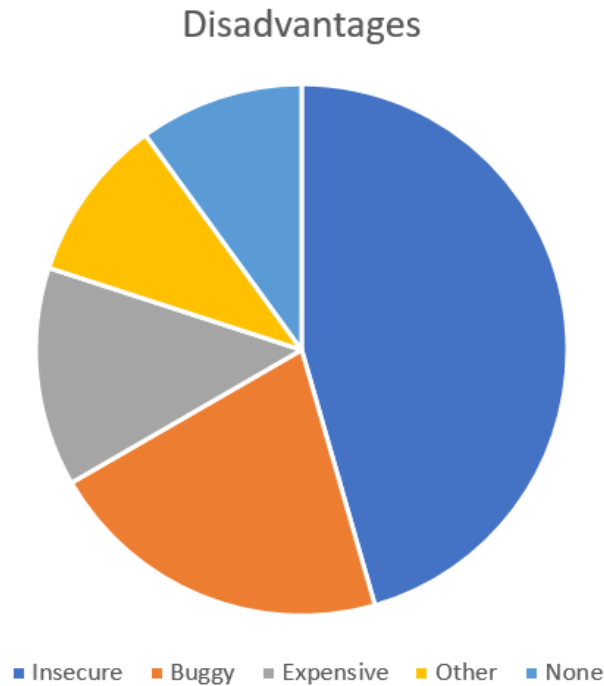


Figure 2.9: Seen disadvantages of a smart home assistant

Voice assistance continues to improve over time, as does the “intelligence” behind them. One of the promises of the Internet of Things (IoT) is the expansion of hands-free interactions. Some of these will be triggered by location sensors and others by gestures or bio-metric methods. However, one of the main causes of interactions will be by voice. The smart home voice assistants are the beginning of that.

2.3 Human-Robot Communication

The traditional conception of conversational robots, as well as earliest systems, is based on a clear human-master robot-servant role assignment, and restricts the robot’s conversational competencies to a simple “motor of command requests” only in most cases.

Notice that here we are not claiming that there is no importance in this research that falls within this strand; we are just mentioning that, as we shall see, there are many other aspects of natural language and robots, which are left unaccounted by such systems. Furthermore, it remains to be seen, how many of these aspects can later be effectively integrated with systems belonging to this strand of research.

2.3.1 Interaction Model of Communication

The Schramm's interaction model of communication describes communication as a process in which participants alternate positions as sender and receiver and generate meaning by sending messages and receiving feedback within physical and psychological contexts [5].

The interaction model is also less message focused and more interaction focused. In the Figure 2.10 we can see that rather than illustrating communication as a linear one-way process, the Schramm's interaction model incorporates feedback, which makes communication a more interactive two-way process. Feedback includes messages sent in response to other messages.

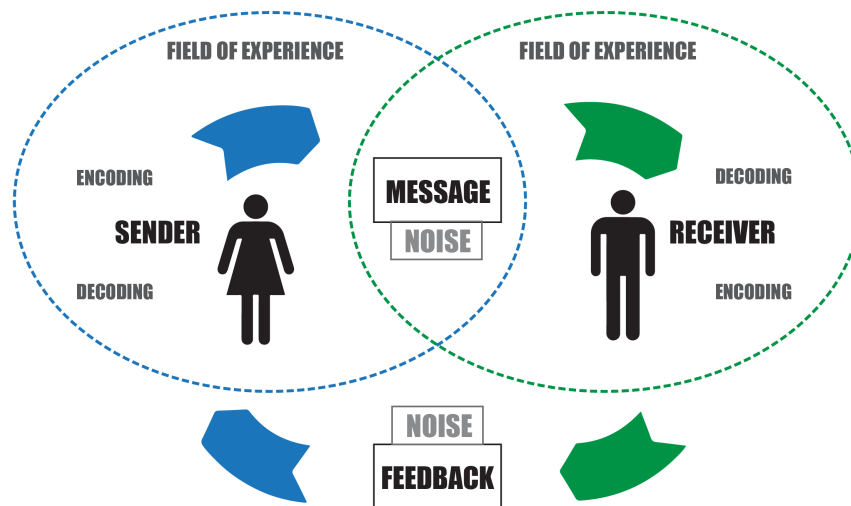


Figure 2.10: Schramm's interaction model

The model suggests that encoding and decoding are the two most important part of a communication process. Encoding assumes a critical part in starting the procedure of correspondence by converting data into information. Encoding is done by a sender or transmitter and sent to a receiver. When data reaches to the receiver, receiver decodes and interprets the data. This data is called a message, and it is transmitted through a medium.

The inclusion of a feedback loop also leads to a more complex understanding of the roles of participants in a communication encounter. Rather than having one sender, one message, and one receiver, this model has two sender-receivers who exchange messages. The interpreted data is known as information. This makes communication effective but might cause problems too as the message sent after encoding might not be the same when decoded by the receiver. Feedback is also a very important component as it lets the sender

know if the receiver has interpreted the message as required or not. The message becomes useless if the receiver does not understand it making feedback different than the expected outcome, then the communication is incomplete if there is no feedback.

The interaction model is more concerned with the communication process itself. In fact, this model acknowledges that there are so many messages being sent at one time that many of them may not even be received. Some messages are also unintentionally sent; therefore, communication is not judged effective or ineffective in this model based on whether or not a single message was successfully transmitted and received.

Based in this perception of Human Robot Interaction, when the receiver is switched for a state-of-the-art robot like the one revised in the point 2.1.1, or the robotic museum guides MINERVA [6], ROBOVIE [7], REEM [8] and the previous version of *MASHI* [2] several points of this communication model get more difficult to carry and thus the process itself involves a number of limitations:

- The receiver (now a robot) only accepts a fixed and small number of simple prerecorded commands, and it responds with a set of prerecorded answers.
- In the philosophical way [9], the only speech act that the receiver can handle are requests and feedback.
- The dialogue that is supported is visibly not flexible in the way of a mixed initiative, this means that the participants do not alternate roles as sender and receiver in order to keep a communication encounter going because the communication process is just originated by the sender.
- The receiver is not aware about the physical situations and events that are happening around both the sender and the receiver, except for the fixed number of pre-programmed variables.
- The robot is not able to handle affective speech, rather than generate the emotion-based payload, this payload it is just understood and interpreted by the sender during the feedback phase.
- The non-verbal communication capabilities are limited [10], gestures, facial expressions, and head signals are again just recognized by the sender.
- No real speech planning or focused dialogue generation is taking place, and certainly not in combination with the motor other subsystems of the robot.
- There is no real OFFLine or ONLine learning taking place, verbal and motion behaviors have to be pre-programed.

Taking these limitations, it gradually becomes more seeming that the connection between local environment and technical semantics of a speech is quite crucial. Therefore, when dealing with robots and language, it is quite impossible to separate the linguistic subsystems from awareness and action, and just plug-and-play with a simple speech-in speech-out black box chatterbot of some sort.

2.4 Automatic Speech Recognition

In terms of technological development, we may still be at least a couple of decades away from having truly autonomous, intelligent AI systems communicating with us in a genuinely “human-like” way. However, in many ways, we’re progressing steadily towards this future scenario at a surprisingly fast pace thanks to the continuing development of what is known as automated speech recognition technology. And at least so far, it’s looking to promise some truly useful innovations in user experience for all sorts of applications.

Automatic Speech Recognition (ASR, is the technology that allows human beings to use their voices to speak with a computer interface in a way that, in its most sophisticated variations, resembles normal human conversation. The most advanced version of currently developed ASR technologies revolves around what is called Natural Language Processing (NLP). This variant of ASR comes the closest to allowing real conversation between people and machine intelligence and though it still has a long way to go before reaching an apex of development, we’re already seeing some remarkable results in the form of intelligent smart phone interfaces like the Siri program on the iPhone and other systems used in business and advanced technology contexts. However, even these NLP programs, despite and “accuracy” of roughly 96 to 99% can only achieve these kinds of results under ideal conditions in which the questions directed at them by humans are of a simple yes or no type or have only a limited number of possible response options based on selected keywords.

The basic sequence of events that makes any ASR software, regardless of its sophistication, pick up and break down your words for analysis and response goes as follows:

- You speak to the software via an audio feed
- The device you’re speaking to creates a wave file of your words
- The wave file is cleaned by removing background noise and normalizing volume
- The resulting filtered wave form is then broken down into what are called phonemes. (Phonemes are the basic building block sounds of language and words. English has 44 of them, consisting of sound blocks such as “wh”, “t”, “ka” and “t”.

- Each phoneme is like a chain link and by analyzing them in sequence, starting from the first phoneme, the ASR software uses statistical probability analysis to deduce whole words and then from there, complete sentences
- Your ASR, now having “understood” your words, can respond to you in a meaningful way.

2.4.1 Types of ASR software

The two main types of ASR software variants are Directed Dialogue conversations and Natural Language Conversations (with NLP):

DIRECTED DIALOGUE

Directed Dialogue conversations are the much simpler version of ASR at work and consist of machine interfaces that tell you verbally to respond with a specific word from a limited list of choices, thus forming their response to your narrowly defined request. Automated telephone banking and other customer service interfaces commonly use directed dialogue ASR software.

NATURAL LANGUAGE CONVERSATIONS

Natural Language Conversations are the much more sophisticated variants of ASR and instead of heavily limited menus of words you may use, they try to simulate real conversation by allowing you to use an open-ended chat format with them, the Siri’s interface on the iPhone is a highly advanced example of these systems.

2.4.2 Natural Language Processing

NLP is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things. NLP researchers aim to gather knowledge on how human beings understand and use language so that appropriate tools and techniques can be developed to make computer systems understand and manipulate natural languages to perform the desired tasks [11].

The foundations of NLP lie in a number of disciplines, computer and information sciences, linguistics, mathematics, electrical and electronic engineering, AI and robotics, psychology, etc. Applications of NLP include a vast number of fields and studies, such as machine translation, natural language text processing and summarizing, user interfaces, multilingual and Cross Language Information Retrieval (CLIR), speech recognition, AI and expert systems, and so on.

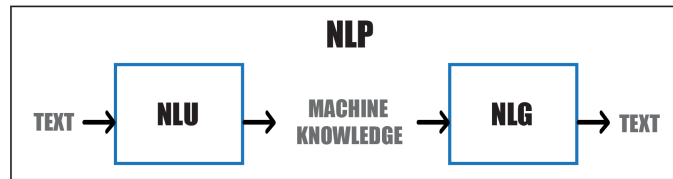


Figure 2.11: Natural Language Processing

NATURAL LANGUAGE UNDERSTANDING

As a subdivision of NLP the is the Natural Language Understanding (NLU) is the process of disassembling and parsing inputs and is more complex than the reverse process of assembling output in Natural Language Generation (NLG) because of the occurrence of unknown and unexpected features in the input and the need to determine the appropriate syntactic and semantic schemes to apply to it, factors which are pre-determined when outputting language.

The process of building computer programs that understand natural language involves three major problems: the first one relates to the thought process, the second one to the representation and meaning of the linguistic input, and the third one to the world knowledge. Thus, an NLP system may begin at the word level to determine the morphological structure, nature (such as part-of speech, meaning) etc. of the word and then may move on to the sentence level to determine the word order, grammar, meaning of the entire sentence, etc. and then to the context and the overall environment or domain. A given word or a sentence may have a specific meaning or connotation in a given context or domain, and may be related to many other words and/or sentences in the given context.

NATURAL LANGUAGE GENERATION

The second part that constitute the NLP is the Natural Language Generation. This subdivision is concerned with the construction of computer systems that can produce understandable texts in English or other human languages from some underlying non-linguistic representation of information [12]. Natural language generation systems combine knowledge about language and the application domain to automatically produce documents, reports, explanations, help messages, and other kinds of texts.

The most common use of natural language generation technology is to create computer systems that present information to people in a representation that they find easy to comprehend. Internally, computer systems use representations which are straightforward for them to manipulate, such as airline schedule databases, accounting spreadsheets, expert system knowledge bases, grid-based simulations of physical systems, and so forth. In

many cases, however, these representations of information require a considerable amount of expertise to interpret. This means that there is often a need for systems which can present such data in an understandable form to non-expert users.

2.4.3 ASR Technologies

ASR can be found as software programs that respond to voice commands in order to perform a range of tasks. They can find an opening in a consumer's calendar to schedule an appointment, place an online order for tangible goods, and act as a hands-free facilitator for texting, among many, many other tasks. Up to now, the most relevant ASR technologies are:

PocketSphinx

PocketSphinx, is one of Carnegie Mellon University's open source large vocabulary, speaker-independent continuous speech recognition engine. High speed of indexation, flexible search capabilities, integration with the most popular data base management systems and the support of various programming language APIs (e.g. for PHP, Python, Java, Perl, Ruby, .NET etc) all that make the search engine popular with thousands of e-Commerce developers and merchants.

Project Oxford

Microsoft's Language Understanding Intelligence Service (LUIS) is a component of the Microsoft Cognitive Services focused on creating and processing natural language models. LUIS provides a sophisticated tool-set that allow developers to train the platform in new conversation models. LUIS can also be used in conjunction with other text processing APIs in MCS such as linguistic analysis and text analytic. The platform provides a deep integration with Microsoft's Bot Framework and can be used by other bot platforms.

Alexa Voice Service

Alexa Voice Service (AVS) is Amazon's intelligent voice recognition and natural language understanding service that allows you as a developer to voice-enable any connected device that has a microphone and speaker. AVS will drive \$10 billion of revenue to Amazon by 2020 not to mention the artificial intelligence-based system currently owns 70 percent of the voice market. It comes with Alexa Skills Kit ASK also supports more sophisticated multi-command dialogues and parameter passing.

Google Natural Language API

Google Natural Language (NL) API is a recent addition to Google Cloud focused on NLP and NLU capabilities. The NL API enables capabilities such as intent-entity detection,



sentiment analysis, content classification and relationship graphs. Google NL platform is actively used by several high-profile services such as Google's Assistant.

Wit.ai

Wit.ai is the platform behind the NLP-NLU capabilities of Facebook Messenger platform. Facebook acquired Wit.ai in January 2015 and, since then, has rolled out major updates to the platform. One of the best capabilities of Wit.ai is the sophisticated tool-set that can be used to train the platform in new conversation models as well as monitoring the interactions between users and the platform.

Api.ai

The Api.ai platform lets developers seamlessly integrate intelligent voice command systems into their products to create consumer-friendly voice-enabled user interfaces. Functionally, Api.ai includes capabilities such as speech recognition, fulfillment and NLU as well as a robust management tool-set. Api.ai provides integration with several bot platforms and is particularly popular within the Slack community.

Sensory

The firm, which has been developing computer-driven voice technologies like speech recognition, speech and music synthesis, and speaker verification since the late '90s, has its technology embedded in more than a billion devices around the world. Sensory's solution, TrulyHandsFree, takes the form of an "assortment" of speech synthesis models tailor-made for a spectrum of electronics. The voice engines, of which there are more than 20, range from no-frills, single-word models to powerful algorithms capable of deciphering speech in "highly noise robust" environments. One, an ultra-low-power model designed for use in smart-phones, smart-watches, and other battery-operated mobile devices which lack a dedicated power source, requires "less than one milliamperere" of power under load.

2.4.4 ASR Tuning

The training of ASR systems, works on two ways. The first and simpler of these is called "Tuning" and the second and also much more advanced variant is called "Active Learning".

DIRECT TUNING

This is a relatively simple means of performing ASR training. It involves human programmers going through the conversation logs of a given ASR software interface and looking at the commonly used words that it had to hear but which it does not have in its pre-programmed vocabulary. Those words are then added to the software so that it can

expand its comprehension of speech.

ACTIVE LEARNING

Active learning is the much more sophisticated variant of ASR and is particularly being tried with NLP versions of speech recognition technology. With active learning, the software itself is programmed to autonomously learn, retain and adopt new words, thus constantly expanding its vocabulary as it's exposed to new ways of speaking and saying things. This, allows the software to pick up on the more specific speech habits of particular users so that it can communicate better with them. So, if a given human user keeps negating the auto-correct on a specific word, the NLP software eventually learns to recognize that particular person's different use of that word as the "correct" version.

2.5 Online resources and services

Another interesting element towards enhanced communication between human and robots is that nowadays more and more robots can be constantly connected to the internet, now all data and programs that the robot uses need to be onboard its hardware. Consequently, a robot could potentially utilize online information as well as online services, in order to enhance its communication capacities. Thus, the intelligence of the robot is partially offloaded to the internet; and potentially, thousands of programs and/or humans could be providing part of its intelligence, even in real-time. For example, going much beyond traditional cloud robotics [13], in the human-robot cloud proposal [14], one could construct on-demand and on-the-fly distributed robots with human and machine sensing, actuation, and processing components.

There exists also a European project called RoboEarth [15], which is described as the World Wide Web for robots: a giant network and database repository where machines can share information and learn from each other about their behavior and their location. Bringing a new meaning to the phrase, the goal of RoboEarth is to allow robotic systems to benefit from the experience of other robots, paving the way for rapid advances in machine cognition and behavior, and ultimately, for more subtle and sophisticated human-machine interaction.

As a big part of human knowledge, information, as well as real-world communication is taking place either through writing or through such electronic channels, inevitably more and more systems in the future will have corresponding abilities. Thus, robots will be able to more fluidly integrate within human civilizations and environments, and ideally will be enabled to utilize the services offered within such systems for humans. More important, robots might also one day become able to support, maintain and improve the physical

human-robot social networks.

2.5.1 WakeWord Engine

A WakeWord is fundamentally an ASR that only reacts in presence of one word or command, this ASR works with trained voice models which takes a recording way sound of the command itself and regenerates the same command with differences in volume, gravity of the voice etc, in order to have more samples in which to learn. This service is highly customizable allowing to freely define a hotword to trigger any other process we make. Is always listening but protects the privacy since it does not connect the Internet or stream voice anywhere. Lastly, the WakeWord service is light-weight and embedded allowing it to runs it on Raspberry Pi's consuming less than 10% CPU on the smallest Pi's.

2.5.2 Application Programming Interfaces

Web services are a widely accepted solution to provide a single programming interface to multiple languages. Web services are purpose-built web servers that support the needs of a side or any other application. Client programs use Application Programming Interfaces (APIs) to communicate with web services. Generally speaking, an API exposes a set of data and functions to facilitate interactions between computer programs and allow them to exchange information. As is shown in Figure 2.12, a Web API is the face of a web service, directly listening and responding to client requests.



Figure 2.12: Application Programming Interface

Since its introduction in 2000 [16], the Representational State Transfer (REST) architectural style has been more commonly applied to the design of APIs for modern web services. SOAP is also a popular technology used for it [17], but is burdened with significant setup and processing overhead for the client. REST in contrast, encourages the reuse of HTTP technology to send and receive data in the same way a Web browser requests and receives a Web page via Uniform Resource Locators (URLs). Finally, REST architecture does not impose format restrictions on the returned data.

A Web API conforming for the REST architectural style is a REST API. Having a REST API makes a web service “RESTful”. A REST API consist of an assembly of in-

terlinked resources. This set of resources is known as the REST API's resource model.

Well-designed REST APIs can attract client developers to use web services. In today's open market where rival web services are competing for attention, an aesthetically pleasing REST API design is a must-have feature.

A number of native third-party APIs have been developed to help access the REST API in languages such as Python, R and JavaScript, which demonstrates the usefulness of our REST API to these increasingly popular informatics languages. Similarly, REST has shown itself to be a sustainable model for the distribution of data to multiple programming languages.

2.5.3 Cloud Computing

Where in the past, people would run applications or programs from software downloaded on a physical computer or server in their building, cloud computing allows people access to the same kinds of applications through the internet.

In the simplest terms, cloud computing means storing and accessing data and programs over the Internet instead of your computer's hard drive. The cloud is just a metaphor for the Internet. It goes back to the days of flowcharts and presentations that would represent the gigantic server-farm infrastructure of the Internet as nothing but a puffy, white cumulus cloud, accepting connections and doling out information as it floats.

What cloud computing is not about is your hard drive. When you store data on or run programs from the hard drive, that's called local storage and computing. Everything you need is physically close to you, which means accessing your data is fast and easy, for that one computer, or others on the local network.

For it to be considered "cloud computing", you need to access your data or your programs over the Internet, or at the very least, have that data synced with other information over the Web. The end result is the same: with an online connection, cloud computing can be done anywhere, anytime.

2.6 Miscellaneous abilities

Beyond the desired information examined so far, there exist a number of other abilities that may be not required for a simple assistant robot but would be imperative towards a complete and fluid ASR Interface regarding of data acquisition.



2.6.1 Data Scraping

Data scraping is a technology solution to extract data from web sites or APIs, in a quick, efficient and automated manner, offering data in a more structured and easier to use format [18].

Data collection in a handcrafted way is truly inefficient: search, copy and paste data in a spreadsheet to later process them. This is a tedious, annoying and tiresome process. Therefore, it makes much more sense to automate this process. Scraping allows this kind of automation, as the majority of the available tools provide an API to facilitate access to the content generated.

Normally, data transfer between programs is accomplished using data structures suited for automated processing by computers, not people. Such interchange formats and protocols are typically rigidly structured, well-documented, easily parsed, and keep ambiguity to a minimum. Very often, these transmissions are not human-readable at all.

Thus, the key element that distinguishes data scraping from regular parsing is that the output being scraped was intended for display to an end-user, rather than as input to another program, and is therefore usually neither documented nor structured for convenient parsing. Data scraping often involves ignoring binary data (usually images or multimedia data), display formatting, redundant labels, superfluous commentary, and other information which is either irrelevant or hinders automated processing.

Data scraping is most often done either to interface to a legacy system which has no other mechanism which is compatible with current hardware, or to interface to a third-party system which does not provide a more convenient API. In the second case, the operator of the third-party system will often see screen scraping as unwanted, due to reasons such as increased system load, the loss of advertisement revenue, or the loss of control of the information content.

2.6.2 Internet of Things

Previously, connected devices such as vending machines needed mains charging and relied on Wi-Fi, cellular or Bluetooth connectivity. Now our connectivity requirements have expanded. We want connectivity in less accessible places: factories in remote locations, rural fields or construction sites, potentially everywhere.

We now have power-efficient sensors and a multitude of different ways of connecting devices to cloud-based analytics, using low-power wide-area networks, 4G cellular and

of course the super-efficient new 5G cellular standard, which is being designed from the ground up to enable the IoT.

As we can see in Figure 2.13, by 2020 more than 50 billion things, ranging from cranes to coffee machines, will be connected to the internet. That means a lot of data will be created, in fact, to be manageable or to be kept forever affordably. Gateways can help; they not only dispatch traffic but carry out some analytics functions, so that data can be better managed. For example, they could be used to filter out ‘normal’ data over time and to look for unusual patterns which may indicate a problem. They can also improve the costs of the transmission and storage of all that data. In next-generation network technology, these gateways will be used dynamically as part of the network where and when needed.

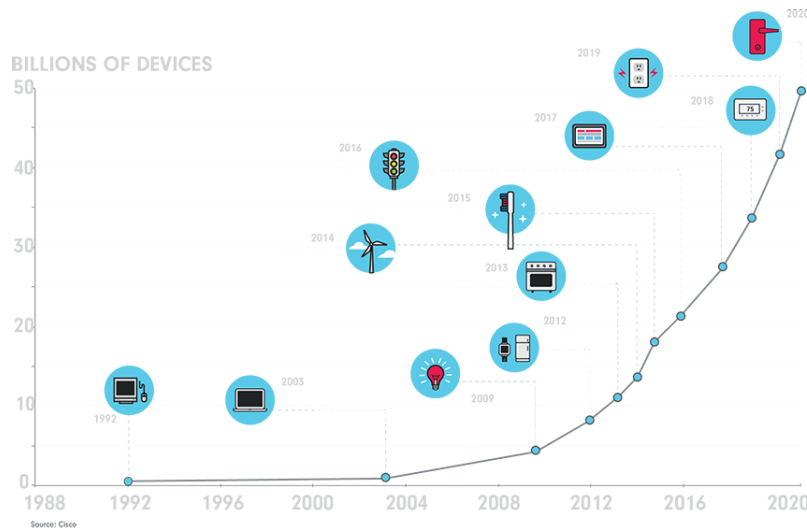


Figure 2.13: Internet of Things devices

But this brave new world is not without its challenges. One by-product of more devices creating more data is that they are speaking lots of different programming languages. Machines are still using languages from the 1970s and 80s as well as the new languages of today. In short, applications need to have data translated for them before they can make sense of the information.

Then there are analytic and data storage. These can be done in-site, regionally or centrally, with different network constraints. Different organizations might want to host applications in their factory, at a regionally aggregated level or centrally, but not all those locations might be convenient in terms of access to secure data centers, capacity or reliable network access. And of course, security becomes even more important as there is little

human interaction in the flow of data from device to data center, the so-called machine to machine communication.

Chapter 3

Design Methodology

As we declared at the specific objectives we will based our design in a Master-Slave architecture, thus avoiding modifications and/or subtractions of the tele-operation capabilities achieved by the previous platform. However, we would need to do a preliminary analysis of what we want to achieve and how.

The Master-Slave architecture of the system will be implemented between the previous Raspberry Pi 3 (Master-RasPi) and a new Raspberry Pi 3 (Slave-RasPi) in which we will implement the structure that will be proposed at the end of this section.

3.1 Delimitation

Articulating multiple independent software modules in one coherent robotic architecture is not only a technical challenge but also an entire theoretical challenge. Introducing such tools into a public space offers new audiences participatory opportunities while also adding new social interaction questions, as demonstrated recently by the previous work with the *MASHI* platform [2]. Nevertheless, and learning from the mentioned work, these are some potential usages of an assistant robot with a reliable ASR Interface in a museum environment:

- An internal tool to help museum staff more quickly learn and find information about an artwork, including information that is not available to the public.
- Alternative to a browser based tool and browser search.
- An external tool for exhibition and art information.
- A public-facing tool in the museum placed in common areas such as a museum's charging stations or cafes to be used to serve general information, and also to help us gather information and learn about visitor interests.

- A public-facing in-gallery assistant to provide information about very specific exhibitions, artists, and artworks.
- A public-facing alternative tool to support in accessibility options.

3.2 Preliminary Slave-RasPi Structure

We will based our design starting from Schramm’s Communication Model, from this model the receiver is switched from a human to a robot as is represented in the Figure 3.1, subsequently we focused our attention in what the model suggest are the two most important phases of a communication process: encoding and decoding.

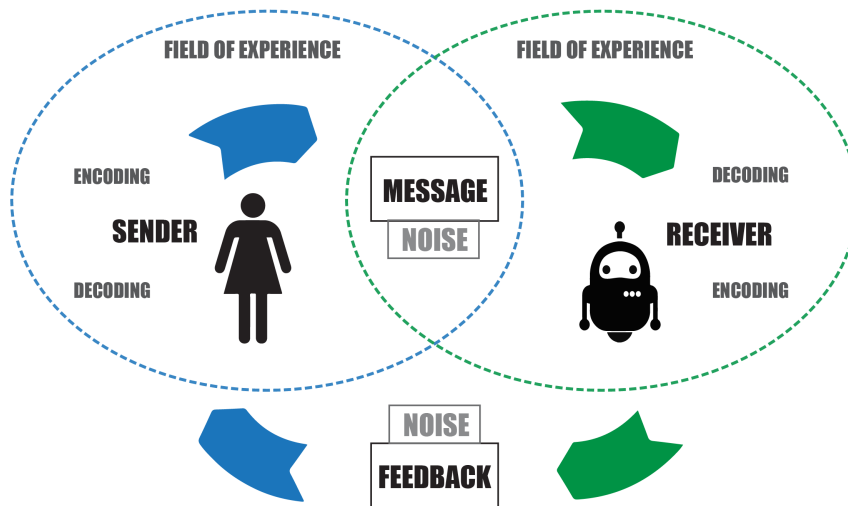


Figure 3.1: Modified Schramm’s Model

First of all, we established the modules that could solve the encoding and decoding paradigm of the Human-Robot communication (see Figure 3.2).

The decoding phase can be separate in three main processes; the speech is received by the robot as an audio input, the Speech to Text (STT) module transform the audio in to text data and then the NLU interprets the representation and meaning of the linguistic text input. From this point, the data can be analyzed for the machine extracting the intent, which is composed of and object and attributes. Afterwards, with this intent the processor can search the desired data or to execute the desired actions (if these actions are pre-programed as skills).

The encoding phase would be basically the opposite; once the data has been extracted from the source, the NLG module translate the information to text data that can be

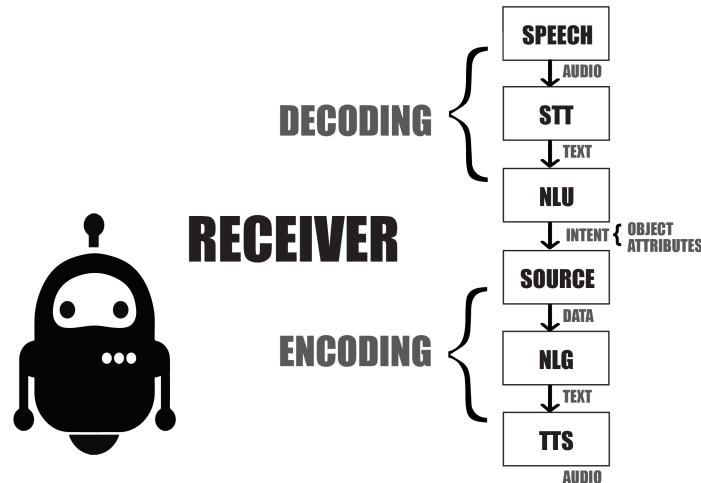


Figure 3.2: Robot communication

understand by the human, and finally (in a VUI’s perspective) this data is received by the Text to Speech (TTS) module and is sent it to the Human as an audio feedback.

3.3 ASR technology analysis

As it was mention in section 2.2, NLP is a much more advanced field of computer science than speech recognition, NLP is concerned with understanding of the real meaning of the user’s speech. Technological advances are making ASRs more capable, these improvements fall into two categories: improvements in AI, specifically NLP and machine learning. Also gains in computing and telecommunications infrastructure, like more powerful computers, better Wi-Fi networks, and faster cloud computing.

The“best” ASRs working with a NLP perspective are not open-sourced, meaning that not only is practically impossible to modify their behavior or access to certain kind of data they work with, but also are limited about the devices that can control and only can be used focusing in the products or services in which they have partnerships.

In order to implement the considered Slave-RasPi structure of section 3.2, we will first analyze the available State-of-the-Art ASR technologies, to afterwards develop a stable and reliable version of the modules proposed.

POCKETSPHINX

Advantages:

- It is possible to set a custom Hotword.

- It can work offline, thus there is a lower battery consumption.
- It can be used as an offline solution only for Hotword recognition and later, we could require some ASR service to handle the requests.

Disadvantages:

- PocketSphinx is not accurate enough to get the results we want to achieve.
- It reacts not only to the “*MASHI*” hotword, giving a lot of false positives.
- There is a pause after Sphinx recognizes a keyword and launches the cloud service.

PROJECT OXFORD

Advantages:

- Powerful in combination with Language Understanding
- Intelligent Service (LUIS) and Custom Recognition
- Intelligent Service (CRIS).

Disadvantages:

- There is only possible to implementing the beta version of LUIS and CRIS.
- It may be not stable enough due to is a beta version.
- It is needed to implement intents manually.

ALEXA VOICE SERVICE

Advantages:

- Alexa provides a set of built-in skills and capabilities available for use freely.
- It can answer general knowledge questions, get the current time, provide weather forecast information and query Wikipedia.
- It returns direct voice speech as an answer.

Disadvantages:

- To get custom intents within AVS it is necessary to create, register and test them with the ASK.
- It has a complicated documentation.

- To capture the user's utterances, the device needs to have a button to activate the microphone (push-to-talk).
- Far-field voice recognition or using a custom Hotword to trigger activation of the AVS is currently unavailable.

GOOGLE NATURAL LANGUAGE API

Advantages:

- Includes sophisticated tooling for training and authoring new NL models.

Disadvantages:

- With this solution, we are not able to change the "OK, Google" wake word.
- Only launch the application in order to handle intents.

WIT.AI

Advantages:

- It can return JSON as an answer.
- It has already a large number of built-in intents.
- It has the ability to learn from the user.

Disadvantages:

- It is not stable enough.
- It is not so easy to use.

API.AI

Advantages:

- Test application using Api.ai is closest in quality to Alexa.
- It is really easy to implement, it has clear documentation
- It has the ability to learn and adapt using machine learning.
- It has its own complete cloud solution.
- Such as AVS it returns a voice as an answer.
- It is user friendly.

Disadvantages:

- It has only a push to talk input, there is no Hotword option enabled.

SENSORY

Advantages:

- It has an always-listening mode.
- It comes with high accuracy; this solution can respond to commands given from a little bit more than 6 meters away and/or in high noise conditions.
- It has the ability to include pre-built commands. This is a great feature, we could set predefined commands and there's no need for an Internet connection to handle them.

Disadvantages:

- It is a closed solution, if we want to set a custom keyword like “*MASHI*”, we needed to contact Sensory client support.
- It does not include a natural language processing engine and requires other services to handle complicated requests.
- The worst of all, Sensory is Not free, however we could have a free trial of 600 samples.

3.4 Comparison

In order to design a fully effective and feasible Slave-RasPi structure, we will set a comparison between the ASR technologies and then propose an overall architecture in which the most of the requirements focused in the specific objectives of section 1.2 can be achieve, also we will consider the solution of the disadvantages that people perceive about voice assistants shown in section 2.2 and finally the potential usages expressed in the section 3.1

As we can see in Table 3.1, is clearly that the AVS is the best ASR technology overall. However, it still fails into fulfill all the requirements as it has a lack of a custom word and fall short of an off-line mode available.

	Sphinx	Oxford	AVS	Google	WIT	API	Sensory
Custom Hotword	YES	NO	NO	NO	NO	NO	YES
Off-line Mode	YES	NO	NO	NO	NO	NO	YES
Cloud API	NO	NO	YES	NO	NO	YES	NO
NLP	NO	NO	YES	NO	NO	NO	NO
Reliable	NO	NO	YES	NO	NO	NO	YES
IoT Control	NO	NO	YES	YES	NO	NO	YES
Hands-Free	YES	NO	YES	NO	NO	NO	YES
Built-In Skills	NO	NO	YES	NO	YES	NO	YES
Browser Search	NO	NO	YES	YES	NO	NO	NO
Interaction Saving	NO	NO	YES	NO	NO	YES	NO
Direct Tuning	YES	NO	YES	YES	NO	NO	YES
Active Learning	NO	NO	YES	YES	YES	NO	YES
Free of Charge	YES	YES	YES	YES	YES	YES	NO

Table 3.1: ASR technologies comparison

3.5 Proposed Slave-RasPi Structure

After analyzing the advantages and disadvantages of all the ASR technologies, the comparison between them to fulfill the most of the requirements and the Master-Slave architecture we proposed, we conclude that there is not a straight forward answer for our dilemma. After researched possible ways to improve the ASR technologies we found out that a full ASR can be run in order to perform the Hotword detection. In this scenario, the device would watch for specific trigger words in the ASR transcriptions. However, an ASR consumes a lot of device and bandwidth resources. In addition, it does not protect your privacy when one uses a cloud-based solution.

Because of that we had to continue looking for a better solution until we found a new online service called WakeWord engine, this service was provided for one of the ASR technologies analyzed, yet, it was not free-of-charge. Then analyzing the concept, itself we found another service called Snowboy, a highly customizable WakeWord detection engine that is embedded real-time and is always listening and also had the option of an off-line mode.

With this perspective, we could implement this WakeWord engine as the trigger of the AVS in order to fulfill the requirements explained in the comparison of section 3.4. With this in mind and using the preliminary structure we defined in section 3.2, our proposed Slave-RasPi structure to implement can be seen in Figure 3.3.

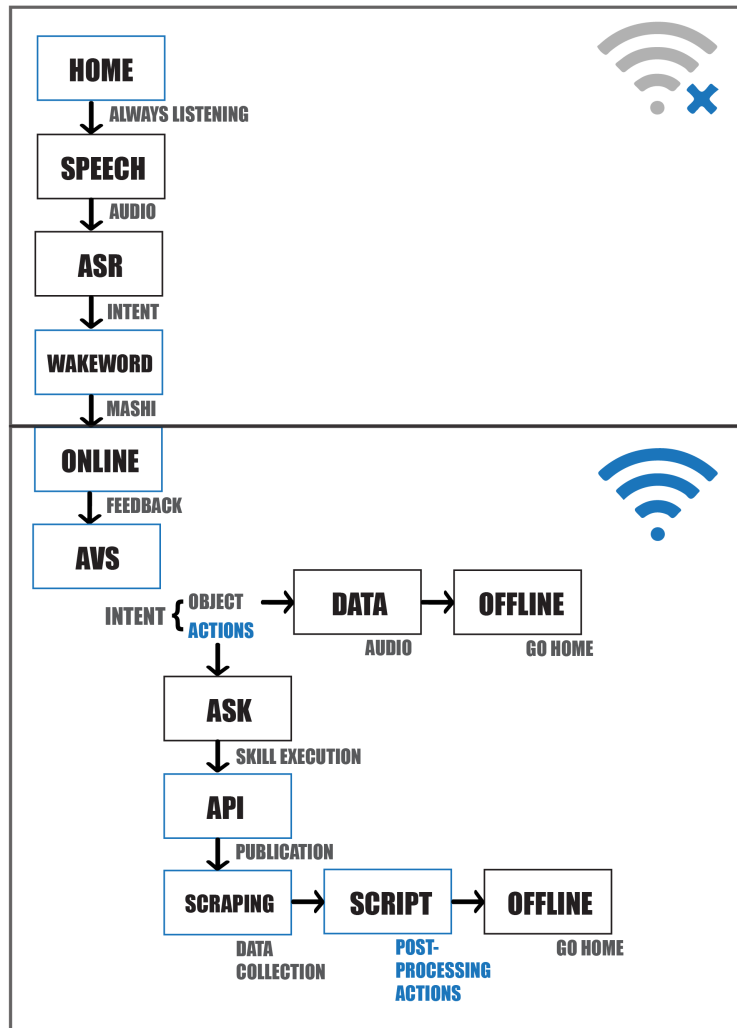


Figure 3.3: Proposed Slave-RasPi Structure

3.6 Preliminary Master Architecture

The proposed architecture is designed as the communication between the Master-RasPi and the Slave-RasPi. This communication is possible due to the published data made by the Slave-RasPi at the AVS API and the Particle API.

Every Particle solution, such as the Photon, has a RESTful API which we can scrape for the information we want to access and post process this data for more complex actions and/or saved it for further analysis.

Also, since AVS saves all queries that have been used, this could be useful information which can help us providing better messages to visitors based on what is being asked or just save this queries as data acquisition.

All the mentioned data can be acquired by the Master-RasPi in a scraping process through a python script (Figure 3.4). In other sense, by scraping, the platform can access to the readings, variables and settings of the reactive modules and to the information of the previous voice HRI's. Also, this python script can be programmed with a series of post-processing actions concerning to the platform itself. It is important to observe that for the scraping process, an internet connection is needed in order to access to the Particle and AVS APIs.

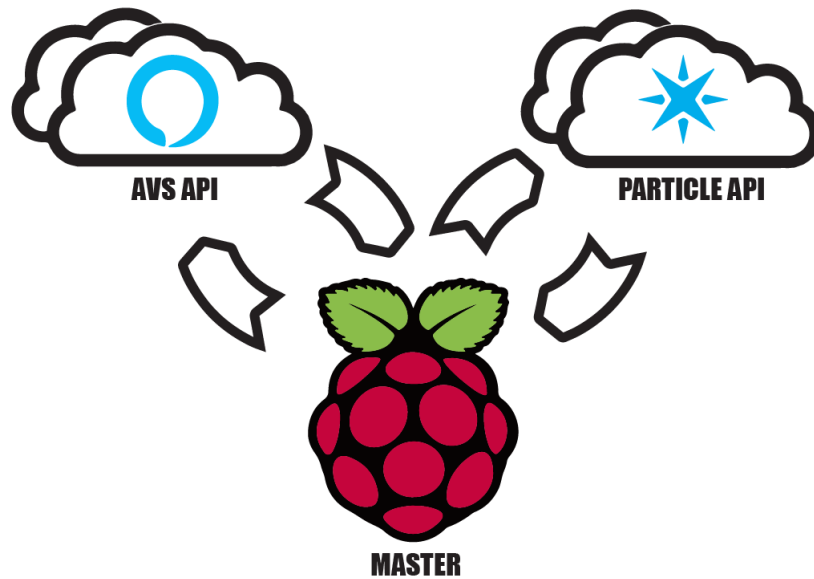


Figure 3.4: Master Scraping

Finally, our hypothesis is that the architecture could be extended with an exploratory usability test and afterwards fine tuning using the information retrieved by the mentioned test, thus the system could fully satisfy more sectors in which we want to put the robot *MASHI*.

Chapter 4

Implementation

In this section, we explore the process of implementing the designed Master-Slave architecture in which the Master-RasPi will be still employing the Wizard-of-Oz setup in the WebRTC environment. This previous work is still necessary due to the limitation of our system of managing a characterization size different than single, or in a different context of a one-to-one communication. Likewise, as the Master-RasPi still has the navigation control, the human-robot collaboration is crucial for the correct functionality of the platform.

4.1 Processing Phase

The processing phase of the Slave-RasPi is implemented using 3 linked online services running in 3 different terminals all the time in our Raspberry Pi. These services are:

- Companion Service: is used to authorize the sample app AVS.
- AVS Java Client: it runs the sample app to communicate with the AVS cloud.
- WakeWord module: is the hotword engine, which switches the ASR model by our model that we trained with Snowboy.

4.2 AVS module

In order to use the AVS we have first to create an account at developer.amazon.com, this in order to use the capabilities of this service for free as a developer. Then we create a device (In our case ASR_Interface) and a security profile, this will give us the next information that we will use for the further process:

- ProductID (also known as Device Type ID)
- ClientID

- ClientSecret

Next, we will open a new terminal in our Master-RasPi and run the next commands:

```
cd Desktop
git clone https://github.com/alexa/alexa-avs-sample-app.git
```

Then we will update the intaller with our information we obtained before as:

```
cd ~/Desktop/alexa-avs-sample-app
nano automated\_install.sh
```

Finally, we wil run the script installer with the next commands:

```
cd ~/Desktop/alexa-avs-sample-app
. automated\_install.sh
```

4.2.1 Execution

When the installation is finished we will type the following commands to bring up the Companion Service which is used to authorize the use of AVS:

```
cd ~/Desktop/alexa-avs-sample-app/samples
cd companionService && npm start
```

Then we open a new terminal window and type the following commands to run the app, which communicates with AVS:

```
cd ~/Desktop/alexa-avs-sample-app/samples
cd javaclient && mvn exec:exec
```

Finally, we will have our screen as shown in Figure 4.1

4.3 WakeWord module

Once AVS has been implemented, it was required to change the name string inside the WakeWord in order to trigger the Service itself. As is widely known the companies who develop speech recognition systems based in AI, use their own trained Hotword that is also cloud-based and which requires a pre-start connection of the system itself. Another problem of this initiator type is that the word string is not customizable to the user choose their own trigger word, being limited (in the case of AVS) to “Alexa”, “Echo” and “Amazon” words. However, using a WakeWord as Snowboy give us the possibility to train and execute our own name string, in our case the Hotword “*MASHI*”.



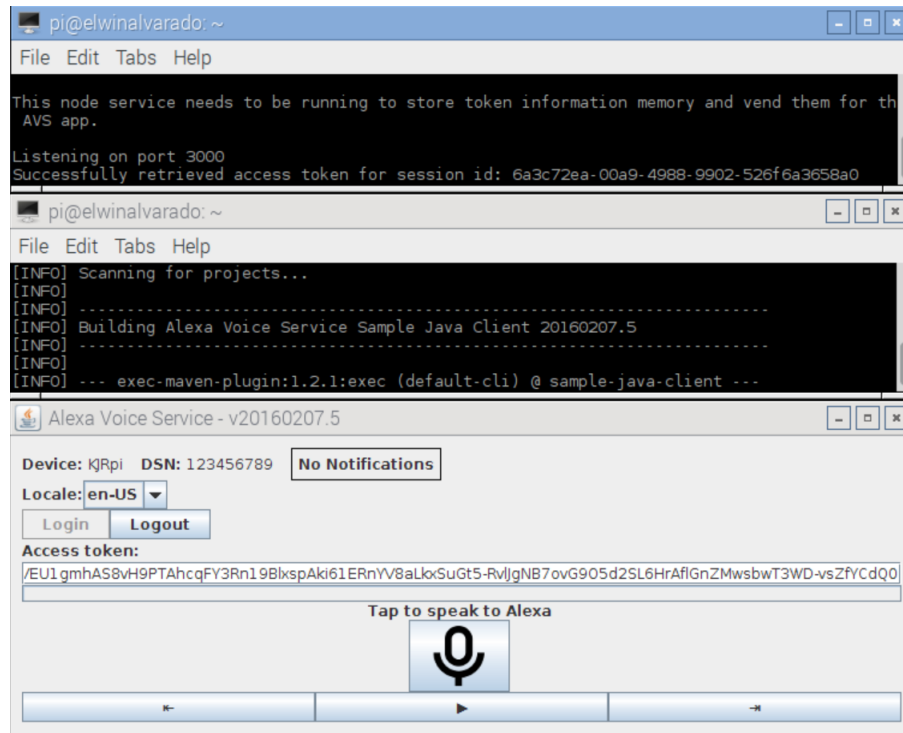


Figure 4.1: AVS Running

4.3.1 Training

In order to we could train the WakeWord agent we first had to make a Snowboy account as developers. Whit this account the online service with give us a token to use for start training our model, the training process then can be executed with the next Python script:

```
import sys
import base64
import requests

def get_wave(fname):
    with open(fname) as infile:
        return base64.b64encode(infile.read())
endpoint = 'https://snowboy.kitt.ai/api/v1/train/'

# Here it goes the information we set in our account such as the token given
# by the service, the hotword we desire to train, the language of the training
# word, the age of the user who is generating the sample, gender of the user,
# and the source of the sample.

token = "XXXXXX"
hotword_name = "MASHI"
language = "en"
```

```
age_group = "20_29"
gender = "M"
microphone = "Slave_RasPi_USB0"

# The following code set 3 samples recording of our voice as a .wav file and
# sent them to the online service in order to train our model.

if __name__ == "__main__":
    try:
        [, wav1, wav2, wav3, out] = sys.argv
    except ValueError:
        print "Usage: %s wave_file1 wave_file2 wave_file3 out_model_name" %
            sys.argv[0]
        sys.exit()

    data = {
        "name": hotword_name,
        "language": language,
        "age_group": age_group,
        "gender": gender,
        "microphone": microphone,
        "token": token,
        "voice_samples": [
            {"wave": get_wave(wav1)},
            {"wave": get_wave(wav2)},
            {"wave": get_wave(wav3)}
        ]
    }
    response = requests.post(endpoint, json=data)
    if response.ok:
        with open(out, "w") as outfile:
            outfile.write(response.content)
        print "Saved model to '%s'." % out
    else:
        print "Request failed."
        print response.text
```

4.3.2 Execution

Once we have trained and download our Personal Model we will modify the AVS files in order to use our personal Hotword “*MASHI*” and do some preliminary work before

execute our trained model using the AVS. First, we will replace our hotword model of AVS (alexu.umdl) with our personal model (MASHI.pmdl) with the next instruction in a new terminal:

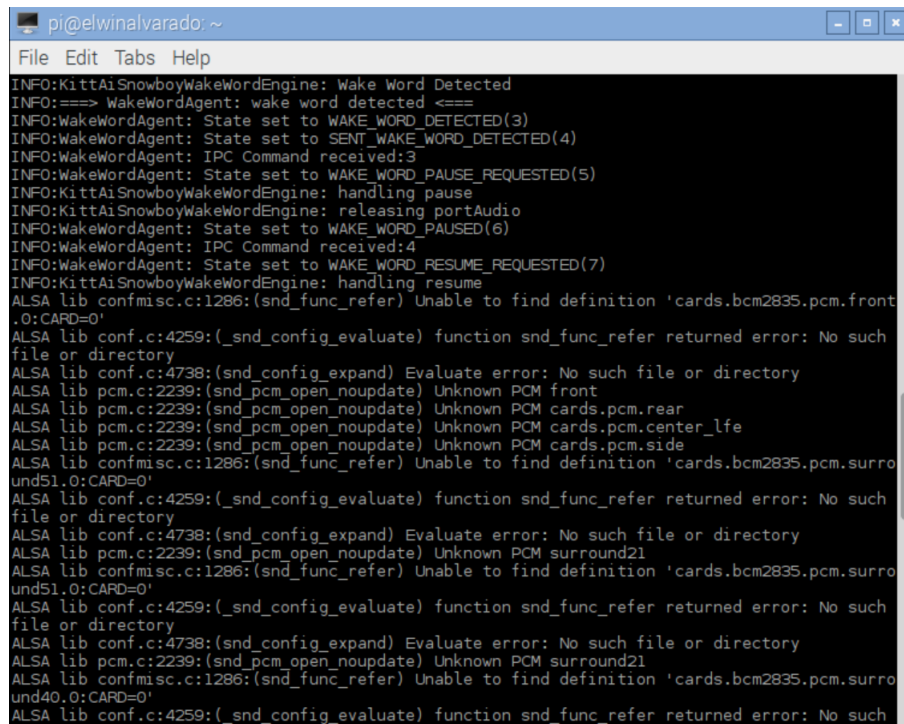
```
cp MASHI.pmdl \${AVS_PATH}/samples/wakeWordAgent/ext/resources/alexu.umdl
```

(Note: AVS_PATH is the path where it has been installed AVS)

The module can be run in a new terminal as follows:

```
cd \${AVS_PATH}/samples/wakeWordAgent/src &&. /wakeWordAgent -e kitt_ai
```

For a correct execution, this software module has to be run after the AVS module. As we can see in Figure 4.2, when the module is running correctly it give us the information, threat started, for any process made correctly.



```

pi@elwinalvarado: ~
File Edit Tabs Help
INFO:KittAISnowboyWakeWordEngine: Wake Word Detected
INFO:====> WakeWordAgent: wake word detected <====
INFO:WakeWordAgent: State set to WAKE_WORD_DETECTED(3)
INFO:WakeWordAgent: State set to SENT_WAKE_WORD_DETECTED(4)
INFO:WakeWordAgent: IPC Command received:3
INFO:WakeWordAgent: State set to WAKE_WORD_PAUSE_REQUESTED(5)
INFO:KittAISnowboyWakeWordEngine: handling pause
INFO:KittAISnowboyWakeWordEngine: releasing portAudio
INFO:WakeWordAgent: State set to WAKE_WORD_PAUSED(6)
INFO:WakeWordAgent: IPC Command received:4
INFO:WakeWordAgent: State set to WAKE_WORD_RESUME_REQUESTED(7)
INFO:KittAISnowboyWakeWordEngine: handling resume
ALSA lib confmisc.c:1286:(snd_func_refer) Unable to find definition 'cards.bcm2835.pcm.front
.0:CARD=0'
ALSA lib conf.c:4259:(_snd_config_evaluate) function snd_func_refer returned error: No such
file or directory
ALSA lib conf.c:4738:(snd_config_expand) Evaluate error: No such file or directory
ALSA lib pcm.c:2239:(snd_pcm_open_noupdate) Unknown PCM front
ALSA lib pcm.c:2239:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2239:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2239:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib confmisc.c:1286:(snd_func_refer) Unable to find definition 'cards.bcm2835.pcm.surro
und51.0:CARD=0'
ALSA lib conf.c:4259:(_snd_config_evaluate) function snd_func_refer returned error: No such
file or directory
ALSA lib conf.c:4738:(snd_config_expand) Evaluate error: No such file or directory
ALSA lib pcm.c:2239:(snd_pcm_open_noupdate) Unknown PCM surround21
ALSA lib confmisc.c:1286:(snd_func_refer) Unable to find definition 'cards.bcm2835.pcm.surro
und51.0:CARD=0'
ALSA lib conf.c:4259:(_snd_config_evaluate) function snd_func_refer returned error: No such
file or directory
ALSA lib conf.c:4738:(snd_config_expand) Evaluate error: No such file or directory
ALSA lib pcm.c:2239:(snd_pcm_open_noupdate) Unknown PCM surround21
ALSA lib confmisc.c:1286:(snd_func_refer) Unable to find definition 'cards.bcm2835.pcm.surro
und40.0:CARD=0'
ALSA lib conf.c:4259:(_snd_config_evaluate) function snd_func_refer returned error: No such

```

Figure 4.2: WakeWord Running

4.4 Alexa Skills Kit

Using the AVS and the Alexa Skill Kit (ASK), we will provide to our system the numerous ready-to-use capabilities available for use freely in the ASK library. For now, the skill that we will enable in order to implement the interaction between our Slave-RasPi and the

reactive module (and thus with the Particle API) is the skill called Patriot, also inside the particle code we will use the Patriot IoT library in order to send and control our Photon device.

4.5 Particle Photon

The Particle Photon is a Wi-Fi enabled microcontroller that has the advantage against the Arduino of being built with a Cloud-Based processor in which we can access virtually from every computer with internet connection. This microcontroller can work as a reactive IoT module for the ASR Interface. This module acts as an extension of the processing phase and gives feedback data useful for further control implementations such as machine learning.

A structure can be implemented with several number of Photons in order to extend the system control and communications to the necessary environment and can be as big as our internet Wi-Fi network. This structure also improves the presence of our mobile robot in a larger embedded system. As it is an IoT solution for the wired Arduino microcontroller, it depends totally on Wi-Fi connection, however, for no internet connection access an Arduino can implement this reactive module.

When we connect our Particle device to the Cloud for the first time, it will be associated with our account, and only we will have permission to control our Particle device using an access token provide by the API.

4.5.1 Particle API

The Particle Cloud API is a REST API meaning that we use the Uniform Resource Locator (URL) in the way that it's intended. In this case, the unique "resource" in question is our Photon. As every device has a URL, which can be used to GET variables, POST a function call, or PUT new firmware. The variables and functions that you have written in your firmware are exposed as sub-resources under the device. All requests to the device come through our API server using TLS security. The Particle API accepts requests in JavaScript Object Notation (JSON) and it always replies with JSON

4.6 Master-RasPi Implementation

The proposed Master-RasPi architecture will be implemented as the communication between the Master-RasPi and the Slave-RasPi.



4.6.1 API Scraping

API Scraping is possible using two python libraries for Web API Scraping;

- urllib3
- requests

In order to get a streaming response for the given event feed using Web API scraping, a general python code is implemented as follows:

```
# Function using urllib3
def with_urllib3(url):
    import urllib3
    http = urllib3.PoolManager()
    return http.request('GET', url, preload_content=False)

# Function using requests
def with_requests(url):
    import requests
    return requests.get(url, stream=True)
```

Then in the same script we have to set the URL depending on which section of the Particle API we want to access, as we mentioned in the section 4.1.4, for any authentication or access granted we need the **DeviceId**, **AccessToken** and if we want to stream directly into a specific event the **EventName** also will be needed.

There are numerous options in which we can set the access to the information about our device or devices, the most relevant ones for our purpose are listed below.

```
# For accessing to all the data in our Device the URL is set as follows:
url = 'https://api.particle.io/v1/devices/*DeviceId*?access_token=*AccessToken*'

# For accessing to all the events in our Device the URL is set as follows:
url = 'https://api.particle.io/v1/devices/*DeviceId*/events?
access_token=*AccessToken*'

# For accessing to a particular event in our Device the URL is set as follows:
url = 'https://api.particle.io/v1/devices/*DeviceId*/events/*EventName*?
access_token=*AccessToken*'
```

Finally, in order to GET the data into our Processing Module we finalize our script with the next code:

```
response = with_urllib3(url)
# or response = with_requests(url)

client = sseclient.SSEClient(response)
for event in client.events():
    pprint.pprint(json.loads(event.data))
```

From this point we can use the information obtained entirely as we wanted, for example, we could set this data into a variable for study its behavior through the time and/or used this variable data for statistical analysis. Also, this data could use to learn the users' behaviors, predict and adapt our robot to the users' requirements.

Chapter 5

Evaluation

As at the time we did not have any pilot test in which we would concentrate the Tuning of the modules that interact with the system, for the evaluation phase, we tested the principal modules individually in order to check their stability and reliability. Having these results would give us feedback for the future setting of the complete platform.

5.1 WakeWord engine

Using our own variant of the hypothesis testing [19], we design a practical test where we would analyze the performance of the module that perhaps will be the most important of the system. We analyzed that the WakeWord could be one of the most important fragments for the ASR Interaction and for the user due to the next points:

1. Is the first module that interact with the user.
2. Is the module which preserve the Off-life structure (thus the safety we want to achieve depends on it).
3. As it has a personal trained model, we don't know the effectiveness of it.

Now focusing in points 1 and 2, we will need to avoid “false positives”, which will leave us with false connections to the AVS cloud and consequently the safety we proposed before will be insignificant.

Then, we will focus into avoid “false negatives” which will basically be focused in point number 3, giving us the efficiency of the personal model that we trained.

5.1.1 False Positives Analysis

As we presented in the section 3.3 with the ASR technology analysis, we could rather use PocketSphinx as the trigger for the AVS or also use the Snowboy for the same task, then considering these two options we carried out our test having in mind what we presented previously in section 5.1.

This first test was made using just the WakeWord module without any other module that interacts with it. We put our microphone in front of the radio for 7 days, then we analyzed the false positives that were presented in the AVS API. According to the National Center for Voice and Speech [20], the average rate for English speakers in the US is about 150 wpm. Taking this average in mind we set device into a radio station that used the around the same average of wpm and the results are the next ones:

	Total Samples	False Positives	Accuracy (%)
PocketSphinx	1512000	150652	90.0362
Snowboy	1512000	587	99.9611

Table 5.1: WakeWord comparison

As we can observe in the Table 5.1, Snowboy stands out significantly, having almost 10% more effectiveness than PocketSphinx.

Our hypothesis about why PocketSphinx presented quite more false positives (or false alarms) can be analyzed in the way both algorithms are trained. In one hand, the PocketSphinx is trained with direct tuning as a sub-word-based speech recognition, consequently making a lot of false negatives, this process can be improved by an adaptation of the acoustic model [21], however this adaptation takes several intricate steps that will fall into a more complex implementation.

5.1.2 False Negatives Analysis

The performance of a hotword detection usually depends on the actual environment, for example, if it is used with a quality microphone, on the street, in a kitchen, or if there is any background noise, etc. So, we feel it is best to evaluate by the response against false positive and false negatives in a real environment situation.

According to Alpine hearing protection [22], humans can hear sounds between 0 and 140 decibels. 0 decibel does not mean that there is no sound, merely that we cannot hear it. 0 decibel is the so-called hearing threshold for the human ear. Because of that, we set the Quiet speech to 40 dB, Audible to 60 dB and Loud to 75. Also, using this same scale

we simulate the Quiet library environment at 30 dB, the Average Mall at 55 dB and the Night Club environment at 85 dB.

According to the samples, we took a recording of 3 volunteers (a man, a woman and a little girl), in the 3 different velocities (Slow, Normal and Fast), then we generate a continuous 16 hours recording of the speech of every volunteers (1 sample per minute fluctuating between 40 to 75 dB) giving us 960 samples to test with. In a first experimentation about the environments, we noticed that simulating a mixed noise around the 30 dB wasn't significant due to the WakeWord engine detected all the intents generated over 42 dB. Because of that, we just experiment simulating the environmental noise of an Average Mall and a Night Club.

	Total Samples	Slow	Normal	Fast	Total
Man	(3x) 960	923	941	925	
Accuracy (%)		96,1458	98,0208	96,3541	96,8402
Woman	(3x) 960	688	706	742	
Accuracy (%)		71,6666	73,5416	77,2916	74,1666
Kid	(3x) 960	690	699	697	
Accuracy (%)		71,8750	72,8125	72,6041	72,4305

Table 5.2: Correct Triggering in an Average Mall Environment

As it is seemed in Table 5.2 the Snowboy engine does a really good job detecting the man's voice, achieving an effectiveness of 97%. However, this efficiency falls down to 74% and 72% detecting the woman and the girl's voice.

	Total Samples	Slow	Normal	Fast	Total
Man	(3x) 960	223	263	278	
Accuracy (%)		23.2291	27.3958	28.9583	26.5277
Woman	(3x) 960	198	197	133	
Accuracy (%)		20.6250	20.5208	13.8541	18.3333
Kid	(3x) 960	121	116	175	
Accuracy (%)		12.6041	12.0833	18.2291	14.3055

Table 5.3: Correct Triggering in a Night Club Environment

Also in Table 5.3 is important to notice that in an 85 dB environment is basically impossible to detect correctly any intent of any subject.

Finally, in both tests we could conclude that there is a lot of ineptitude detecting the woman and the girl's voice, and this is mainly because Personal Models are supposed to

only work well for the person who provides the audio samples. Consequently, for a better efficiency and to use our model that works well for everyone we have to train our model as a universal model.

In order to solve our efficiency problem and looking for the universal model we figured out that we are required to take 500 samples in order to train a universal model. Pursuing this goal, we observed that there is two ways of train our model, as we saw in section 4.1.1 the direct recording can be implementing locally, however there is an online method to record samples, giving us the capacity to explode the social media networks in order to recollect the necessary real data.

Being conceived as a Social Platform, it was decided to use social networks such as Facebook and Twitter in order to training the WakeWord with this online method, then by the time we can recall the 500 samples we would have the possibility to create our Universal Model which would be more effective than the personal model.

As we can see in Figure 5.1, by the time we wrote this Master's Thesis, 28 samples have been uploaded using social networking. We expect that soon we could achieve the 500 samples using more social media as a boost the solution for our problematic.

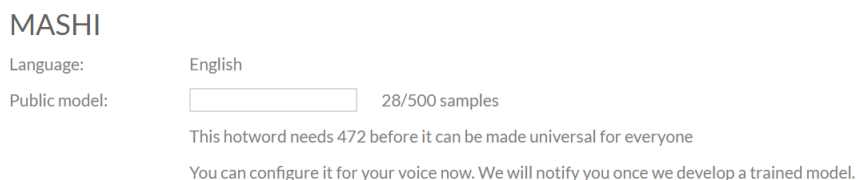


Figure 5.1: Social Networking Samples

5.2 Reactive IoT Module

As we declared in section 4.4, our architecture will be using the ASK abilities as a bridge between the AVS and our Particle Photon (and thus to the Particle API). In order to test this interaction between the voice commands and the Particle Photon and, what it will be the reactive module, we designed a test where in an imaginary scenario we control 4 lights of a house (garage, room, kitchen and living room), then we set the code of the Particle Photon as we can see in the Figure 5.2.

```

1 #include <IoT.h>
2 #include <PatriotLight.h>
3
4 IoT *iot;
5
6 void setup() {
7     iot = IoT::getInstance();
8     iot->setControllerName("Particle API");
9     iot->begin();
10
11     // Create devices
12     Light *red = new Light(A5, "red");
13     Light *yellow = new Light(A4, "yellow");
14     Light *blue = new Light(A3, "blue");
15     Light *green = new Light(A2, "green");
16
17
18     // Add them
19     iot->addDevice(red);
20     iot->addDevice(yellow);
21     iot->addDevice(blue);
22     iot->addDevice(green);
23
24     // Setup behaviors for our devices
25     iot->addBehavior(new Behavior(red, "garage", '>', 0, 100));
26     iot->addBehavior(new Behavior(yellow, "room", '>', 0, 100));
27     iot->addBehavior(new Behavior(blue, "kitchen", '>', 0, 100));
28     iot->addBehavior(new Behavior(green, "living room", '>', 0, 100));
29 }
30
31 void loop() {
32     iot->loop();
33 }

```

Figure 5.2: Photon Code

5.2.1 ASK and AVS API Setting

Once the code is verified and flashed into the Photon, we go to the AVS API and activate the skill called Patriot. This will give us the capacity of search and control the devices we created inside the code of the Photon.

Finally, we go to the Smart Home option of the AVS API and select the “Discover” option inside the Devices Window. From now on, we should see the devices that we can control in the API as we can see in Figure 5.3.

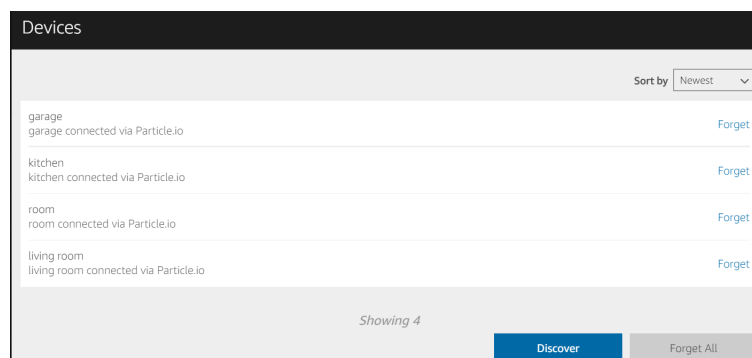


Figure 5.3: AVS API

In order to execute the test and see the results, we implement a circuit built on breadboard with LEDs (see Figure 5.4), where each color represents the intent attribute

which we can be set to the instructions “on” and “off”. The intent, attributes are:

- “garage” - Red light
- “room” - Yellow light
- “kitchen” - Blue light
- “living room” - Green light

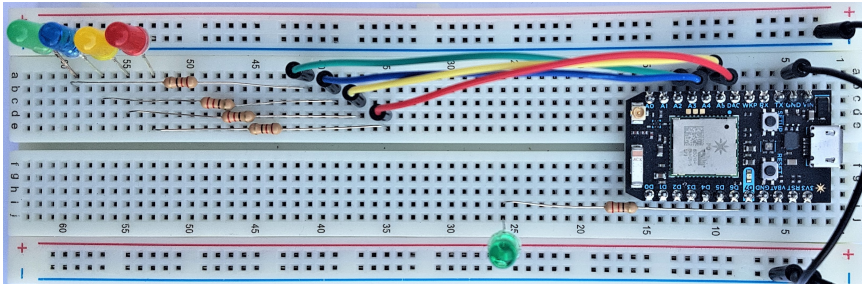


Figure 5.4: breadboard

5.2.2 Results

Testing 100 times turning “on” and “off” the lights with voice commands we achieve the 100% of effectivity, obviously avoiding the issues we exposed in the section 5.1 in context of the WakeWork module. We also could see the interaction and the variation of the variables at the Particle API as we can see in the Figure 5.5).

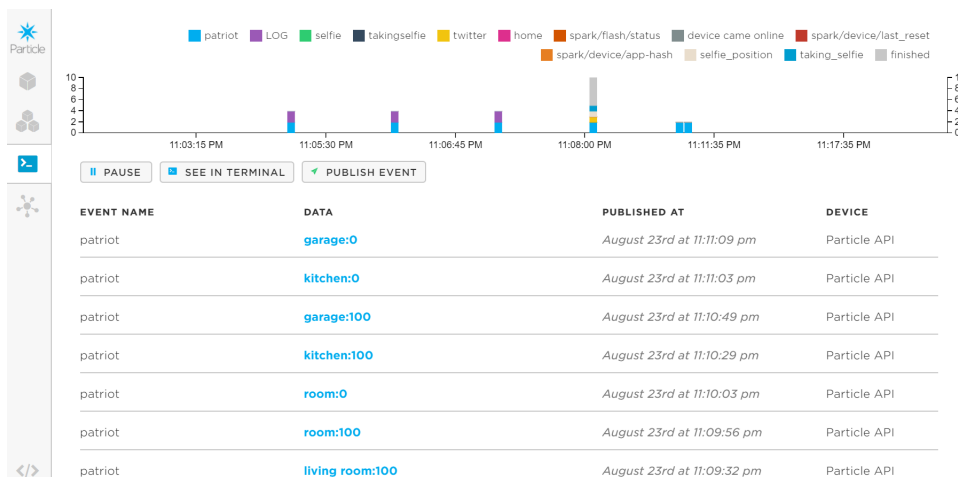


Figure 5.5: Particle API

From here we could also obtained the interaction data by scraping method, a preview of what we could acquire is shown in the Figure 5.6).


```

:ok

event: patriot
data: {"data":"room:100","ttl":60,"published_at":"2017-08-23T21:09:56.419Z","coreid":"api"}

event: patriot
data: {"data":"room:0","ttl":60,"published_at":"2017-08-23T21:10:03.844Z","coreid":"api"}

event: patriot
data: {"data":"kitchen:100","ttl":60,"published_at":"2017-08-23T21:10:29.797Z","coreid":"api"}

event: patriot
data: {"data":"garage:100","ttl":60,"published_at":"2017-08-23T21:10:49.703Z","coreid":"api"}

event: patriot
data: {"data":"kitchen:0","ttl":60,"published_at":"2017-08-23T21:11:03.141Z","coreid":"api"}

event: patriot
data: {"data":"garage:0","ttl":60,"published_at":"2017-08-23T21:11:09.039Z","coreid":"api"}

```

Figure 5.6: Particle API scraping

5.3 General Evaluation

During the testing and evaluation phase of the principal modules, we receive an invitation of Dr. Cecilio Angulo in order to participate in the Barcelona Maker Faire 2017. An event that was made during the 17th to the 18th of June at la Fira Barcelona Montjuic. This Maker Faire hosts the most inspired, talented and ambitious makers, enthusiastic to share all their knowledge and creativity through exhibitions, workshops, and interactive demos. Thus, this opportunity gave us the chance to present and discuss the latest results and contributions about our platform with other leading international experts and receive feedback about the work doing so far.

The Maker Faire gave us more than just data to have in mind using the platform with people, also it provided us with user's feedback about the social weight that people experiment when are in presence of a robot, having more with *MASHI* interaction around the 80% of the kids, being the other 20% people that asked about the platform itself.

Since, the first interaction between the semi-finished platform and the users we noticed an improvement that would affect the final test of our Interface, as this improvement was involving more the mechanical part rather than the speech interaction with the user we didn't realize that, when the tests for the Master-Raspi arrive will affect the fluidness of our tests. Once we were executing the test for the Slave-Raspi at the time we had a moment into concentrate in this mechanical improvement that the platform could receive. The mechanical improvement as we can appreciate in Figure 5.7, was the arms system.



Figure 5.7: Mechanical Arms

5.3.1 Mechanical Design

Due we have to make several tests into the arms control with the ASR interface, we have to have a platform planned to make a lot of tests, after some interaction at the Maker Faire, we analyzed that an improvement in the mechanical system had to been made before continuing with our tests. For this purpose, we take our knowledge gained from our 5 years' experience in the mechanical design field and applied into the actual platform in order to expand the capabilities of the *MASHI* Robot. Also, as at that time, we were working as mechanical designers for an engineering company, we decided to take a look of the robot arms they used, their mechanical advantages and for what purposes they used these robot arms.

Using as an example a Palletizing Robot Arm (see Figure 5.8) and another robot arms as reference, we design in a Computer Aided Design (CAD) software a symmetric robot arm that uses an important mechanical advantage for the mechanical architecture of the platform. The servomotors are as close as possible form the principal base, with this structure the torques of the servomotors can be exploited in a better way due the moment of inertia is minor than in the previous design, and then the rotation becomes easier.

As we can appreciate in Figure 5.9 the designed concept arm will stay as a 2 Degree of Freedom (DoF) arm but then it will be easily extended to 4 DoF with the use of another 2 smaller servomotors. Also, we had in mind to reutilized all the fasteners we could find around in order to help the environment.



Figure 5.8: Palletizing Robot Arm



Figure 5.9: CAD Design

Once all the parts were designed, we pre-assembled all the joints and move then around to assure the correct movements of the arm and to avoid collisions of the parts.

5.3.2 3D Printing and Assembly

With all the parts tested in the CAD assembly we start printing the components. First, we saved all parts as STL files in order to convert them with Slic3r to G-Code, and thus been printed in our personal 3D printer. Once all parts were printed, we continued assembling the entire arm and set the selfie camera at the end of the griper (as it was in the previous arm). Finally, we could see in Figure 5.10 the new arm look in a side to side

comparison to the old arm, this preassembly will stay in this way for further tests in the complete structure.



Figure 5.10: Arm Comparison

5.3.3 Programming

As one of the many uses of the platform, *MASHI* was declared the first selfie robot in the world, take-a-selfie execution was made by the tele operator and sending to the social network Twitter. As we want to execute this action automatically, it gave us the opportunity to prove what we concluded in section 4.2.1.

With this in mind we adapt our code of the what we could achieve using the Master-RasPi code as follows:

```
# First we import all the necessary libraries for the stated process

import json
import pprint
import sseclient
```

```
import time
import sys
import os
import pygame
import pygame.camera
from pygame.locals import *
from twython import Twython
from bs4 import BeautifulSoup

# Get a streaming response for the given event feed using urllib3

def with_urllib3(url):
    import urllib3
    http = urllib3.PoolManager()
    return http.request('GET', url, preload_content=False)

# We give the access Keys for the Twitter API

CONSUMER_KEY = 'XXXXXXXXXX'
CONSUMER_SECRET = 'XXXXXXXXXX'
ACCESS_KEY = 'XXXXXXXXXX'
ACCESS_SECRET = 'XXXXXXXXXX'

api = Twython(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_KEY, ACCESS_SECRET)

# We give the access Keys for the Photon API searching for a defined event

url = 'https://api.particle.io/v1/devices/340020001147353236XXXXXX/events/
taking_selfie?access_token=XXXXXXXXXX'

response = with_urllib3(url)

client = sseclient.SSEClient(response)

for event in client.events():
    pprint.pprint(json.loads(event.data))

    # We give define the scraped data to a variable
    r = json.loads(event.data)

    # Check that r is not empty, if is empty means that the event
```

```
# have not occur yet
if not r:
    print("List is empty")
else:
    from dxl.dxlchain import DxlChain

    # We open the serial device for the servomotors
    chain=DxlChain("/dev/ttyUSB0",rate=1000000)

    # Load all the motors and obtain the list of IDs
    motors=chain.get_motor_list()

    # Discover all motors on the chain and return their IDs
    print motors

    # Move to the pre-saved Selfie position
    chain.goto(9,1023,39) # Motor ID 9 is sent to Selfie position
    chain.goto(10,390,58) # Motor ID 10 is sent to Selfie position

    # We Take the Selfie
    time.sleep(3)
    pygame.init()
    pygame.camera.init()
    cam = pygame.camera.Camera("/dev/video0", (640,480))
    cam.start()
    image = cam.get_image()
    pygame.image.save(image, 'webcam.jpg')

    photo = open('webcam.jpg', 'rb')

    # We sent it to the Twitter with a comment
    api.update_status_with_media(media=photo, status=
    '#ReceivingUpdates #Testing #MASHIhasvoice #ASRinterface #VUI')
    cam.stop()
    pygame.quit()

    # Move to Home
    chain.goto(10,710,58) # Motor ID 10 is sent to Selfie position
    chain.goto(9,800,39) # Motor ID 9 is sent to Selfie position

    # Disable the motors
```

```
chain.disable()
```

We have been interacting with the system several times and, by now the interface has all positive outcomes. However, we will still trying to make the system fail in order to see if there is still gap for improvements in this ASR interface we could analyze. Lastly, the result of this general implementation will be shown during the presentation of the Thesis.

5.4 Future Work

The first future work that we consider, is the link between our ASR Interface and the Facial Recognition System, this in order to have a better interaction with the users and also to create new ways of interaction, such as state a given name for each user as a feedback and thus create a more natural human-robot communication.

As our objectives are to insert to *MASHI* in public buildings such as Hospitals, Museums and Elderly Care Homes, we have in mind to design a new Exploratory Usability Test for each location, this in order to tune the system with the skills and configuration to satisfy the necessities and requirements for each particular sector.

Depending of the necessities of each sector, we will also contemplate the design of a display interface which could be used in addition to the ASR Interface system during the interaction; subsequently the users could receive a visual feedback such as a video or an image instead of just an audio feedback.

Last but no least, we consider as a future work to study the different ways we could apply machine learning in order to take advantage of the data acquisition as we implemented with API scraping, and with this, could learn from the users' behaviors.

Chapter 6

Usability Testing Design

In order to explore the possibility of a new collaboration with the authors of the previous work [2], we decided to design a Usability test. First, because we will apply it to tune the platform with the capabilities and skills necessary to satisfy the real requirements of the cultural center, and the second, because will use the received feedback in order to appreciate the performance and the possible improvements of the complete robotic platform given by the users' behaviors.

As there is more probability to develop this usability test inside the cultural center, and then repeat the same approaches that were implemented before with *MASHI*, we will focus our Test in the same museum-type environment. Thus, before the test we will have to tune our system in order to follow all the commands described.

After the interaction, we will also ask to the people, rate their quality of their interaction with the ASR Interface and finally, we will ask them to record their voice saying the word "*MASHI*" 3 times, using the python script given in the section 4.3.1, with this extra collaboration we can use their recordings in order to achieve our goal of 500 samples and give to *MASHI* an universal model for a better performance of the WakeWord module.

6.1 Presentation

Dear user,

We are working in the implementation of an Automatic Speech Recognition Interface in order to be able to interact and control our robotic platform *MASHI* with voice commands. The objective of this test is to verify that the implemented voice commands are useful, easy and intuitive to use for the people. Also too see if new commands could be added in order to improve the human-robot communication.

Your collaboration is highly appreciated. This experimental test has a duration of 15 minutes. The analysis and the results are completely anonymous. We will use this data only for academically purposes.

Thank you in advance.

6.2 Instructions

In front of you is our Robot *MASHI*, she gives you a noise feedback when she hears her name, she can answer a question than can be searched at internet or follow a set of commands we implement on her. Also, in front of the robot is a board with different LEDs that simulate the different rooms lights in the cultural center, the main temperature, the principal and the back door. Finally, *MASHI* has the capacity of taking a selfie with you at her side and send it automatically to her Twitter.

Please, follow the following instructions:

1. Say: "MASHI".

If you hear a "beep" continue asking something you would search in Wikipedia.

2. Say: "MASHI".

If you hear a "beep" continue asking a mathematics calculation that you know the answer.

3. Say: "MASHI, what time is in " and say the your current city.

If the answers are correct continue. Otherwise repeat the previous steps.

The following commands will test the voice controls:

4. Say: "MASHI, turn on the hall room lights". (Instruction 1)
5. Say: "MASHI, raise the temperature 5 degrees". (Instruction 2)
6. Say: "MASHI, close the principal door". (Instruction 3)
7. Say: "MASHI, set the temperature to 72". (Instruction 4)

8. Say: "MASHI, lock my back door". (Instruction 5)

9. Say: "MASHI, turn off all the lights". (Instruction 6)

10. Say: "MASHI, turn on the bathroom light". (Instruction 7)

11. Say: "MASHI, tell me a fun fact". (Instruction 8)

12. Say: "MASHI, tell me a joke". (Instruction 9)

Finally;

13. Put yourself aside of her and Say: "MASHI, let me take a selfie".
(Instruction 10)

6.3 Evaluation

Instruction 1	High	—	—	—	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 2	High	—	—	—	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 3	High	—	—	—	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 4	High	—	—	—	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 5	High	—	—	—	Low
Useful:					
Understanding by the device:					
Response:					

Instruction 6	High	——	——	——	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 7	High	——	——	——	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 8	High	——	——	——	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 9	High	——	——	——	Low
Useful:					
Understanding by the device:					
Response:					
Instruction 10	High	——	——	——	Low
Useful:					
Understanding by the device:					
Response:					

6.4 Questionnaire

1. Are you interested in technology?
2. Are you interested in robotics?
3. Would you be interested to work with this ASR interface?
4. Would you be interested to develop systems with this interface?
5. Do you thought the interface seemed practical to work with?
6. Do the interface seems to be safe to work with?
7. Do you use speech recognition in your daily life?
8. Do you think you would like to use speech recognition for work?
9. Do you think the speech recognition seemed practical?
10. Do you think you would like to use the control by hand rather than speech?
11. Do you think the control by hand seemed more practical?
12. Would you add more voice commands? If yes, could you propose some?

Thank you very much for your time.

Chapter 7

Costs

7.1 Time

As we could see during the development of the Master's Thesis, we could not follow a lineal process due the complexity of the task. We were jumping from process to process, and phase to phase, because some results were not as we expected and then we had to analyze the process from the beginning and change the architecture design with every interaction of each module.

Because of that, the implementation of a Gantt diagram is quite hard (also because there was quite a lot of tasks and some of them were not linked between each other). However, analyzing the some of the tasks, merging all the related into a one general task, and setting them from the start of the first one to the end of the last one we could arrive to the Gantt diagram that we can see in Figure 7.1.

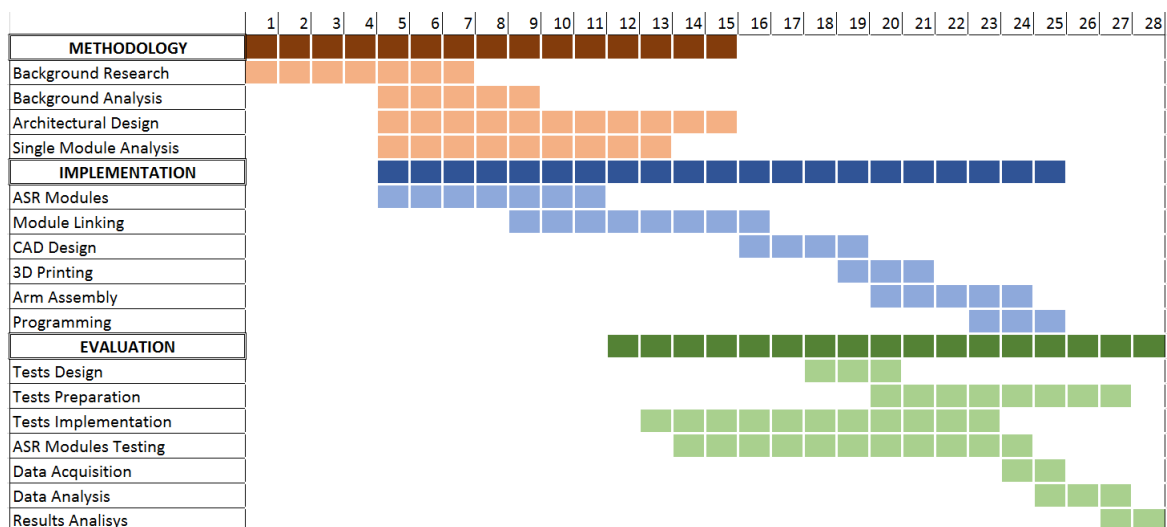


Figure 7.1: Gantt Diagram

7.2 Budget Breakdown

The total cost of the Implementation of this Master's Thesis was 8905€. Being the just the 3% from Materials and Online Services and the 97% from human labor. Now we are going to break it down.

7.2.1 Material Cost

The total material cost of the ASR Interface was 235€. Having in account that almost a half of the price was due to an extra work we proposed, we would conclude that our Interface is quite cheap in comparison of the other Assistant Robots that use a VUI, that cost around 1000€. However, in order to make a fair comparison we would have to add the other cost of the mechanical and software component that constitute *MASHI*'s platform.

Concept	Units	Unit price	Cost
ASR INTERFACE	7		133
Raspberry Pi 3	1	39	39
USB Microphone	1	12	12
Power Supply 2A	1	12	12
Stereo Speakers	1	25	25
MicroSD card 32GB	1	16	16
Particle Phonton	1	19	19
Electronics Package	1	10	10
ONLINE SERVICES	4		0
AVS	1	0	0
ASK	1	0	0
Snowboy	1	0	0
Scraping	1	0	0
MECHANICAL UPGRADE	1		102
PLA 1.75 Red Camel	120m	0,1	12
Dynamixel Servomotors 12-AX	2	45	90
Aluminum profiles	2	0	0
Fasteners Package	1	0	0
TOTAL			235

Table 7.1: Detailed material cost

7.2.2 Personal Cost

In order to calculate the worked cost per hour, we make a market research about what engineers perceive from their job. Then, we arrange this costs depending of the complexity of the task, giving 5€ to the concepts in which the task were completed almost

automatically or with a low use of cognitive skills, thus to the most complex task we are charging 25€.

As we can realize, the personal costs are the 97% of the total cost, this is basically because we focused in the use of Free-of-Charge online services, Open-Source hardware, 3D printing and reused mechanical parts. However, as we considered also the design of the robot arm (something that was not direct related with the ASR Interface), the cost of the Interface went up, being the interface alone just 133€ plus a minimal part of the programming (as we only going to need the tuning of the system). Consequently, the development of a new Interface will be the around the 200€.

Concept	Hours	Hour price	Cost
METHODOLOGY	212		3240
Background Research	84	20	1680
Background Analysis	52	10	520
Architectural Design	56	15	840
ASR Module Analysis	20	10	200
IMPLEMENTATION	213		3565
ASR Modules	28	10	280
Module Linking	56	25	1400
CAD Design	28	15	420
3D Printing	29	5	145
Arm Assembly	12	10	120
Programming	60	20	1200
EVALUATION	250		2100
Tests Design	30	10	300
Tests Preparation	28	20	560
Tests Implementation	8	5	40
ASR Modules Testing	128	5	640
Data Acquisition	16	10	160
Data Analysis	12	10	120
System Last Checks	28	10	280
TOTAL	675		8905

Table 7.2: Detailed personal cost

Environmental impact

Using the cloud computing, the environment gets a little care. When cloud services need to fluctuate, the server capacity scales up and down to fit the energy that needs to be consumed. So, we only use the energy we need and we don't leave oversized carbon footprints. Giving as a result a minimal environmental impact.

As it was planned in section 5.1.1 to re-utilize the fasteners and the aluminum profiles available in the laboratory, the design became more difficult but we did a little extra effort in order to take care of the environment and not to make more unnecessary waste from reusable components.

In the testing phase we wasted a lot of energy due to the implementation of several semi-automatic tests, but once we realize there was a correlation with the samples we improved the testing time, and consequently, wasting less energy through the final trials.

Involving the 3D printing, is widely known as a better process than manufacturing, then we chose to use PLA knowing that the PLA produces less contamination than ABS. Also all the prototype parts that were not used, we have in mind to use them in other prototyping developments.

Conclusions

During the development of the project, through the analysis of several robots and the experimentation with the ASR technologies we found that there's no perfect software or even hardware that accomplishes all the necessities of all the people at the same time. The context of having one robot that can fulfill all the available everyday tasks is still far, nevertheless it may be not as impossible as it sounds due to the continuous improvement of the accessible technologies and the constant growing of new knowledge. Because of that the Human-Robot Interaction can be always improved by using correctly the new emerging technologies.

By now it is clear that a fully voice activated smart assistant is not achievable yet as a plug & play product in order to interact with groups, without at least some amount of guidance and/or limitation. With this perspective and focused in the cultural center environment, we encourage to use the upgraded *MASHI* platform as a collaborating robot in which both human and machine can be used in what they are good at, the human for awareness, perception, flexibility and decision making and the machines for heavy load processes, precision, repetitiveness and fast processing of data.

Using the and testing the proposed WakeWord engine we appreciated that the use of Universal and Personal model could have their own advantages. The Universal model fits better when the trained word is projected to being used with numerous different people at any time, place and situations. However, a Personal Model has their advantages in the way that, as we have analyzed after the testing in section 5.1, gives us the possibility to use our voice as a fingerprint in controlled environments where the users had a different range of voice. For example, in a home where live the father, the mother and a daughter, if the trained model is developed with the father's voice, there is a just a 9% of probability that the same command executed by the mother or the daughter will be conceded by the system as a false positive. Then giving us around 97% of accuracy against the 98% efficacy of the Sensory biometric fingerprint that cost \$5-\$10 per device (than only can be used in devices with a build-in microphone such as phones, tablets and laptops).

Using the scraping method in the Master-RasPi gave us the advantage not just to

control the selfie task in a hands-free manner. Also, as we receive positive result from each interaction we could conclude that this system achieves not just the objective we wanted to accomplish, but also it gave us more to work analyze, basically because now we can execute anything we could put in a python script by voice commands.

Last but not least, we totally believe that once been developed, the usability testing that we proposed in this project will be a great solution in order to improve the interaction with our interface and at the same time will be useful for the system as a custom-made service for each public sector.

Acknowledgements

First of all, I would like to thank my parents who gave me the opportunity to be in this great world and gave me also the chance to pursue my happiness in this life. I would like to thank my girlfriend Catherine Drouin Gilbert for all the unconditional support she has been giving me along all this time together. To all my true friends who are to my side even when they are far away. To all the members of my lovely family who have been always worried about me. To my schoolmates who have been suffered my character but always gave me a smile as an answer.

I would like to thank my assessor Cecilio Angulo who gave me this and other opportunities around the project and also for stay in contact with me all the time. To my assessor Raul Ramirez Lopez for his help and who has been aware about my progress along the Master.

To my roommates who suffered the constant noises of all the tests that I had to implement during the last months. To my friend Juan Manuel Hincapie for all the graphic content provided.

And last but not least, I would like to thank everyone who gave their voice to improved our model: Guillermo Samuel Rodriguez Herrera, Ana Karen Martinez Castellanos, Rola la Rola, Clarisa Gudino Martinez, Andrea Matus, Andrés Felipe Castrillon Lopez, Manuel Gerez, Vianney Munguia, Mariajose Castilla Caro, Rosi Hernandez, Andres Aguilar, Alex Luis, Patricia Bocanegra, Lidija Ceric, Abcvdgyio Sos Rsvg, Raul Ramirez Lopez, Abraham Gurria Viguera, Maher Ahmad Nadar, Javier Poveda Figueroa, Sergi Ubach, Nic Hache, Danne Capelo, Eva Luna and Lito Alvarez.

Bibliography

- [1] INTERNATIONAL FEDERATION OF ROBOTICS, *Executive Summary World Robotics 2016 Service Robots*, available from: <https://ifr.org/downloads/press/02-2016/ExecutiveSummaryServiceRobots2016.pdf>, accessed on March, 2017.
- [2] D. PAILLACHO, C. ANGULO, AND M. DIAZ, *An exploratory study of group robot social interactions in a cultural center*. An exploratory study of group robot social interactions in a cultural center. IROS 2015 Workshop Designing and Evaluating Social Robots for Public Settings, pages 44_48, 2015.
- [3] S. PINKER, *The Language Instinct*, HarperCollins, New York, 1994.
- [4] J. SMITH, *THE VOICE ASSISTANT LANDSCAPE REPORT: How artificially intelligent voice assistants are changing the relationship between consumers and computers*, available from: <http://www.businessinsider.com/voice-assistant-report-2017-3>, accessed on March, 2017.
- [5] W. SCHRAMM, *Procedures and effects of mass communication in Henry*, N. B. (1954) Mass, media and education: The fifty-third yearbook of the national society for the study of education, Part II. University of Chicago Press: Chicago.
- [6] S. THRUN, M. BENNEWITZ, W. BURGARD, A. CREMERS, F. DELLAERT, D. FOX, D. HAHNEL, C. ROSENBERG, N. ROY, J. SCHULTE, AND D. SCHULZ , *Minerva: a second-generation museum tour-guide robot*, in Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on, vol. 3, 1999, pp. 1999_2005 vol.3.
- [7] H. ISHIGURO, T. ONO, M. IMAI, T. MAEDA, R. KANDA, *Robovie: an interactive humanoid robot*, Industrial Robotics, 28, 498-503. 2001.
- [8] J. PUIGBOA, A. PUMAROLAA, C. ANGULO, R. TELLEZ, *Using a cognitive architecture for general purpose service robot control*, University of Catalonia, Barcelona, Spain; Pal Robotics, Barcelona, Spain. v3.4 released April 2013.
- [9] J. SEARLE, *Speech Acts: An Essay in the Philosophy of Language*, Cambridge University Press, Cambridge. 1969.

- [10] K. VOGLEY, G. BENTE, *Artificial humans: psychology and neuroscience perspectives on embodiment and nonverbal communication*, Neural Networks, 23 (8) (2010), pp. 1077-1090.
- [11] G. CHOWDHURY, *Natural language processing*, Annual Review of Information Science and Technology, 37. pp. 51-89. ISSN 0066-4200. 2003.
- [12] G. ADORNI, M. ZOCK, *Trends in Natural Language Generation an Artificial Intelligence Perspective*, Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), vol 1036. Springer, Berlin, Heidelberg. 1996.
- [13] E. GUIZZO, *Robots with their heads in the clouds*, IEEE Spectr., 48 (3) (2011), pp. 16-18.
- [14] N. MAVRIDIS, T. BOURLAI, D. OGNIBENE, *The human-robot cloud: situated collective intelligence on demand*, IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems, CYBER, IEEE (2012), pp. 360-365.
- [15] M. WAIBEL, M. BEETZ, J. CIVERA, R. D'ANDREA, J. ELFRING, D. GALVEZ-LOPEZ, K. HAUSSERMANN, R. JANSSEN, J. MONTIEL, A. PERZYLO, *Roboearth*, IEEE Robot. Autom. Mag., 18 (2) (2011), pp. 69-82.
- [16] R. FIELDING, *Architectural styles and the design of network-based software architectures*, PhD Thesis. University of California, Irvine, Information and Computer Science Department. 2000.
- [17] H. MCWILLIAM, *Analysis tool web services*, from the EMBL-EBI, Nucleic Acids Res., 2013, vol. 41 (pg. W597-W600).
- [18] O. CASTRILLO, *European Public Sector Information Platform*, Topic Report No. 2015 / 10 Web Scraping: Applications and Tools. Published: December 2015.
- [19] D. SHESKIN, *Handbook of Parametric and Nonparametric Statistical Procedures*, CRC Press. p. 54. ISBN 1584884401. 2004.
- [20] THE NATIONAL CENTER FOR VOICE AND SPEECH, *Average rate for English speakers*, available from: <http://www.ncvs.org/ncvs/tutorials/voiceprod/tutorial/quality.html>, accessed on August, 2017.
- [21] CMUSPHINX, *Adaptation of the acoustic model*, available from: <https://cmusphinx.github.io/wiki/tutorialadapt/>, accessed on August, 2017.

- [22] ALPINE HEARING PROTECTION, *5 sound levels in decibels* , available from: <https://www.alpinehearingprotection.com/wiki/5-sound-levels-in-decibels/> , accessed on September, 2017.
- [23] Y. DENG, *The impact of manufacturing parameters on submicron particle emissions from a desktop 3D printer in the perspective of emission reduction*. Building and Environment, 2016.