INTERACTIVE DEMONSTRATION USING NAO ROBOTS

Panagiotis Filiotis
H00203394
MSc IT Software Systems

Supervisor:
Dr Patricia A. Vargas

Co-Supervisor:
Matthias U. Keysermann

SCHOOL OF MATHEMATICAL AND COMPUTER SCIENCES
HERIOT WATT UNIVERSITY
17 / 08 / 2015
DECLARATION

I, Panagiotis Filiotis confirm that this work submitted for assessment at Heriot-Watt university for the academic year 2014-2015 is all my own and is expressed in my own words. Any use made within it from books, journals, and websites regarding the works of others in any form (equations, ideas, tables, figures, and program) have been properly cited. A list of the references used is included at the end of the dissertation.

Name: ..................................................

Date: ..................................................

Signature: ............................................
I would like to thank my supervisor Dr Patricia A. Vargas for the great help and support which provided me any time I needed during the whole duration of completing my dissertation.

I would like to thank my co-supervisor Matthias U. Keysermann for the help and great assistance who provided me during the whole progress of my dissertation.
ABSTRACT

Robots nowadays are all around us and interact more and more in our everyday lives. One of the main characteristics of autonomous robots is their ability to perform tasks and adapt behaviors by receiving commands through compatible software environments.

In this dissertation a desktop and an Android application will be developed having as main purpose the creation of a user friendly interface that will give users the opportunity to interact with the NAO robot. Both applications will consist of a set of controllers that will deal with multiple tasks related to the humanoid autonomous robot. Tasks will include the manual control of robot movements in all dimensions of space and interactive behaviors with people through a series of predefined keywords. Communication between the robot and the software will be established wirelessly via the Internet and in the case of the smartphone application, there will be no middleware machine, instead the connection will be direct. The humanoid robot NAO from Aldebaran will be the one that is going to be the core during the designing, development and testing at each stage of the application.

All the technological aspects around NAO robots and the existent technologies regarding remote control tasks, will be described and discussed thoroughly before starting the creation of the software part. Then programming languages like Python and Java will be used to constitute the fundamentals for developing both applications.

At the final stage of this project a demonstration will take place in the Robotics lab, having as main purpose to present both applications to the public and at the same time assess their overall functionality. Questionnaires and individual comments from the users will be the core that the evaluation will be based on.
# TABLE OF CONTENTS

DECLARATION ......................................................................................................................... 1
ACKNOWLEDGEMENTS ............................................................................................................. 2
ABSTRACT ................................................................................................................................. 3
TABLE OF CONTENTS ............................................................................................................... 4
LIST OF FIGURES ..................................................................................................................... 7
LIST OF TABLES ....................................................................................................................... 9

1 INTRODUCTION .................................................................................................................... 10
   1.1 Objectives ....................................................................................................................... 10
   1.2 Professional, Legal and Ethical issues ............................................................................ 12
      1.2.1 Professional Issues .................................................................................................. 12
      1.2.2 Legal Issues .......................................................................................................... 13
      1.2.3 Ethical and Social Issues ....................................................................................... 13

2 LITERATURE REVIEW ......................................................................................................... 15
   2.1 Humanoid Robots ......................................................................................................... 15
      2.1.1 Early History of Humanoid Robots ....................................................................... 15
      2.1.2 Recent History of Humanoid Robots ..................................................................... 17
   2.2 Robots Used For Demonstrative Purposes .................................................................. 25
      2.2.1 Robothespian ........................................................................................................ 25
      2.2.2 Kodomoroid and Ontonaroid ............................................................................... 26
      2.2.3 Robotinho ............................................................................................................ 27
      2.2.4 REEM-H1 ............................................................................................................ 28
      2.2.5 TalkTorque .......................................................................................................... 28
   2.3 NAO Robots .................................................................................................................. 30
      2.3.1 Hardware .............................................................................................................. 30
      2.3.2 Software ............................................................................................................... 32
   2.4 Applications with NAO ............................................................................................... 33
      2.4.1 Desktop applications ............................................................................................. 33
         2.4.1.1 Choregraphe ................................................................................................. 33
         2.4.1.2 Webots for NAO ......................................................................................... 35
         2.4.1.3 NAO MATLAB API ................................................................................. 36
      2.4.2 Mobile applications .............................................................................................. 37
         2.4.2.1 NAO Robot Controller .............................................................................. 37
         2.4.2.2 NAO Communicator ................................................................................. 39
         2.4.2.3 Full Manage NAO .................................................................................... 40
3 ORGANIZATION ................................................................................................. 43

3.1 Requirements Analysis .................................................................................. 43
3.1 Performance Assessment ............................................................................... 47
3.2 Risk Assessment ............................................................................................ 49
3.3 Project Plan ..................................................................................................... 51

4 METHODS AND RESULTS .............................................................................. 54

4.1 Prototype 1 – Familiarization by testing basic functions of NAO robots .......... 54
  4.1.1 Design ......................................................................................................... 54
  4.1.2 Implementation ........................................................................................... 54
  4.1.3 Evaluation .................................................................................................. 57
4.2 Prototype 2 – Developing desktop’s application interface ............................. 58
  4.2.1 Design ......................................................................................................... 58
  4.2.2 Implementation ........................................................................................... 59
  4.2.3 Evaluation .................................................................................................. 64
4.3 Prototype 3 – Robot moves through commands from desktop .................... 66
  4.3.1 Design ......................................................................................................... 66
  4.3.2 Implementation ........................................................................................... 67
  4.3.3 Evaluation .................................................................................................. 70
4.4 Prototype 4 – Robot interacts through predefined keywords ......................... 72
  4.4.1 Design ......................................................................................................... 72
  4.4.2 Implementation ........................................................................................... 72
  4.4.3 Evaluation .................................................................................................. 75
4.5 Prototype 5 – Developing Android Application .............................................. 76
  4.5.1 Design ......................................................................................................... 76
  4.5.2 Implementation ........................................................................................... 76
  4.5.3 Evaluation .................................................................................................. 81
4.6 Prototype 6 – Establishing communication between Android app and NAO ... 84
  4.6.1 Design ......................................................................................................... 84
  4.6.2 Implementation ........................................................................................... 84
  4.6.3 Evaluation .................................................................................................. 86
4.7 Prototype 7 – Robot moves through commands from Android application ...... 87
  4.7.1 Design ......................................................................................................... 87
  4.7.2 Implementation ........................................................................................... 87
LIST OF FIGURES

Figure 2.1  Model of Da Vinci’s knight in Berlin and the sketches found in his notebook .......... 15
Figure 2.2  Electro Man infrastructure ................................................................. 16
Figure 2.3  Wabot – 1 ......................................................................................... 17
Figure 2.4  Wabot – 2 ......................................................................................... 17
Figure 2.5  Prototypes E1, E2, E3 created by Honda in the period 86-91 ....................... 18
Figure 2.6  Prototypes E4, E5, E6 created by Honda in the period 91-93 ....................... 18
Figure 2.7  Prototypes P0, P1, P2 created by Honda in the period 93-97 ....................... 19
Figure 2.8  ASIMO latest version ....................................................................... 19
Figure 2.9  Models by Sony AIBO and QRIO ..................................................... 20
Figure 2.10 Icub Robot......................................................................................... 20
Figure 2.11 REEM prototypes since 2006-2013 ................................................. 21
Figure 2.12 Darwin-OP Humanoid Robot ............................................................ 22
Figure 2.13 NAO robot ...................................................................................... 23
Figure 2.14 Pepper ............................................................................................. 24
Figure 2.15 Robothespians ............................................................................... 25
Figure 2.16 Ontanaroid on the left and Kodomoroid on the right ......................... 26
Figure 2.17 Robotinho perform inquiring and greeting gestures ............................ 27
Figure 2.18 REEM-H1 ....................................................................................... 28
Figure 2.19 TalkTorque ....................................................................................... 29
Figure 2.20 NAO H-25 joints and sensor parts ................................................... 31
Figure 2.21 Choregraphe interface demonstrating a sequence of connected behaviors 34
Figure 2.22 Camera viewer, Memory viewer ....................................................... 35
Figure 2.23 The Laser monitor plug-in only available on NAO Laser Head model .... 35
Figure 2.24 Webots for NAO platform .............................................................. 36
Figure 2.25 MATLAB SDK .............................................................................. 37
Figure 2.26 NAO Robot Controller .................................................................. 38
Figure 2.27 NAO Communicator Application ..................................................... 39
Figure 2.28 Full Manage NAO .......................................................................... 40
Figure 2.29 NAO Control application ................................................................ 41
Figure 2.30 iControlNAO platform ................................................................... 41
Figure 2.31 NAO Remote Control ..................................................................... 42
Figure 3.1  Project Plan chart ............................................................................ 53
Figure 4.1  NAO executing a simple behavior in Choreograph environment .......... 55
Figure 4.2  Schema diagram showing all the functionalities that the interface can support ...... 58
Figure 4.3  Main Panel of the application ............................................................ 59
Figure 4.4  The Motion window ........................................................................ 60
Figure 4.5  The Speech Window ........................................................................ 61
Figure 4.6  Settings Window which provides options related to the modification of IP, volume and the name of the robot .................................................................. 62
Figure 4.7  Demonstrations and Robotics Lab Info windows .............................. 63
Figure 4.8   Manual window providing all the necessary instructions that the robot can connect with the application................................................................. 63
Figure 4.9  Motion window providing full control of the robot's movements................................. 66
Figure 4.10  Tables displaying the results of the Speech Recognition's evaluation................... 75
Figure 4.11  Main Panel Activity on landscape and portrait mode ............................................... 77
Figure 4.12  Motion Activity on landscape and portrait mode .................................................... 78
Figure 4.13  Speech Activity on landscape and portrait mode .................................................... 79
Figure 4.14  Settings Activity on landscape and portrait mode..................................................... 80
Figure 4.15  Under Construction Activity on portrait and landscape mode................................. 80
Figure 4.16  Local network providing communication between NAO and Android Tablet........... 85
Figure 4.17  Motion panel ........................................................................................................... 87
Figure 4.18  First page of the questionnaire .................................................................................. 92
Figure 4.19  User during the evaluation process interacting with NAO................................. 93
Figure 4.20  Responses to the question "How did you find the layout of the interface".................. 94
Figure 4.21  Responses to the questions "How easy is to navigate through the application and "How easy is to use the application's interface and explore it's functionalities""....................... 95
Figure 4.22  Responses to the question "How difficult was it to find out how NAO can be connected to the application" ........................................................................................................................................................................... 95
Figure 4.23  Responses to the questions "Have you read the subtitle's messages of each window" and "How useful is the included manual window 'How to connect NAO'"............................. 96
Figure 4.24  Responses to the questions "How precisely did the keyboard commands respond" and "Has the robot executed the right action anytime you used a word to make it move?".................. 97
Figure 4.25  Responses to the question "How many times the robot hasn't responded to the voice commands" ................................................................................................................................. 97
Figure 4.26  Responses to the question “Has the robot executed the right action anytime you used a word to make it move” both on the Motion and RLI panel ................................................................. 99
Figure 4.27  Comparison on the question “How many times the robot couldn’t hear which word you used on average and asked you back to repeat the phrase” between the Motion and RLI panel ................................................................................................................. 99
Figure 4.28  Responses to the questions "Which part have you found the most easy to use" and "Which part have you found the most difficult to use" ................................................................................................................. 100
LIST OF TABLES

Table 3.1  Project Requirements ................................................................. 43
Table 3.2  Performance Assessment .............................................................. 47
Table 3.3  Project Risks and Responses.......................................................... 49
Table 3.4  Project plan tasks ........................................................................... 52
Table 4.1  Table of tests taken to check the functionality of the interface ........... 65
Table 4.2  Table of tests and results regarding the Motion functionality ............. 71
Table 4.3  Table of tests and results regarding the Android's GUI functionality .... 83
Table 4.4  Table of tests and results regarding the Android's Motion functionality . 91
1 INTRODUCTION

The word Robot as a terminology can find its roots many years ago in the Czech vocabulary as Robotica. The term definition specifies a worker executing manually compulsory labour without getting any remuneration (Xie 2003). As the term reveals, while in the first years of their existence, robots have limited use in a series of tasks including obtuse and repetitive tasks, but through the years have been developed into something more complex and useful for humans.

In nowadays robots have been increasing in numbers and can support a combination of manipulative, perceptive and communicative abilities. That expands their functionality, make them capable of executing more complicated tasks simultaneously and autonomously. Humanoid robots are a revolutionary step which has given another aspect in the field of robotics. They have turned into being more interactive and much more look like humans.

One of the robots that belongs in this category of Humanoids is the NAO robot. Aldebaran created this model having as main target to develop an engaging and interactive companion who would be able to offer many different functionalities such as speaking or dancing (Wallach & Allen 2010).

In the terms of this project, a desktop and an Android application would be developed and designed around the model of the NAO robot which will be used for demonstrative purposes. User can get familiarized with the robot through a number of tasks that the humanoid will execute each time depending the commands that will receive manually either the desktop or Android application. People with no background in programming will interact with the robot in real time and watch it demonstrate different desirable tasks.

1.1 Objectives

This dissertation includes different kinds of objectives with different weight for each one. The first objective which will be the base of the knowledge for accomplishing this project is the familiarization with the NAO humanoid robot. Because no relevant work has been done before in the robotics field a more detailed view in the infrastructure of the robot in both software and hardware aspects would be necessary. The second objective would be the development of a graphical user interface (GUI) for the desktop application which will be functional and at the same time friendly and easy to use. The third objective has been set the establishment of a
communication with the robot wirelessly and the development of some behaviors that the robot will execute. After the completion of these objectives a stage of general testing will follow and the desktop application will be fully functional. As next objective in the hierarchy is the designing and the development of a graphical user interface which will be the fundamental for the Android's part. Once this one has accomplished then communication establishment and development of features that will define the robot's behavior will be the next target. Another objective that includes all the others will be the development of my programming skills at a higher level and more specifically in Python and Java. The project that will be based on these two programming languages, surely will require higher levels of expertise, something that will make me dive more in depth in both of them. The last objective that has been outlined in this thesis would be the assessment of both applications. A demonstration will take place in the Robotics laboratory, having as main purpose to get feedback from users which will help in the final evaluation of the project. Summarizing, this project is challenging regarding my background, but simultaneously will be a great opportunity to gain knowledge and a lot of experience in both programming skills and the robotics field.

Summarising the objectives that the project has to fulfill:

1) Familiarization with the NAO architecture
2) Development of a graphical user interface for the desktop application
3) Establishment of a communication channel between NAO and the desktop app
4) Implementation of motion and speech recognition modules to the desktop app
5) Development of a graphical user interface for the Android app
6) Establish a communication channel between NAO and Android app
7) Implementation of motion and speech recognition modules to the Android app
8) Evaluation of both desktop and Android applications
9) Improvement of my personal programming skills
1.2 Professional, Legal and Ethical issues

1.2.1 Professional Issues

As the years go by, computer professionals deal increasingly with a variety of different tasks in their working environment on a daily basis. Tasks which include designing software, developing applications, administrate databases, writing code, analysing or testing systems. All these activities imply to a large amount of responsibilities that have to be faced consciously mainly because directly or not can cause harm to third parties such as employers, clients, other professionals or more general in public. The responsibilities that every computer professional has to follow respectfully contained in the code of conduct principles which has been set out officially from the British Computer Society. According to the BCS every professional is expected to have always as a basic rule to act under his own judgment and in case something is unclear or controversial an advice has to be sought.

The first guideline to be taken into account is the public interest. Privacy, security and the legitimate rights of third parties should have not trampled at any occasion. Any comportment has to avoid discriminations on the aspects of nationality, color, sex, religion, age, sex orientation or disabling condition (Mintz 2014). Also the advantages that information technologies offer have to be shared equally and generously without any differentiation.

The second chapter of the code of conduct is called professional competence and integrity. Every piece of work that is committed must be completely possessed and all the content fully understandable. Each professional has to be kept updated around all the new trends of technological world, broaden his knowledge in the field of information technology that specializes and improve his skills on a regular basis (Mintz 2014). At the same time all the other opinions and thoughts of other colleagues have to be faced respectfully.

Duty to relevant authority is one of the four sub-categories of the code. As relevant authority can be described any person or organization that controls any actions taken from the individuals in their profession. Requirements and specifications, most of the times, are these two that set under which guidelines individuals have to execute and accomplish their work. From their point of view professionals has to act with responsibility and thoroughness to avoid any kind of conflicts with the relevant authority (Mintz 2014).

Last but not least is the duty to the profession. The reputation of the profession keeps the scepters in this category and puts as the most important responsibility of each professional to keep it at a high level. Unethical or any similar kind of activities has to be avoided because they will drive to disputation of the field. Instead ways to improve the existing standards has to be
found as main purpose, except from the personal development to aid in the professional evolution of other colleagues.

In this MSc dissertation all these principles described above will be followed. The project idea is to provide a software application which will make the use of NAO robots easier for people which have no background in programming languages. For that reason the public interest goal is fulfilled because at the end of the project the final product will give the opportunity to people, meet and manipulate robot’s capabilities in an easy way without the need of having any programming skills. According the professional competence issue arises, I can admit for sure that this dissertation will help me to develop my programming skills at a higher level and expand my knowledge in the field of robotics. Heriot-Watt University will be the relevant authority which will set the guidelines that this project has to be done and will be followed strictly. All rights will be under the name of the institution and taken as granted that at any time if an issue occurs about the requirements I will seek for advice to avoid any conflicts. Finally, I will maintain the reputation of the profession in a high level through my work and my manners. Integrity and respect will be the two elements that will guide me until the end of this dissertation.

1.2.2 Legal Issues

The two major issues that come in the foreground and are related to the legal issues are the intellectual rights and any inappropriate use of third parties work without mentioning(Mintz 2014). The Intellectual rights of this dissertation are reserved to the Heriot –Watt University and any work has been committed with the relevant documents included belong to the ownership of the institute. The second part of the legal aspects refers to the obligation which everyone has when includes or partly uses somebody else’s work. Author’s name, year of publication, the title of the document, the edition number and the name of the publisher will be mentioned by using Harvard referencing system. Citing of references is the appropriate way that will be used accurately in this project to declare anything has to do with other’s work which either can be included in books, magazines, websites or scientific articles.

1.2.3 Ethical and Social Issues

Personal data privacy and appropriate use of robots as described by Isaac Asimov on 1964 are the two main concerns which a computer professional has to consider in a project which involves robots(Mintz 2014). The three laws of robotics as called by the famous scientist of 50s
can briefly enclosed below. The first suggestion is about the behavior that the robots should have against human beings and more specifically mentioned that should never have as a purpose to harm or injure a human. The second one refers to the aspect of behavior in which a robot must fully obey a human’s commands except if that contradicts with the first rule. Thirdly is referred that a robot cannot harm another robot of its kind. All these suggestions will be considered during the implementation of the code and anything is related to the actions of the NAO robot (Krichmar & Wagatsuma 2011). At the end of this thesis an evaluation procedure will take place so other colleagues or students can expose their opinion and suggest any other ways of improving both applications. This is the second leg of ethical issues which potential personal data may be exposed. All personal data which will be provided by third parties in the questionnaires will be handled very carefully so any conflicts with ethical issues will be avoided.
2 LITERATURE REVIEW

2.1 Humanoid Robots

2.1.1 Early History of Humanoid Robots

Humanoid robots are not something new in our world, but have been many years which have started employing humans how to create something that will look like them. The first attempt was in 1495 and belongs to a very famous polymath of the High Renaissance period, Leonardo Da Vinci. Sketches that found in his notebook many years after his death reveal that he has designed in detailed plans, a mechanical armored knight. The design based completely in the knowledge that Da Vinci had in the anatomy of the human body. The robot knight was capable of standing, sitting, raising its visor and without any help moves each of its arms, neck and jaw. Probably the knight had been designed to move with fluidity in combat (Da Vinci 2014).

![Image of Da Vinci's knight and sketches](image)

Figure 2.1 Model of Da Vinci’s knight in Berlin and the sketches found in his notebook (Da Vinci 2014)

Many centuries later and more specifically at 1940 the tall motor man Electro created by the engineer Joseph Barnett at the Westinghouse's factory in Mansfield. Having a humanoid appearance of a man who was 2.10 cm tall and weighting around 264lb. The robot could smoke cigarettes, play recorded speech from a vocabulary of 700 words, blow up balloons, drive on wheels which we were installed on his feet and move its head and arms. Electro couldn’t be controlled remotely, but instead could respond to voice commands using a telephone handset.
connected to its chest. Each word set up vibrations which were converted into electrical impulses, which they were operated 48 different electrical relays controlling eleven motors. Robot’s walking was based on four rubber rollers under each foot, which were related by chains and shafts connected to the main motor. Fingers, arms, head and smoking capability were controlled by ten separate motors (Schaut 2007).

Figure 2.2 Electro Man infrastructure (Buckley 2008)

In 1973 the WABOT-1 model designed by Ichiro Kato and was the first anthropomorphic robot able to walk on two legs. The development of the project started in the laboratories at Waseda University in Tokyo. The robot consisted of a limb-control system, a vision system and a conversation system. It was able to communicate with a person in Japanese and to measure distances and directions to the objects by using external receptors, eyes, an artificial mouth and artificial ears. The WABOT-1 walked with his lower limbs and was able to grip and transport objects with hands that used tactile sensors. It has been assessed that the WABOT-1 has the intellectuality of a one-and-half-year-old child.
In 1980 Ichiro Kato brought alive the WABOT-2 project. A development of the previous model had this time as a target to accomplish a more complex task such as playing a keyboard instrument. For the reason that an activity like that would require human intelligence and dexterity WABOT-2 was defined as a specialized robot rather than a versatile robot like the WABOT-1. The robot musician WABOT-2 can talk with a person, read a normal musical score with the eye which was installed in the middle of the head and play the tunes of average difficulty on an electronic instrument. The WABOT-2 is also able of accompanying a person while he listens to the person singing. This model of the robot was the first milestone in developing a "personal robot"(Kajita et al. 2014).

2.1.2 Recent History Of Humanoid Robots

In 1986 Honda engineers had a pioneer idea of setting up a project which would create a robot that could make some first steps and walk as a human being does. The first prototypes created in the laboratories of Honda called E1, E2, E3 and didn’t have initially an upper torso look like a human’s although they had a pair of legs in which the engineers focused on and tried to simulate the walk of humans(Behnke 2008). The best point they managed to reach at the end of 1991 was the third model E3 whose legs could walk at the speed of 3km/h in a similar way to the normal human walking speed.
The next steps followed from 1991 to 1993 were again three prototypes created, keeping the name of the series the same and called E4, E5, E6. In the first model they managed, by increasing the height of the knees to reach an average speed of 4.7km/h. In the final prototype of this series the designers have succeeded except for the speed to make the robot autonomously control the stability when it had to face tasks like going up or down from stairs or even when stepping over an obstacle (History of ASIMO 2015).

![Figure 2.5 Prototypes E1, E2, E3 created by Honda in the period 86-91 (Honda 2015)](image)

![Figure 2.6 Prototypes E4, E5, E6 created by Honda in the period 91-93 (Honda 2015)](image)

Honda finally managed in 1997 to design a humanoid robot that combined an upper torso body and legs looking like human’s. Again, as the previous series three prototypes implemented from 1993 to 1997 and this time the names given were P1, P2 and P3. The second model of the series was the one that made the biggest impression in the world for the stability and the realistic way it could walk. Some of the special functionalities which these robots introduced was climbing and descending stairs by using a 6-axis force sensor, open and close doors. Also, it could lift weights up to 2kg for each arm and push boxes with the limitation don’t face any resistance (History of ASIMO 2015).
After all these years in which Honda has managed to develop so many prototypes ASIMO comes into the scene and is the one will stay in the foreground from 2000 until today. Its initials which stands for Advanced Step In Innovative Mobility can clearly clarify that the goal from the first moment of its creation was not only the design aspect but the future of this robot. Through five different stages from its birth ASIMO has reached in the final stage of 2011 prototype which can be characterized as an autonomous machine, with capabilities such as making decisions to determine its behavior regarding the environmental stimuli(Meadows 2010). Some of the main characteristics are running, walking on uneven slopes and surfaces, turn in all directions smoothly, climb, grasp objects, avoid moving obstacles and respond to simple voice commands(History of ASIMO 2015).

The world of robotics is far too vast and fast-growing that makes it very difficult to be a comprehensive list with all the robots developed after ASIMO. Below are some of the most notable developments in robotics since 2000.
In 2003 Sony unveils a bipedal humanoid robot called QRIO. The initials stand for Quest for cuRIOsity and engineers of Sony expressed the idea that they wanted to develop a robot for entertainment reasons that would be the next generation of the cute robotic dog AIBO which have created in 2001. QRIO was about 0.6m tall and weighed 7.3kg had a camera in the centre of its forehead, a total of 24 joints within its body, make it capable of only balance on one foot and even kick a ball except for the basic movements that could execute. The maximum speed that QRIO could run was 23cm/s and noteworthy is that in 2005 put its name in Guinness World Records as being the first bipedal robot that was running with both legs off the ground at the same time (Sony 2015).

![Figure 2.9 Models by Sony AIBO and QRIO (PC Magazine 2010; Arobose 2009)](image)

iCub is a humanoid robot developed at Italian Institute of Technology in 2006 as part of the famous project RoboCup and has been used widely since then for research purposes into human cognition and artificial intelligence. The robot’s name stands for Cognitive Universal Body, it has a height of 100 cm, weighs 23 Kg, can recognise and manipulate different objects. Each of its hands has 9 degrees of freedom and can feel when touching objects almost like a human. The head is responsible for all the types of movements, recognizing and dealing with commands. Two cameras, two microphones, gyroscopes and accelerometers are installed inside its head. In the heart of the robot, which brings it to life, there is an Intel Core2Duo 2.16 MHz processor. Fifty three different motors move the head, hands, waist and the legs. Functionalities make the robot capable to see, hear, it has the sense of proprioception and obviously moves. Future work has been focused on

![Figure 2.10 iCub Robot (Juxi n.d.)](image)
improving the sense of touch and to grade how much force it exerts on the objects (Frank et al. 2012).

REEM-C is another humanoid robot built by PAL Robotics in Spain. The robot’s history counts three ancestors before takes its latest form of 2013. The humanoid has a height of 1.65cm, weighs 80kg with 44 degrees of freedom and has the ability to move with the speed of 1.5km/h. The upper partial body consists of a torso, two motorized arms and a motorized head which is controlled by an Intel CPU i7 -2.10 GHz and has 9 sensors which include 4 sonars and 2 lasers, 2 six axis force sensors in ankles and 1 UMI. All these sensors allow the robot avoid obstacles, both dynamic and static, and walk safe. Some of the main functionalities of the robot obtain walking, object and face recognition, obstacle avoidance, autonomous navigation in crowded environments, recognition with multi mapping and grasping objects (PAL 2015).

REEM-A, REEM-B and REEM are the previous prototypes developed since 2006. Each of them has something different from the other robots of the series that made it special. REEM-A was the lighter of all, weighting 49kg. REEM -B could stand a workload on its hands around 12kg and REEM could reach a maximum speed of 4km/h mainly because it was the only one of the series that the locomotion was based on wheels. Mentionable is that REEM - C is the first humanoid robot developed based 100% on the Robotic Operating System known as ROS.

Figure 2.11 REEM prototypes since 2006-2013 (Rédac’ 2012; Foote 2013; PAL 2015)

One of the biggest manufacturers in Korea in cooperation with the University of Pennsylvania, Purdue University and Virginia Tech manufactured in 2011 a miniature humanoid robot called
Darwin. The humanoid holds the same name as the famous naturalist and geologist of the 18th century who is best known for his contributions to evolutionary theory Charles Robert Darwin (BBC 2014). DARWIN-OP as it is the full name, stands for Dynamic Anthropomorphic Robot with Intelligence–Open Platform and it is a robot with advanced computational power, high payload capacity, dynamic motion abilities and sensors which make it more intelligent and interactive with its surroundings. It has a height of 45.45cm, weights 2.9kg, can reach a max walking speed of about 24cm/s and obtains 20 degrees of freedom. The main purposes that this robot was developed for research and also help people in the fields of humanoid robotics, artificial intelligence, vision, inverse kinematics and linguistics. DARWIN holds the title of champion in the RoboCup of 2011, 2012 and 2013 at the kid size league’s category (Robotis 2013).

![Figure 2.12 Darwin-OP Humanoid Robot (Drakos 2014)](image)

The core of this project that all the applications will be built around it is the NAO robot. NAO is one of the most famous humanoid robots in the scientific world and the reason which created is that Aldebaran wanted to supply the market with a robot that will become the best companion for a human at home. Because of its simplicity, NAO opens the horizons to education for both
amateurs and professionals who wanted to either start programming or develop more complicated things. With the software called NAOqi which is provided by the French company, users can create behaviors or use existing ones to interact with the robot straight away. The humanoid consists of many different sensors and motors, is 58cm tall, weighs about 5.4kg and some of the main functionalities are the ability to dance, move in all dimensions, reply back when you talk to him and take photos (Aldebaran NAO 2012).

One of the latest projects of Aldebaran in cooperation with SoftBank mobile operators is Pepper. The two companies have created a robot which is able to speak properly with humans, recognize them, react in different ways to their emotions, move and live autonomously. The humanoid introduced in 2014 and already can be found interacting with people in many stores of SoftBank in Japan. The robot consists of four microphones, two high definition cameras, located on the head and the mouth, and a 3D depth sensor. Also a gyroscope, six lasers, two sonars and touch sensors in hands and head are hardware parts of Pepper. By its ability to analyse facial expressions and its knowledge of basic emotions such as joy, anger and sadness can translate the emotional state of the human being interacting with it. Pepper also can connect to the internet, find important information when is needed and speak in several languages (Aldebaran
Pepper 2014). On 27th of February Pepper became available to the first 300 developers have applied to get him.

It is very obvious from this brief review on the history of robots that their use is not anymore only for industrial, factory and laboratory environments. They have become a significant part of the society and part of our daily life. As the years come by, and the costs of these robots will be reduced, they will be accessible to more and more people. The number of humanoid robots is not very high at the moment, mainly because there is no demand, reflecting to the high prices. Development of the robots is still at a very primitive level as they don’t even have perfected how to walk and speak yet. The future is very interesting and promising with no one knowing to which levels the human ability and technology can bring the functionalities of humanoid robots.
2.2 Robots Used For Demonstrative Purposes

2.2.1 Robothespian

Robothespian is a humanoid robot that the main purpose of its construction from Engineers Arts in 2009 was to interact with humans in a public environment and used for demonstration. As Will Jackson, the founder of Engineers Art, said “That was the mission. It was to make an automated presenter, something that could stand in a spot all day, every day, and tell the people things, but do it in an entertaining way and use gestures.” (Hickey 2014).

The humanoid robot supports full interaction functionalities, is multilingual and very friendly with its users. In its third generation stage is a very well tested and stable platform that was used in different venues over the years for demonstration and entertaining purposes.

Figure 2.15 Robothespian (Enériz 2013)

Robothespian has an interactive touch screen supplied with a user-friendly interface in which it can supply the user with a list of pre-installed commands and functionalities. Large and visible buttons with graphic icons are designed and when selected can start preset routines straight away. Therefore new movements can be created with simple drag and drop. Users can have direct control of the robot by accessing its camera on top of its head, play short audio clips and choose through a list of facial expressions. The touchscreen interface can run in most of the...
browsers for that reason the robot can be controlled remotely from desktop, laptops or tablets using Adobe Flash. Some of the functionalities that make the robot special are the voice interaction, movements and gesture recognition and Telepresence. With voice interaction the robot replies accordingly to the predefined keywords that the user have set up and get the answers from looking at searching engines such as Wolfram Alpha or Wikipedia.

Also Robothespian can track a number of humans with its gaze, recognize basic gestures and mimic a body pose with its hands. The next stage of development will be the creation of Byrun, a humanoid robot which will weigh 30kg, about 170cm tall and will be able to walk, hop and jump.

2.2.2 Kodomoroid and Ontonaroid

One of the latest creations of professor Hiroshi Ishiguro, who has been creating humanoid robots for years, is Kodomoroid and Ontonaroid. These two humanoid androids have been working as tour guides at the National Museum of Emerging Science and Innovation at Miraikan in Japan since 2014. Ontonaroid which looks like an adult female has as main responsibilities in the museum to continuously tell the news in a different way by using multiple type of voices and languages. Kodomoroid from the other side is a youth female which is able to make a limited range at the moment conversations with the visitors. They both have strange expressive faces and the way they have been designed, gives them a limited functionality to move and communicate like real people. Ontonaroid and Kodomoroid can move their heads, arms and torsos, but stand up and walk around the museum are not available scenarios yet.
Another full-body humanoid robot which was created at the University of Bonn in Germany and can be found in the museum of the town is Robotinho. Robotinho’s main role is to be a tour guide which simultaneously interacts with humans in a multimodal communication. The term multimodal communication is about the ability that the robot has to show interest in multiple persons and shifts its attention between them so that they feel involved in the conversation. The robot focuses on the person who talks and from time to time change the direction of its gaze to involve others in the conversation. That characteristic is that makes this robot special. For having a more anthropomorphic appearance eyebrows, eyelids, mouth and jaw have been added. It has also been equipped with a head of 15 degrees of freedom, two USB cameras in the eyes and a microphone that is used to draw the attention of the person the robot is talking to. The robot performs several natural human gestures and supports a speech recognition system. To provide a safe navigation, inside the museum, to the visitors the robot uses a technology called Fast SLAM (Simultaneous Localization and Mapping) which can acquire maps of the unknown environment. Using these maps and its sensors it can avoid static and dynamic obstacles.

Figure 2.17 Robotinho perform inquiring and greeting gestures (Nieuwenhuisen & Behnke 2013)
2.2.4 REEM-H1

Robot manufacturer, Pal Robotics after the creation of REEMs series have created another humanoid robot named REEM-H1. The robot designed to provide services to the travellers at various tourist attractions. More specifically, this humanoid can be either used as a hotel assistant or as a tour guide at the museum. REEM doesn't have any legs, but its motion is based on wheels instead and cannot produce any kind of facial human-like expressions. A stereophonic camera located in the back of its eyes can recognize familiar faces and also uses a microphone to speak in many different languages. The max speed that the robot can reach is 4km/h and ultrasonic with laser sensors help it to detect obstacles and avoid them (Edwin 2010). One of the strong points of the robot is that its battery can last about 8 hours on a single charge.

![REEM-H1](image)

*Figure 2.18 REEM-H1 (Guide Robot 2010)*

2.2.5 TalkTorque

In Japan and in particular at Tsubuka University in 2010 released Talk Torque, a robot a bit different from the others of its kind. The role of the robot is to guide visitors around a museum showing and explaining everything with gestures without using voice at all. The robot doesn’t have functionalities of speaking or even interacting with the humans at all. Its face is more an alien look like and the only role that has been to act as a tour guide who tries to direct the
attention of the crowd only through its hands. The robot is equipped with three cameras located on its collar and a laser rangefinder to determine a person’s position. However opposite to its head appearance, its arms, hands, and hips look like similar to a human's being.

Figure 2.19 TalkTorque (Plastic parts 2010)
2.3 NAO Robots

One of the most famous humanoid robots in the scientific world distinguished for its simplicity is NAO. The NAO robot was designed in 2006 by Aldebaran Robotics in France. The main purpose that birthed the idea of making a robot like this was the need of creating a daily friendly companion for home use. Since its birth it has been constantly evolving in multiple ways counting five versions that have been developed through these nine years. Its simplicity and its visual appeal can capture the attention of the audience very easily.

2.3.1 Hardware

NAO is 58cm tall, weighs about 5.4kg and has been designed as an all-purpose humanoid robot. The heart of NAO is an Intel ATOM 1.6ghz CPU, which is located in the head and runs a Linux kernel that can support Aldebaran’s proprietary middleware called NAOqi. A second CPU is installed and located in the torso.

The body has 25 degrees of freedom which are working related to electric motors and actuators. The functionality of walking is using a simple dynamic model called linear inverse pendulum combined with quadratic programming. The sensors are responsible to send signals to the joints so the robot can move and stabilize its body. Also NAO has the ability to walk on a different kind of surfaces and move from one type to the other without facing any problems. From the other side, the motion of the body functionality is based on inverse kinematics theories which control the joints, the balance level and prioritize the order that the tasks will occur. Furthermore, there is a protection module which safeguards the robot while a fall is detected. All motions stop instantly, the arms take a protective positioning and the stiffness is reduced to zero. In that way the possibilities of a potential damage are eliminated.

On the sensor’s part two cameras can recognize images and faces. The first camera is located on the forehead and records everything in the sight of NAO and the second one is at the height of mouth responsible to scan the environment around the robot. The camera can capture 30 frames per second up to 1280x960 resolution, which can be transferred easily to a computer when is connected to NAO (Aldebaran 2012).

Four directional microphones are another part of the sensor’s network that makes it feasible for the robot to interact with the humans. All the microphones are located on the head and a have a frequency range of 150Hz to 12kHz. The robot can identify where the sound comes from, by using an approach called “Time Difference of Arrival.” When a source generates a signal wave, then the robot receives that sound in all of the four microphones with a time difference of some
milliseconds between them. These differences calculated in a mathematical way and determine which direction the signal came from. Additionally NAO has a stereo broadcast system made up of 2 loudspeakers in its ears.

![Figure 2.20 NAO H-25 joints and sensor parts (Aldebaran 2012)](image)

On top of its head and on its hands NAO has eight pressure sensors and nine tactile sensors. That lets the developer or the user pass information to the robot only by touching it. There can be many different and complicated approaches which subsequently can give different results. For example, one possible action would be to touch the head of the robot twice, which will have been programmed to react when this happens and take a relative action such as sit down or sit up. More complex functionalities can be implemented as well. LED lights have been installed and can indicate each time the type of contact.(Aldebaran 2012)

Also NAO has four sonar rangefinders which obtain two transmitters and two receivers. That gives the robot the ability to detect and calculate the distances of potential objects and obstacles in its environment in a range of 1 to 3 meters. There is a limitation which doesn't allow the robot to get any information about the distance which is less than 15cm but can only understand the presence of the object instead (Aldebaran 2012).
Noteworthy is that a 48.6-watt-hour battery provides NAO with 1.5 or more hours of autonomy depending on usage.

2.3.2 Software

An embedded software called NAOqi which always runs on the robot, on top of the operating system, is responsible for managing all the robot modules, communications and scheduling. NAOqi Software Development Kit (SDK) is a set of Application Programming Interfaces (APIs) provided in several languages such as C++, Python, Matlab, Java and Urbi that help programmers develop applications for NAO. The NAOqi API is currently available in at least 8 languages and except from some minor differences it is more or less the same in all languages. The NAOqi Framework is the programming framework used to program NAO. The three main characteristics of the framework are the cross platform and the cross language capabilities and also offers introspection. With the term cross platform all the range of Operating Systems are covered in which code can be developed for the NAO. Software in the terms of this project will be developed in Java and Python, and the cross language feature means that programming methods are exactly the same in both programming languages. Also with introspection the framework identifies which functions are available in the different modules.
2.4 Applications with NAO

In information technology, as the application can be described as a computer program which is designed to assist people perform an activity or a task when interacting with a computer machine. More specifically, an absolutely well organized set of instructions that, when executed, define the computer machine’s behavior in a predetermined way. All the applications have been developed exclusively for the end user’s ease. Users have the option to communicate with the computer machine directly through the applications’ capabilities. Software applications can be developed with different approaches either in a desktop or on Android platforms. NAO has already applications on both platforms which offer the user the chance to handle the robot in different ways, learn from the robot and play with it. The embedded software called NAOqi which always run inside the robot manages all the robot modules, communications and scheduling.

2.4.1 Desktop applications

2.4.1.1 Choregraphe

Choregraphe is the programming software environment developed from Aldebaran, which provides a user friendly interface to everyone who wants to have a full control of the robot’s functionalities. A very well structured and documented software development tool kit (SDK) is available to everyone and allows all kinds of users interact and create behaviors for the robot. Beginners, without the need of having any programming skills, just by connecting some predefined modules and experienced programmers by writing their own code can interact directly with the robot in different aspects. The graphical interface, the existing libraries of standard behaviors, and the advanced programming scenarios meet the needs of beginners and experts (Aldebaran Choregraphe 2014).

The development kit contains all the necessary compilers and debuggers which through programming code give functionality to the robot. The software is compatible with most of the famous high level languages and robotics platforms, including Java, JavaScript, C++, Python, MATLAB, Urbi and.Net. The concept of the application for someone that doesn’t want or even doesn’t know how to program at all has been based on creating sequences of predefined behavior boxes and just simulating the application. Anyone can create behaviors simply by dragging and dropping actions from the library on the left hand side of the interface or by using
the customized boxes and save them to custom libraries. Choreograph allows the user to program in temporal logic according to the sequence and combinations that put the boxes. The predefined behaviors can be modified easily just by changing some values in the boxes, but in case someone wants to create his own can implement it by using the Curve Editor to edit movements or write them in Python scripts. All these different ways to handle NAO open up a huge world for everyone that either prefers to start learning and developing his programming skills or go deeper in programming and develop his own NAO behaviors.

![Figure 2.21 Choregraphe interface demonstrating a sequence of connected behaviors (Aragon 2013)](image)

Along with the Choregraphe suite there is an additional tool called Monitor which gives the user valuable basic feedback from the robot actions and allows access to the camera. Three plugins are available and these are, camera viewer, memory viewer and laser monitor. The camera viewer provides functions to the user, such as taking pictures, recording videos and display in a panel what the robot sees in real time. A very important feature of the camera viewer is that teaches NAO to recognize objects. The second plugin allows users to collect and analyse statistics from the data held in the memory of the robot. This is a very useful tool because makes it feasible to diagnose behaviors through the data being used. The last feature is only available for NAO Laser Head model and allows mapping of what is recorded by the laser telemeter. The monitor is very useful for debugging applications developed either in Choregraphe or using the SDK (Aldebaran Monitor 2014).
2.4.1.2 Webots for NAO

Webots for NAO is an application platform which provides a virtual world that can simulate behaviors and actions that NAO could execute in the real world. In this environment all of the behaviors that the robot could have can be tested more safely and accurately.

On November of 2013 Julien Fremy, one of the engineers of Aldebaran presented the platform and described it as a tool that you can program NAO without NAO. A bit oxymoron description, but actually he wanted to focus that the platform could simulate multi-robots and interact with them in a 3D editable world without using NAO Robots as feedstock.

Two worlds are available for the simulation an empty space and an apartment. The application is supported on multiple operating systems (Windows, Linux, and Mac) in six different languages(Aldebaran Webots 2014).
2.4.1.3 NAO MATLAB API

The MATLAB NAO application programming interface can deal with the humanoid robot through MATLAB and some related boxes. Users can get related data from sensors and control all the actuators of the robot. MATLAB establishes a wireless or a wired connection, support both types (Ethernet, Wi-Fi), between the robot and the personal computer and asynchronously sends and receives data. It is a straightforward procedure for programming NAO, setting up an interactive communication and debugging. By installing and using additional toolboxes in the application users can expand the capabilities and develop programs according to the signals received from the sensors and the signals send to the actuators.

All the sensors of the robot like the microphone, the camera, the accelerometer, and actuators such as motors, speakers and LEDs can be controlled and used by the platform. An example that describes the functionality of the API is elaborating data from images taken from the camera of the robot. By using face detection algorithms from Computer Vision System toolbox, the API can detect faces compared to the images received. Then MATLAB can send a voice message or any kind of command, such as the name of the person, to the speakers on NAO or give the robot the command to walk towards or away from a particular person.
As smartphones are becoming increasingly popular all around the world over the last few years is a natural consequence that more and more applications would be developed to support these phones and make them, except from competitive in the market but very utilitarian at the same time. NAO could not be an exception and since the point came to the market some mobile applications have created for that purpose. Smartphones, tablet computers and other mobile devices support proper software, giving the opportunity to the user to interact with the NAO humanoid robot. All the applications that would be described can be found either in the robot App Store or in the Google Play Store.

### 2.4.2.1 NAO Robot Controller

One of the applications that has been developed the last years by a team of three students of the Saxion University, Robin Bonnes, Daniel Elsing and Aron Faasia is NAO Robot Controller. (Bonnes 2013). The application controls and deals with several functionalities of the NAO and provides a very user-friendly interface.
Initially the user has two options of how to establish a communication link between the mobile device and the robot. The first option is to insert the IP manually in the browser to make the robot identifiable to the device and through the device settings find the robot and select it. The second way is to connect the NAO and the mobile phone to the local network and then from the tab settings insert the IP of the robot manually again. Also the user can check the battery status and the temperature of the CPU through some graphics. From other tabs the user can choose to take photos by using the camera and move the robot in all directions from pressing arrows dealing with the movements. Extra features are to make the robot sit down, stand up, dance, sleep, speak and do pushups. Last but not least user can manage existent behaviors or even create new ones (NAO Robot Controller 2013).
2.4.2.2 NAO Communicator

The NAO Communicator is a combination of two projects which are responsible to control NAO wirelessly when used in an Android application. It is based on a Python server and a Java Android application that sends commands to the server. More specifically, NAO Communication Server runs in the background of NAO and receives commands from the NAO Com application. Most of the functionalities and modules of the robot are stored on the server and each time a command is received from Android then executed relatively. One of the most important features is that the application can send data to the memory of the robot and interact with Choregraphe. Some of the possible modules that the robot can execute are speech control by allowing the user to define his own custom text, walking, shake hands, stand up, sit down, many animations including different type of dances and led animations related to feelings such as showing the robot, happy, angry or thinking (NorthernStars 2014).

![NAO Communicator Application](image)

*Figure 2.27 NAO Communicator Application (NorthernStars 2014)*
2.4.2.3 Full Manage NAO

Full Manage NAO, is one Android application developed from Gerashchenko Ilya Nikolayevich in 2013. One of the disadvantages of the application is that it is written in Russian which makes it very difficult to use for users who don’t know this language. The main functionalities that the application supports, is the management of behaviors, detection of the robot in the network automatically, having the robot generates predefined phrases both in Russian and English, moving the robot in all axis through commands send by a joystick control and turning on or off the stiffness of the joints. Finally the user can check CPU’s and motor sensor’s temperature at any time.

![Image of Full Manage NAO](image)

*Figure 2.28 Full Manage NAO (Full Manage NAO 2013)*

2.4.2.4 NAO Control

In 2012 another IOS application developed from Tommy Kammerer. The application called NAO Control and some of the basic behaviors of the robot have been installed in the suite. The languages supported are English, French and German. The communication with the robot can be established directly without the need of a server as it used to be in earlier versions. By handling a joystick the user can move the robot in all directions of space and another tab gives the opportunity to manage existing behaviors or even install new ones. Some of the already installed functionalities allow the robot to sit down, stand up, take photos and speak by using specific messages.
2.4.2.5 iControlNAO

iControlNAO is another application which gives the user control over NAO and was created by Klaus Engel on 2012. iControlNAO detects all the NAO robots in the area and uses Bonjour to connect automatically to the robot. In the app documentation it is implied that it can be used with the NAO simulation software as well.

The application consists of four tabs. From the first one the user can browse through a list of pre-installed behaviors choose one and the robot will execute it straight away. The next one offers functionalities such as movement (front, backwards, right and left) and by typing any text will make the robot start speaking. The third one is an information tab which shows SDK documentation and the API stored in NAO and the last one handles some basic settings which can be changed preferably.

![Figure 2.29 NAO Control application (NAO Control 2013)](image1)

![Figure 2.30 iControlNAO platform (IControlNAO 2012)](image2)
2.4.2.6 NAO Remote Control

This Android application needs two different components to work properly. The NAO Remote Control Android application and the NAO Server Desktop application. These two and the robot have to be connected to the same network so a communication would be established to allow the user to interact with the robot wirelessly. Although there is the option to connect them in different networks, but this demands change the router’s settings. Initially the user has to insert manually the IP of the robot to the desktop app and then can detect the humanoid from the setting of the Android application (NAO Remote 2012). From the description it is obvious that there is a middle machine needed and there is not the option of direct communication between the robot and the application. Additionally the functionalities of the robot are limited because it can only execute back and forward movements, stand up and sit down.

![NAO Remote Control](image)

*Figure 2.31 NAO Remote Control (NAO Remote 2012)*

2.4.2.7 NAO Remote

NAO Remote is another application started in October of 2014 with the purpose to wirelessly control the NAO Robot. However, it’s still in development stage and the server that will support the robot hasn’t been developed yet.
3 ORGANIZATION

3.1 Requirements Analysis

The requirement analysis has as main target to clearly state all the essentials that has to be taken into account, before the implementation starts. The table below describes all the important requirements, the level of the priority that they have during the development and the predecessors that needed for each phase. Also a brief description given about how each stage will be faced.

<table>
<thead>
<tr>
<th>A/A</th>
<th>Requirements</th>
<th>Priority</th>
<th>Predecessors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Understand NAO architecture</td>
<td>High</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>Develop the desktop graphical user interface</td>
<td>High</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>Connect NAO with desktop application via WI-FI</td>
<td>High</td>
<td>1,2</td>
</tr>
<tr>
<td>4</td>
<td>Develop the Android graphical user interface</td>
<td>High</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>Establish communication between Android app and NAO robot</td>
<td>High</td>
<td>1,4</td>
</tr>
<tr>
<td>6</td>
<td>Integrate motion module on both apps</td>
<td>High</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>7</td>
<td>Integrate speech recognition module on both apps</td>
<td>High</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>8</td>
<td>Environment of demonstration portable</td>
<td>Low</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>Interactive demonstration</td>
<td>High</td>
<td>1,2,3,4,5,7</td>
</tr>
<tr>
<td>10</td>
<td>Evaluation from users after end of demonstration</td>
<td>Medium</td>
<td>2,3,4,5,6,7</td>
</tr>
<tr>
<td>11</td>
<td>Use of applications for lab purposes and exhibitions</td>
<td>High</td>
<td>-</td>
</tr>
</tbody>
</table>

*Table 3.1 Project Requirements*
Requirement 1: The foundations of the projects will be based on the full understanding of the architecture of the NAO robot and all its special characteristics. For the accomplishment of this objective the relevant documentation, which is supported from Aldebaran, will be thoroughly studied initially and then a series of primary tests will be executed. The tests will help to familiarize myself with the robot and its functionalities.

Requirement 2: The second objective will focus on the control of NAO by the desktop application. This requirement will be achieved by designing and developing a graphical user interface. The interface will give the opportunity to the user interact with the robot in multiple ways and allow the establishment of a proper communication between them. The graphical user interface will combine efficiency and at the same time a user friendly environment. Python will be the language that will be used to build up the application.

Requirement 3: The next objective is the establishment of a proper communication between the robot and the desktop application which will allow the user interact through the interface directly with the robot. The communication won't use any middleware server, but each application will be connected straight to NAO.

Requirement 4: The fourth requirement will have the same conformation with the second requirement. A user friendly graphical interface will be developed for mobile devices which again will provide the user with the option to interact with the robot directly. The language that the interface will be developed will be Java.

Requirement 5: The establishment of a proper communication between the robot and the Android application would be the next target. The communication won't use any middleware server, but the application will be connected straight to NAO. In the case of the Android application has been taken into account, that if the existing version of NAO is not compatible with NAOqi libraries which support direct communication with the robot a middleware server will be used.

Requirement 6: After the development of the desktop’s and Android’s interface the next step will be the motion modules of the robot to be embedded on the functionality of the application. The move module will make the robot able to stand up, sit down and move in all directions of the
space regarding the received commands. The code that the motion module will be written is Python for the desktop app and Java for the Android one.

Requirement 7: Another module that the robot will be able to execute is the speech interaction. The main idea of this functionality will focus on keywords. The robot after ‘understanding’ what the users ask will execute a relevant command. The user will have the option to choose from a list of predefined keywords and speak to the robot which in turn will perform specific tasks related to the keywords. The robot can reply and providing information about the lab, the current research projects and other subjects related to the laboratory. The code for this module will be written in Python for the Android and in Java for the desktop application.

Requirement 8: One of the requirements that have to be fulfilled at the end is about the flexibility that the robot can show on performing the tasks in different type of surfaces. The area will always bounded during the tests. NAO has to be independent and can properly execute all the modules that have been programmed to. The surfaces would include the floor, and the table, but in both cases a safe environment will be provided to keep the robot secure in case of possible falls.

Requirement 9: After both applications will be finalized and tested a demonstration will take place in the robotics lab at Heriot Watt University. The purpose of this exhibition is to present the robot and the applications to a number of people, which will be able to interact with NAO. The focus would be the user not only to handle the robot, but to interact with NAO as well. This is the reason that the speech recognition module will be involved to the application’s functionalities.

Requirement 10: The next stage after the demonstration of all the functionalities of the robot and the time that the visitors will spend interacting with the robot would be the evaluation. Questionnaires will have been prepared and given to the visitors so they can express their opinion for the project and also make their suggestions about what they would like to see in the future included in these applications. Through a series of different questions the evaluation will target to gather as much information as possible in order the weaknesses and the strengths to be revealed and also to help other developers to make improvements in the next versions.
Requirement 11: One of the project’s targets will be to provide the Robotics lab at Heriot Watt with an application which can handle the robot and use it as a lab guide for everyone who visits the lab. The robot initially will have a number of functionalities which would be executed by the interface of the applications, but at the same time will be available to be upgraded and more functionalities to be added in the future.
3.1 Performance Assessment

In the terms of the performance assessment all the prototypes will be evaluated separately. A plan that is described in the table below has been set up including specific dates and different ways that the evaluation will be executed. After the completion of each prototype a mini demonstration will take place which will prove that all the targets of each stage have been achieved. Performance assessment is very crucial and also can guarantee that there is a good progress for reaching the final objective which is none other than the accomplishment of the project.

<table>
<thead>
<tr>
<th>Objective</th>
<th>Due Date</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prototype 1</strong>: Familiarization by testing basic functions of NAO Robot</td>
<td>09/03/15</td>
<td>Full understanding of NAO’s architecture</td>
</tr>
<tr>
<td><strong>Prototype 2</strong>: Developing desktop’s application interface</td>
<td>16/04/15</td>
<td>Demonstrate graphical user interface</td>
</tr>
<tr>
<td><strong>Prototype 3</strong>: Robot moves through commands from desktop app</td>
<td>13/05/15</td>
<td>Demonstration of robot moving, stand up and sit down</td>
</tr>
<tr>
<td><strong>Prototype 4</strong>: Robot interacts according predefined keywords</td>
<td>25/05/15</td>
<td>Demonstrate robot interacting with user after receiving voice commands</td>
</tr>
<tr>
<td><strong>Prototype 5</strong>: Developing Android application</td>
<td>17/06/15</td>
<td>Demonstrate graphical user interface</td>
</tr>
<tr>
<td><strong>Prototype 6</strong>: Establishing communication between robot and Android application</td>
<td>28/06/15</td>
<td>Assess communication quality through testing basic functions of NAO</td>
</tr>
<tr>
<td><strong>Prototype 7</strong>: Robot moves through commands from Android app</td>
<td>05/07/15</td>
<td>Demonstration of robot moving, stand up and sit down through app</td>
</tr>
<tr>
<td><strong>Prototype 8</strong>: Evaluation of prototype 4 by users</td>
<td>20/07/15</td>
<td>Evaluation of questionnaires</td>
</tr>
</tbody>
</table>

Table 3.2 Performance Assessment
The first prototype would include a series of tests that will have as main target the familiarization with the architecture of the robot. After these tests the ability to understand basic scenarios and the way that NAO can execute some of its functionalities will be assessed.

The second prototype will include the designing and the development of a full user graphical interface for the desktop application. After completion will be assessed for its functionality and how user friendly is. Suggestions will be proposed for any improvements.

The third prototype it will demonstrate the robot move, stand up and sit down after getting commands from the application on the desktop.

The fourth prototype will develop the speech interaction module. A demonstration of the robot interacting with predefined words will be assessed. A series of different words will be tested and the option that the user will have to insert and delete keywords in the existing list would be assessed as well.

The fifth prototype will present the application’s interface of Android. The full design and development of the user graphical interface will be demonstrated to the supervisor. Potential alterations having as a purpose to improve the application will be discussed.

The sixth prototype will check if the connection has been established between the robot and the application is working properly. Some basic functionality will be tested to verify the communication among NAO and Android.

The seventh prototype will support the move module which would be executed through the Android application. A series of tests will check the efficiency and demonstrate that the robot can move, sit down and stand up after receiving the relevant commands from the mobile device.

The last prototype will be the evaluation stage of the desktop application. The assessment of this prototype will be done after the completion of the questionnaires that will be provided to the visitors and evaluate the project in total.
### 3.2 Risk Assessment

The table describes the risks that will be taken into account from the beginning until the end of the project.

<table>
<thead>
<tr>
<th>A/A</th>
<th>Risk</th>
<th>Possibility to happen</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Computer machine failure</td>
<td>Low</td>
<td>Another laptop or pc will replace the faulty one</td>
</tr>
<tr>
<td>2</td>
<td>NAO hardware failure</td>
<td>Low</td>
<td>A second alternative NAO will be used</td>
</tr>
<tr>
<td>3</td>
<td>Difficulties with NAO programming</td>
<td>Low</td>
<td>Spend more time to learn required techniques and architecture of NAO</td>
</tr>
<tr>
<td>4</td>
<td>Difficulties developing Desktop app</td>
<td>Medium</td>
<td>Spend more time on studying python language</td>
</tr>
<tr>
<td>5</td>
<td>Difficulties developing Android app</td>
<td>High</td>
<td>Spend more time on studying and ask advice for colleagues with relevant experience on Android apps</td>
</tr>
<tr>
<td>6</td>
<td>NAOqi library for Android compatible with NAO software version 1.14.5</td>
<td>High</td>
<td>A server will be used as it has already tested in other Android applications that are not compatible with the existent version of NAO</td>
</tr>
</tbody>
</table>

*Table 3.3 Project Risks and Responses*

1) The first risk that has been outlined is the potential of having a computer failure. If that happens another laptop or desktop computer will be used to replace the faulty one. In the meantime backups will be taken regularly in many different other machines, so any progress that has been made, is not wasted.
2) As all electronics, NAO also has the chances to face hardware failures. If that is the case another NAO robot that already exists on the premises of the university will be used as long as the first one is repaired.

3) Because this is my first project that a NAO robot is included, maybe I will struggle to understand from the beginning all the architecture that the robot is based on. Taking into account that this will happen the project plan has been set up in a way that enough time will be spent on studying the software and hardware capabilities of the robot.

4) The fourth risk that this project will face is when the stage of developing the desktop application will be reached. As I have enough experience in developing graphical user interfaces before the beginning of this project the level of the risk has been set to medium. However, in case those difficulties may arise, more time will be invested around GUIs development in python and advice will be sought by colleagues who have relevant experience.

5) The risk of facing difficulties with developing the Android application has many similarities with the previous risk described. The main difference is that because there is a lack of experience with the Android applications maybe it would be, compared to the desktop app, more challenging and difficult task. This is the reason that this risk’s level has been set to high. Valuable will be the help of my colleagues that have already knowledge about the subject and obviously more time will be spent on studying and learning Android applications.

6) Due to writing this dissertation and start learning about the existent Android applications I discovered that the NAOqi libraries on the version of 1.14.5 are not compatible with the Android apps. At the moment, the robots of the lab that are available for the accomplishment of this project support the version 1.14.5 and cannot be upgraded to the next one which is compatible. If there is no other solution then a middleware server will be used, as has been already done and tested in other wireless Android applications.
3.3 Project Plan

The project plan shows all the tasks that have to be accomplished and how the time is going to be organised around them. In each of the tasks, has assigned a specific period of time within which the task has to be completed.

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Duration</th>
<th>Start</th>
<th>Finish</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>30 days</td>
<td>Mon 02-02-15</td>
<td>Tue 03-03-15</td>
</tr>
<tr>
<td>Project title proposals</td>
<td>7 days</td>
<td>Mon 02-02-15</td>
<td>Sun 08-02-15</td>
</tr>
<tr>
<td>Present project idea to supervisor</td>
<td>1 day</td>
<td>Mon 09-02-15</td>
<td>Mon 09-02-15</td>
</tr>
<tr>
<td>Confirmation from supervisor</td>
<td>7 days</td>
<td>Tue 10-02-15</td>
<td>Mon 16-02-15</td>
</tr>
<tr>
<td>Designing and sending a formal working plan</td>
<td>10 days</td>
<td>Tue 17-02-15</td>
<td>Thu 26-02-15</td>
</tr>
<tr>
<td>Approval of plan</td>
<td>5 days</td>
<td>Fri 27-02-15</td>
<td>Tue 03-03-15</td>
</tr>
<tr>
<td><strong>Familiarization</strong></td>
<td>13 days</td>
<td>Wed 04-03-15</td>
<td>Mon 16-03-15</td>
</tr>
<tr>
<td>Introduction to hardware parts of NAO Robots</td>
<td>2 days</td>
<td>Wed 04-03-15</td>
<td>Thu 05-03-15</td>
</tr>
<tr>
<td>Read manuals and software features</td>
<td>3 days</td>
<td>Fri 06-03-15</td>
<td>Sun 08-03-15</td>
</tr>
<tr>
<td>Execute simple examples of programming lab robot</td>
<td>8 days</td>
<td>Mon 09-03-15</td>
<td>Mon 16-03-15</td>
</tr>
<tr>
<td><strong>Desktop Application</strong></td>
<td>41 days</td>
<td>Sun 01-03-15</td>
<td>Tue 26-05-15</td>
</tr>
<tr>
<td>Designing GUI</td>
<td>6 days</td>
<td>Sun 17-03-15</td>
<td>Sun 22-03-15</td>
</tr>
<tr>
<td>Design desktop application interface</td>
<td>5 days</td>
<td>Tue 17-03-15</td>
<td>Sat 21-03-15</td>
</tr>
<tr>
<td>Presentation of GUI design</td>
<td>1 day</td>
<td>Sun 22-03-15</td>
<td>Sun 22-03-15</td>
</tr>
<tr>
<td>Submitting Draft of Technical Report</td>
<td>17 days</td>
<td>Sun 01-03-15</td>
<td>Wed 18-03-15</td>
</tr>
<tr>
<td><strong>Developing GUI</strong></td>
<td>15 days</td>
<td>Wed 01-04-15</td>
<td>Thu 14-05-15</td>
</tr>
<tr>
<td>Develop GUI interface in Python</td>
<td>15 days</td>
<td>Wed 01-04-15</td>
<td>Thu 16-04-15</td>
</tr>
<tr>
<td>Completing/Submitting Technical Report</td>
<td>33 days</td>
<td>Sun 01-03-15</td>
<td>Fri 03-04-15</td>
</tr>
<tr>
<td>Exams 2nd Semester</td>
<td>20 days</td>
<td>Thu 16-04-15</td>
<td>Wed 06-05-15</td>
</tr>
<tr>
<td>Programming modules of the robot</td>
<td>20 days</td>
<td>Wed 06-05-15</td>
<td>Tue 26-05-15</td>
</tr>
<tr>
<td>Programming robot moves according the commands</td>
<td>7 days</td>
<td>Wed 06-05-15</td>
<td>Wed 13-05-15</td>
</tr>
<tr>
<td>Task</td>
<td>Duration</td>
<td>Start Date</td>
<td>End Date</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>----------</td>
<td>---------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Programming robot sits/stands up</td>
<td>2 days</td>
<td>Wed 13-05-15</td>
<td>Fri 15-05-15</td>
</tr>
<tr>
<td>Programming robot's behaviors according to predefined keywords</td>
<td>10 days</td>
<td>Fri 15-05-15</td>
<td>Mon 25-05-15</td>
</tr>
<tr>
<td>Exhibit application to the supervisor</td>
<td>1 day</td>
<td>Mon 25-05-15</td>
<td>Tue 26-05-15</td>
</tr>
<tr>
<td><strong>Android Application</strong></td>
<td>45 days</td>
<td>Tue 26-05-15</td>
<td>Fri 10-07-15</td>
</tr>
<tr>
<td><strong>Designing Android app</strong></td>
<td>6 days</td>
<td>Tue 26-05-15</td>
<td>Mon 01-06-15</td>
</tr>
<tr>
<td>Designing of the Android application</td>
<td>5 days</td>
<td>Tue 26-05-15</td>
<td>Sun 31-05-15</td>
</tr>
<tr>
<td>Present it to the supervisor for confirmation</td>
<td>1 day</td>
<td>Sun 31-05-15</td>
<td>Mon 01-06-15</td>
</tr>
<tr>
<td><strong>Developing an Android app</strong></td>
<td>34 days</td>
<td>Mon 01-06-15</td>
<td>Sun 05-07-15</td>
</tr>
<tr>
<td>Develop Android's application interface in Java</td>
<td>16 days</td>
<td>Mon 01-06-15</td>
<td>Wed 17-06-15</td>
</tr>
<tr>
<td>Establish a connection between the robot and Android</td>
<td>11 days</td>
<td>Wed 17-06-15</td>
<td>Sun 28-06-15</td>
</tr>
<tr>
<td>Embedded functions of motion and speech recognition to Android</td>
<td>7 days</td>
<td>Sun 28-06-15</td>
<td>Sun 05-07-15</td>
</tr>
<tr>
<td><strong>Testing / Re-Assess</strong></td>
<td>5 days</td>
<td>Sun 05-07-15</td>
<td>Fri 10-07-15</td>
</tr>
<tr>
<td>Finalize desktop and Android applications / Re-assess with the supervisor</td>
<td>5 days</td>
<td>Sun 05-07-15</td>
<td>Fri 10-07-15</td>
</tr>
<tr>
<td><strong>Evaluation/Submit</strong></td>
<td>40 days</td>
<td>Fri 10-07-15</td>
<td>Wed 19-08-15</td>
</tr>
<tr>
<td>Evaluation of demonstration</td>
<td>2 days</td>
<td>Fri 10-07-15</td>
<td>Sun 12-07-15</td>
</tr>
<tr>
<td>Questionnaire Analysis</td>
<td>8 days</td>
<td>Sun 12-07-15</td>
<td>Mon 20-07-15</td>
</tr>
<tr>
<td><strong>Submit dissertation</strong></td>
<td>30 days</td>
<td>Mon 20-07-15</td>
<td>Wed 19-08-15</td>
</tr>
<tr>
<td>Send draft to supervisor</td>
<td>15 days</td>
<td>Mon 20-07-15</td>
<td>Tue 04-08-15</td>
</tr>
<tr>
<td>Finalizing dissertation writing/Submit</td>
<td>15 days</td>
<td>Tue 04-08-15</td>
<td>Wed 19-08-15</td>
</tr>
<tr>
<td><strong>Poster</strong></td>
<td>9 days</td>
<td>Wed 19-08-15</td>
<td>Fri 28-08-15</td>
</tr>
<tr>
<td>Poster design</td>
<td>7 days</td>
<td>Wed 19-08-15</td>
<td>Wed 26-08-15</td>
</tr>
<tr>
<td>Poster presentation</td>
<td>1 day</td>
<td>Wed 26-08-15</td>
<td>Thu 27-08-15</td>
</tr>
<tr>
<td>Project Ends</td>
<td>1 day</td>
<td>Thu 27-08-15</td>
<td>Fri 28-08-15</td>
</tr>
</tbody>
</table>

**Table 3.4 Project plan tasks**
Figure 3.1 Project Plan chart
4 METHODS AND RESULTS

4.1 Prototype 1 –Familiarization by testing basic functions of NAO robots

4.1.1 Design

The first prototype of the dissertation mainly focused on providing a first approach and an interaction with the robot by using already existing resources. The purpose of designing and having a prototype like that in the first place was clearly the familiarization with everything is related to the NAO robot. The path that has been followed was based on the achievement of some specific goals that have been set up from the beginning. Initially by reading all the documentation related to the robot and then by creating and executing simple tasks through basic functions such as speaking or moving around within the space.

4.1.2 Implementation

The implementation of the first prototype was not literally a development stage of the application, including programming or designing but basically was about learning and fully understanding of the fundamentals that the robot is built on and is used on the background to support different functionalities. At the beginning, the first days spent on reading the documentation and covering subjects such as the libraries used, how the software development kit (SDK) interacts with the code, and in general which and how languages are used to program the humanoid robot NAO. The two main tools that used for the task of understanding the basic functions of the robot was the Choreograph software (Aldebaran Choregraphe 2014) provided from Aldebaran and an iPhone application called iControlNAO (iControlNAO 2012)

After experiments with the two existent applications, some code was written in Python to test again some very simple behaviors of the robot. As mentioned above, the first attempt, after reading the documentation was the direct interaction with Choreograph and the creation of some basic functions that the robot could execute and demonstrate. The environment that the software provided was separated into three panels. On the left side there was a list with all the potential functionalities, in the center another bigger panel, which the user could combine these behaviors and the last on the right in which a virtual robot could demonstrate each action the robot has been commanded to execute. The environment with the very first basic behavior that has been created is shown below. In this task, the robot could express a welcome message and
simultaneously raise its hand and greet. After the first simple example, many different ones followed, which was either a combination of different movements or only standalone behaviors.

Figure 4.1 NAO executing a simple behavior in Choreograph environment

The second approach was based on the iPhone application iControlNAO and could provide similar functionalities like the Choreograph suite did. Because in this application, there was not a virtual robot that could demonstrate the actions that have been selected for the robot, a direct connection with a real robot have been set it up for the first time. For the task, because the application could not provide a direct connection between the robot and the software, a wireless access point used as a middleware agent to establish a WI-FI network and a proper communication. Both robot and the application connected to the network through the modem and a communication started between them started instantly. After the stage that a connection has been set up, the rest was an easy procedure as everything was ready and only the press of a button could make NAO proceed with the desirable action.

After experimenting with both the desktop and the mobile application the third step on this prototype was the development of some small snippets of code in Python that will allow the robot to proceed with very simple tasks as the ones tested with the applications. Two very small pieces of code have been written for testing and familiarization with the language. The first one made no speak and express an introductory message and the second one was able to make the robot
stand up and in the meantime express a relevant message that has to do with the direction of the movement. The two snippets are shown and explained in detail.

```python
from NAOqi import ALProxy
tts = ALProxy("ALTextToSpeech", "137.195.108.91", 9559)
tts.say("Hello everyone. How are you?")
```

By these three lines of code NAO can speak. In the first line the ALProxy module from the NAOqi API is imported. ALProxy is responsible to give full access to the attributes and the methods of the module which will be created. In the second line the ALTextToSpeech module is created and the proper IP address with the port number is declared. Finally the third line selects the desirable message.

```python
from NAOqi import ALProxy
tts = ALProxy("ALTextToSpeech", "137.195.108.91", 9559)
motion = ALProxy("ALTextToMotion", "137.195.108.91", 9559)
motion.setStiffnesses("Body", "1.0")
motion.moveTo("Stand")
tts.say("I am standing up")
```

Similar to the first example, which explained thoroughly, in this test another module (ALTextToMotion) have been created and the stiffness of the joints has been activated in order to help the robot proceed with a movement action. Then the robot moves and produces a voice message.
4.1.3 Evaluation

At the end of this prototype there is only positive feedback, mainly because there was a question at the beginning if it had to be exist as a separate prototype or excluded completely from this thesis. The introduction to the technologies of NAO became very smoothly and the procedure that followed step by step I believe helped a lot to understand in a more easy and convenient way how can the robot be programmed according to simple tasks that will become more complex as the time goes by.

The first approach that had to do with the Choreograph suite was helpful as the code could be extracted and studied separately, line by line, something that provided a better and more spherical understanding of the steps that have to be followed. Also the virtual robot that exists in the platform could display the results of the selected behaviors without having a real robot to interact with, from the beginning of the testing. Something that will be very useful in the future and much safer, mainly because some new custom behaviors possibly have to be tested, first in the simulation and not directly in the real robot.

The second approach of the mobile application was not that helpful eventually as the only positive result that can be counted was the use of the wireless access point. The need that has been generated for the establishment of a proper communication between the robot and the application maybe would be used in the first steps of developing the Android application. In that case, which a direct connection cannot be set up, the already known approach will be proved, in the near future, very helpful. The experience and the knowledge of using that, will not allow the time to be wasted and a similar situation can be faced faster.

The last stage of this first prototype was the development of the code, which was written in Python and could make the robot able to execute some simple tasks. This was proved the most important part as the familiarization of writing lines of code in Python according to actions related to the robot was closer to the application that will be developed later on. However, the examples were very simple it was a good induction of how NAO can be programmed.
4.2 Prototype 2 – Developing desktop’s application interface

4.2.1 Design

The second prototype that has been set up was responsible for the designing and development of a graphical user interface having as a base the Python language. The main toolkit that was the fundamental of developing the interface was Tkinter. Tkinter is a thin object-oriented layer which has been set for Python on top of Tcl/Tk. The first thing before the designing part, was to become clear from the beginning what type of information and functionalities the interface could offer to the user. After discussions with the supervisor and co-supervisor, have been decided what exactly the robot could support and an initial plan designed. This plan could be the base of the GUI’s development stage. The three main functionalities of the robot that would be used and become the keystones, in which the application would be built around, would be the motion, the speech and the speech recognition. The diagram below shows what was planned to be included in the application and the relationships that will be arise during the use of each class elements.

![Schema diagram showing all the functionalities that the interface can support](image)

*Figure 4.2 Schema diagram showing all the functionalities that the interface can support*
4.2.2 Implementation

After knowing the requirements which the system and the application would include and having a design on the paper the implementation stage followed. The first dilemma, which came up at the beginning of the development, was how the sub-panels would be opened through the main panel. The common element in both options would be to have a main window providing buttons to the user in order to access each different task every time a button is clicked. The first option that has been considered was that the new functionalities would open in a new pop-up window and the second option would be the creation of many different frames, inside the main panel, which will handle the swap between each other every time the user clicks the relevant button. Eventually, after both suggestions tried, the decision that has been taken was the option of opening the new functionalities in a new window as taken into consideration that this would be the easiest way for the users navigate through the functionalities. The next step was the development of the main panel. During the implementation, the main panel appearance has passed from three different stages until its final look. At the first attempt there was a window title, a sub-header displaying a general message about what the main panel does and the buttons. At the second try of changing the main window, has been added a top bar including the name of the robot and details about the battery. Also an image has been placed on the right hand side of the window. At the third and last modification, some of the widgets have just changed position and a label displaying the IP address of the robot has been added to the bottom right corner of the window. The menu on the upper left has been removed, the fonts of the labels, the background and the foreground of the buttons has been changed.

![Figure 4.3 Main Panel of the application](image-url)
The designing and development of the main window was the fundamental of how the rest of the windows will be developed and look like. Six separate windows have been created each one serving a different functionality. The first one was the motion window which has been separated into two different sections, one including keyboard commands and one based on voice commands. The section related to the keyboard commands consisted of nine buttons, each one providing a special movement in a different direction of the space. These movements allow the robot move along the four axises, turn right and left, sit down, stand up and go to the crouch position. The voice command’s part includes two buttons which are responsible to start and stop a communication with the robot. Also a list box has been added on the right hand side in order to hold a log history of the movements coming from the keyboard or by voice messages from the user. This window during the development stage has passed from three versions, similar to the main panel, before reach its final appearance.

![Image of the Motion window](image)

*Figure 4.4 The Motion window*

The speech window included everything that was related to the functionalities of the robot that allowed it to speak. The window was separated into two sections which could provide to the user the option of writing each own text and pass it to the robot or choose from a list of predefined phrases and make the robot speak accordingly as well. The left hand side is the part of the manual entries which consists of four buttons, two entry texts and two labels. On the upper entry
text, users can write a custom text of his preference and the other entry text holds a history of the phrases that the robot produces either the ones written manually or the ones from the list. The button brings the name clear text is responsible to delete any text from the entry text next to it and the speak button to pass the message to the robot in order to produce a voice message.

Furthermore, the add phrase button adds anything is inside the entry text to the list on the right and saves it automatically. The button at the bottom, called Clear Log, clears all the contents of the history log. On the right hand side the panel consists of two buttons, one list and two labels. The “NAO speak” button is responsible to pass the selected phrase of the list to the robot in order to produce voice messages. The other button of the panel called delete entry removes the selected phrase from the list by preference. The entry text in the middle holds and displays a list of phrases defined by the user. The picture below shows the speech window and its elements.

![The Speech Window](image)

**Figure 4.5 The Speech Window**

The Settings window is the only window that provides the user with the full control to modify some of the settings of the robot related to the volume of the speakers, its IP address and the name that the robot holds. The window has three buttons, eight labels, three entry texts and one scale. The labels have used to display different messages regarding each section, the entry texts to allow user enter the proper information and the buttons to confirm these changes. The picture below displays the Settings window.
The next section that designed and developed is the demonstrations window. The reason of the existence of this window is to provide the user with a range of simple and more complicated actions that the robot can execute. There is again a separation into two sub panels, one holding the dances and the other more simple behaviors such as greets and bows. The elements placed on are twelve buttons and two labels.

The window that makes clearer, than any other window the functionality of the speech recognition that the robot can support is the robotics lab information. In this section the services provided to the user can be reached only by the clicking of one button. At the bottom side of the window there is a range of buttons that can allow the user to navigate on the robotics lab website and gather any information needs. On the upper part, has been placed a unique button that makes NAO start an induction presentation of the lab by interacting with the user at the same time. Through a discussion with the user and by using its speech recognition ability, the robot expresses a different kind of information related to the laboratory.
Figure 4.7 Demonstrations and Robotics Lab Info windows

During the development, as hasn’t taken into consideration from the beginning of developing the interface, has been decided to be included a window that provides information about how the robot is connected with the application. In this window the procedure of how the user can connect the humanoid robot with the application is described through an image step by step.

Figure 4.8 Manual window providing all the necessary instructions that the robot can connect with the application
4.2.3 Evaluation

In the paragraphs of the previous chapter have been described all the steps that have been followed during the designing and the development process of the graphical user interface of the application. The first thing noticed and is noteworthy, is that however a good plan has been done from the beginning of the designing stage, there was always something during the development that had to change. Maybe something extra has to be added or something else to be removed. Always the target remained the same, to provide though an interface platform, which included all the necessary tools which the user could proceed with the desirable actions that the robot could execute directly. Most of the windows have been changed the way they look many times until they take their final form. The evaluation about the integrity of the interface’s functionality at the end of this stage has been tested by executing many different checks. The tests which took place in order to guarantee that each button respond to the relevant functionality related to the interface, but not with the functionalities related to the robot, have been recorded and presented. The tests and the results which have been recorded refer only to the responses related to the GUI and not with the functionalities related to the robot. The actions combined with the result that was expected are shown in the table below.

<table>
<thead>
<tr>
<th>Window</th>
<th>Button Pressed</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Panel</td>
<td>Motion</td>
<td>Motion window opened</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Speech</td>
<td>Speech window opened</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Demonstrations</td>
<td>Demonstrations window opened</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Robotics Lab information</td>
<td>Robotics Lab window opened</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>How to connect NAO</td>
<td>Manual window opened</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Settings</td>
<td>Settings window opened</td>
<td>success</td>
</tr>
<tr>
<td>Speech</td>
<td>Clear Text</td>
<td>Empty textbox</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Info message (that the list is empty) displayed</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Filled textbox</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text box emptied</td>
<td>success</td>
</tr>
<tr>
<td>Clear Log History</td>
<td>Empty List</td>
<td>Info message (that the list is empty) displayed</td>
<td>success</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td></td>
<td>List contains items</td>
<td>List box emptied</td>
<td>success</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delete Entry</th>
<th>Empty List</th>
<th>Info message (that the list is empty) displayed</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Item has not selected</td>
<td>Info message (that no item has been selected) displayed</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Item selected</td>
<td>Selected item from the list deleted</td>
<td>success</td>
</tr>
</tbody>
</table>

| Add phrase to your list | Phrase added to the list | success |
| Press Exit on the upper right | Speech window closes and main panel window displays | success |

<table>
<thead>
<tr>
<th>Motion</th>
<th>Press Exit on the upper right</th>
<th>Motion window closes and main panel window displays</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Demonstrations</th>
<th>Press Exit on the upper right</th>
<th>Demonstrations window closes and main panel window displays</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Robotics Lab Info</th>
<th>Press Exit on the upper right</th>
<th>The Robotics Lab Info window closes and main panel window displays</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How to Connect NAO</th>
<th>Press Exit on the upper right</th>
<th>Motion window closes and main panel window displays</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Settings</th>
<th>Press Exit on the upper right</th>
<th>Settings window closes and main panel window displays</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 4.1** Table of tests taken to check the Desktop’s GUI functionality
4.3 Prototype 3 – Robot moves through commands from desktop

4.3.1 Design

After the completion of the graphical user interface the next and third prototype was to develop the code which was related to the functionality of the movements that run in the background of the application. At the beginning, it has been decided, to start developing only the basic movements that can make the robot move along all the directions on the space. Then the plan was to carry on creating some static actions and embed both, basic and static movements, with some checks and conditions before they could execute. Through a range of options, supported from the interface, the users can choose which unique command they would like to send to the humanoid robot. After receiving the command, the robot responds accordingly. The steps that have been followed for the accomplishment of the motion functionality was initially to develop the GUI and then attach to each button of the interface a function which would be responsible for executing the desirable action. The final prototype of the graphical user interface, including all the buttons of the motion window is shown below.

![Motion window providing full control of the robot's movements](image)

*Figure 4.9 Motion window providing full control of the robot’s movements*
4.3.2 Implementation

The study and analysis of the documentation, referred to the API and the modules that should be present, in order the robot proceed to movements, was necessary before starting the implementation of the code.

The module that was the core of the movement's functionality was the ALMotion. The ALMotion module provides a variety of methods which facilitate making NAO move. The four main groups that are associated with ALMotion are the stiffness of the joints, the position of the joints, the locomotion control and the control of the effectors in the Cartesian space.

Another one module that has been used is the ALTextToSpeech and makes the robot able to produce voice messages. The third module that has been declared in purpose of making the robot recognizes specific words from the user, is ALSpeechRecognition. This module in combination with the module of memory allows the robot through a predefined vocabulary of words to recognize specific words and execute relevant commands. The next two modules that have been used are ALMemory and ALRobotPosture. The existence of these two is specifically to retrieve data from the robot directly and more specifically the data related to the position of the robot. The declaration of these modules needs always three parameters which are the name of the module, the IP address of the robot and the port number. In each module has been given a name (tts, motion, posture, memory, speechrec) to be used within the class. The declarations of the modules which have been declared are shown below.

```python
IP = "137.195.108.208"
PORT = 9559

tts = ALProxy("ALTextToSpeech", IP, PORT)
motion = ALProxy("ALMotion", IP, PORT)
posture = ALProxy("ALRobotPosture", IP, PORT)
memory = ALProxy("ALMemory", IP, PORT)
speechrec = ALProxy("ALSpeechRecognition", IP, PORT)
```

After understanding how the modules are working, the functions for each movement has started to be developed separately. One of the main functions that are used in the initialization of the class is the method called connection. The function has a main role to check if the robot is
connected to the application or not. The method’s main check is based on the variable of the IP address which is retrieved from a text file. The first thing when the method is called is the reading of a text file containing the IP address of the robot. If the value equals to “Disconnected” a notification message informing the user that has to connect NAO is displayed. On the other hand, if the robot is connected, then the NAOqi library is imported, the relevant modules are created and some variables change their initial values.

def connection(self):
    # Opens the file and read the lines one by one
    filein = open("ip_and_name.txt", "r")
    temporary_list = filein.readlines()
    temporary_list = [lines.rstrip() for lines in temporary_list]

    if (temporary_list[0]) == 'Disconnected':
        tkMessageBox.showwarning("Error - NAO is Disconnected", "You have not connected NAO. Please go to <Settings> and set a proper IP address")
    else:
        # Sets the value that read from the text to the IP address
        self.ip.set(temporary_list[0])

        # Import NAOqi library and creates all the relevant modules
        from NAOqi import ALProxy
        self.tts = ALProxy("ALTextToSpeech", self.ip.get(), 9559)
        self.motion = ALProxy("ALMotion", self.ip.get(), 9559)
        self.posture = ALProxy("ALRobotPosture", self.ip.get(), 9559)
        self.speechrec = ALProxy("ALSpeechRecognition", self.ip.get(), 9559)
        self.memory = ALProxy("ALMemory", self.ip.get(), 9559)

        # Sets the position according the value is returned from the getpostureFamily method
        # Enables both arms of the robot
        # Initializes the command origin variable by setting the value to 0
        self.robot_position.set(self.posture.getPostureFamily())
        self.arms_status = self.motion.getWalkArmsEnabled()
        self.command_origin.set(0)
# Set the stiffness of all the joints of the robot to 1.0
# imports the words declared into the memory's vocabulary
self.motion.setStiffnesses("Body", 1.0)
self.speechrec.setVocabulary(self.words_list, True)

#The file that has been opened in order to be read closes here
filein.close()

The basic idea that most of the functions have been written on, it is more less the same. Firstly, there is a check about the position of the robot and then depending on its position either a movement in a specified direction is followed or an information voice message from the robot is produced. Another variable is responsible to check the origin of the command, more specifically if comes from the keyboard or by the voice of the user. Regarding where the command comes from, a list containing the log history of the movements is updated. Below is an example of the function stand up which makes the robot move to the upright position.

# Checks initially the position of the robot and then acts according to conditions.
# if robot is sitting down a voice message from the robot informs the user that it is going to stand up
# simultaneously sets robot position variable to “Standing”.
# This is a common variable for all move methods that is used to help recognise every time the position of
# the robot.Otherwise if robot is standing up a voice message informs the user that nothing will happen as it
# is in the desirable position.
def sit_down(self):
    if(self.robot_position.get()) == “Sitting”:
        self.tts.say("I am already sitting down")
    else:
        self.posture.goToPosture("Sit", 0.8)
        if (self.command_origin.get()) == 1:
            self.log_list.insert('end',("Sit down (Voice Command)") )
        else:
            self.log_list.insert('end',("Sit down (Keyboard Command)") )
        self.tts.say("I sat down")
        self.robot_position.set("Sitting")
The code that has been written for the rest of the functions related to the movements of the robot can be found in the appendix section at chapter 7.

### 4.3.3 Evaluation

The evaluation of this prototype has been done by testing all the scenarios by pressing the motion buttons and recording the results. That could guarantee the reliability of the system and can prove its smooth functionality. For that reason the scenarios took place was based on testing all the buttons and recording their responses each time. All the keyboard commands never had another result rather that the expected one. For example, when sit down button was pressed the robot was always going down. Only in case the robot was already sitting down nothing was happening except for expressing a voice information message that it is already in that position. But this response does not count as an error as this condition has been set up from the development stage. The same happened when forward, turn right and all the other buttons pressed. There was no any unexpected result in the section of the keyboard’s command in all the cases tested. Below, the table shows all the tests performed during the evaluation of the prototype regarding the keyboard commands.

<table>
<thead>
<tr>
<th>Initial Robot’s position</th>
<th>Button Pressed</th>
<th>Motion</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sitting</td>
<td>Sit down</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Stand up</td>
<td>Moved into the upright position</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Crouch</td>
<td>Moved into the crouch position</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move back</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move forward</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move right</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move left</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Turn right</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Turn left</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td>Standing</td>
<td>Sit down</td>
<td>Sat down</td>
<td>success</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>Stand up</td>
<td>No motion</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Crouch</td>
<td>Moved in the crouch position</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move back</td>
<td>Moved back</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move forward</td>
<td>Moved forward</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move right</td>
<td>Moved right</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move left</td>
<td>Moved left</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Turn right</td>
<td>Turned right</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Turn left</td>
<td>Turned left</td>
<td>success</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Crouch</th>
<th>Sit down</th>
<th>Sat down</th>
<th>success</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stand up</td>
<td>Moved into the upright position</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Crouch</td>
<td>Moved in the crouch position</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move back</td>
<td>Moved back</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move forward</td>
<td>Moved forward</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move right</td>
<td>Moved right</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Move left</td>
<td>Moved left</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Turn right</td>
<td>Turned right</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Turn left</td>
<td>Turned left</td>
<td>success</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2 Table of tests and results regarding the Motion functionality
4.4 Prototype 4 – Robot interacts through predefined keywords

4.4.1 Design

The next module that has been decided to be developed after the motion functionality stage was the speech recognition. The robot could accept a stimulus from the user and act accordingly. The speech recognition capability of the robot would be implemented in two different sections of the application. The first part would be set up on the Motion panel and the second on the Robotics Lab Information. On the motion panel the user would be able to make the NAO move through voice commands and on the robotics lab the scenario that could be covered would an interaction between the robot and the human. The robot could introduce the user to the Robotics Lab and though a discussion the proper information related to the laboratory could be released to the user. The core of the speech recognition was the ALSpeechRecognition module and the ASR engine which is installed on the robot. So the first approach was to study the steps that have to be followed in order the module can work properly, then to be decided where these actions would be used within the application and after proceed with the implementation.

4.4.2 Implementation

The ALSpeechRecognition module gives to the robot the ability to recognise predefined words or phrases in several languages. The technology that the robot could get and understand the words from an external source are specific. At the beginning NAO has to be fed up with a list of phrases and words. Then places in the key called SpeechDetected, a boolean that specifies if the speaker was perceived or not. If the value of this variable has been set to true, the element of the list that best matches what is heard by the robot is placed in another key called WordRecognized. The format of the key is shown below.

[phrase_1, confidence_1, phrase_2, confidence_2,........, phrase_x, confidence_x]

The two parameters define the behavior of the robot as phrase x is the predefined phrase of the list that have been declared by the developer at the beginning of the procedure. The other side confidence x is an estimate of the probability that this phrase is indeed what has been
pronounced by the external source. Below is described how the speech recognition for the motion module has been developed and which methods have been used during the implementation.

The method that is called getMostConfWord() is one of the three methods which gets the phrase that the user provides and depending its weight returns the relevant word in order an action take place.

```python
# Selects the word with the highest level of confidentiality from a list of predefined words (that has been set at initialization) when users speak to the robot.

def getMostConfWord(self, recogList, threshold):

    # Declaration of local variables used within the method
    i = 0
    wordMax = ""
    confMax = 0

    # A while loop checks that as long as the length of the list is bigger than 0
    # then the first element will be the word and the next the confidence level of that word.
    # there is a step +2 every time to identify the next pair of word and confidence level.

        while i < len(recogList):
            word = recogList[i]
            conf = recogList[i+1]
            i +=2

        # if the confidence level of the word is larger that the max level
        # and at the same time larger that a given value then returns that word.
        # Check_word is called at the end to attach an action to the word has been returned.

            if conf > confMax and conf > threshold:
                confMax = conf
                wordMax = word
                self.check_word(wordMax)
                return wordMax

    return
```
The next method is the one called vocabulary. When this method is called opens a dialog window between the user and the robot that lasts three seconds. During that time the robot listens the words that the user expresses, retrieves from its memory the list of the predefined words declared beforehand and with the collaboration of the getMostConfWord method acts in a different way every time depending on the word has been received. At the beginning of the dialog robot speaks to the user and informs him that is ready to accept any words want to provide.

```python
def start_vocabulary(self):

    # Imports a list of words into the memory's vocabulary of the robot
    self.speechrec.setVocabulary(self.vocabulary, True)

    self.tts.say("Hello everyone.My name is" + self.robot_name.get())
    # Reads an introduction message fro a text file
    self.read_text_info("intro_message.txt")

    while self.speech_counter == True:
        self.speechrec.subscribe("tts")
        time.sleep(4)
        list = self.memory.getData("WordRecognized")
        self.getMostConfWord(list, 0.30)
        self.speechrec.unsubscribe("tts")
    else:
        self.tts.say("You have selected to stop the induction to the lab.Thank you very much")
        pass

    return

The implementation has been based on these two methods described above and a third one called check_word with triggers every time a different action depending on the word that has been recognised from the ASR engine. All the code can be found in the Appendix section at the end of the thesis.
4.4.3 Evaluation

The section of the voice messages had not the perfect functionality that the keyboard commands offered from the first moment. There were times that the robot couldn’t hear properly or some other times that could proceed with a different action than the expected one. Some tests initially done, with the purpose to find the time that the channel has to be open during the user express a word and the time that the robot needs to receive that word. It should have been a time not that short but at the same time not that long. For that task, a variety of time periods tried, but at the end the four seconds has been decided to be the time period that the communication would stay active. The second problem that has been arisen was about an unexpected termination when the connection was active. If the communication channel is on and something happens to the application that forces it to terminate unexpectedly, then the ASR (Speech Recognition) engine that runs on the robot stays on. That problem still remains. The tests that have been taken for the evaluation of the prototype, based on ten unique executions for each command. The distance kept from the robot, while speaking, was two meters and it was the same for all the commands. For each task has been recorded how many times the robot failed to proceed with the expected action and how many times returned a successful response. The results are displayed in the table and shown the performance of the robot.

<table>
<thead>
<tr>
<th>Keyword given</th>
<th>Expected Action</th>
<th>Success</th>
<th>Failed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sit down</td>
<td>Robot to sit down</td>
<td>7/10</td>
<td>3/10</td>
</tr>
<tr>
<td>Stand up</td>
<td>Robot to stand up</td>
<td>7/10</td>
<td>3/10</td>
</tr>
<tr>
<td>Crouch</td>
<td>Robot to move to crouch position</td>
<td>5/10</td>
<td>5/10</td>
</tr>
<tr>
<td>Move back</td>
<td>Robot to move backwards</td>
<td>10/10</td>
<td>0/10</td>
</tr>
<tr>
<td>Move forward</td>
<td>Robot to move forward</td>
<td>7/10</td>
<td>3/10</td>
</tr>
<tr>
<td>Move right</td>
<td>Robot to move right</td>
<td>9/10</td>
<td>1/10</td>
</tr>
<tr>
<td>Move left</td>
<td>Robot to move left</td>
<td>8/10</td>
<td>2/10</td>
</tr>
<tr>
<td>Turn right</td>
<td>Robot to turn right</td>
<td>7/10</td>
<td>3/10</td>
</tr>
<tr>
<td>Turn left</td>
<td>Robot to turn left</td>
<td>8/10</td>
<td>2/10</td>
</tr>
</tbody>
</table>

*Figure 4.10 Tables displaying the results of the Speech Recognition’s evaluation*
4.5 Prototype 5 – Developing Android Application

4.5.1 Design

The fifth prototype was about the designing and development of a graphical user interface for the Android application. Initially, before the development starts, a detailed plan has been done in order to be known from the early beginning, which functionalities and what will be included in the application. The two main functionalities which have been implemented first, was the ones related to the Motion and the Speech capabilities of the robot. Through two relevant panels (Motion and Speech), the user has the opportunity to make NAO move and speak accordingly the commands executed. Another service, providing a number of modifications such as changes related to the name, the IP, the sound of the microphones and the stiffness status would be the Settings window. Also in the plan, have been included, some functionalities based on the speech recognition ability of the NAO and a panel which would allow the humanoid proceed to a sufficient number of demonstrations.

4.5.2 Implementation

The language, that the second part of the thesis has been written and developed, is exclusively Java. Furthermore the IDE, which was the core that all the features built up, was the Eclipse suite and more specifically its Luna Edition. The Android SDK (version 5.1, API 21) was the platform that has been selected and added to Eclipse for the development of the application. Also the custom plugin for the Eclipse IDE, called Android Development Tools (ADT) which provides a powerful, integrated environment has been downloaded and used accordingly. The plugin extends the capabilities of the development environment and allows the developers quickly set up new Android projects, build up graphical user interfaces and debug the application. As the functionalities have been decided during the first step of planning, the implementation stage has separated into many different smaller parts. For the Android device’s development, activities are the alpha and omega so for that reason, each functionality has been attached to a unique activity. The application consists of five activities which are responsible to provide on each panel a number of methods which in turn create for each element a different behavior. The Main window consists of six buttons, one text view and an image on the center of the screen. The buttons allow the user to navigate through the application and their appearance has been
modified as well. The xml file that has been applied to all the buttons of the application specifies the properties of the background color, the size and the shape of the elements. Below is shown the Main panel on the landscape view and the portrait view as well.

![Figure 4.11 Main Panel Activity on landscape and portrait mode](image)

The second step included the implementation of the Motion panel, which is related to all the capabilities of the robot to move. Movements such as forward, backwards, turn on both sides are some of the actions that the robot could execute. Two text views and fourteen buttons are all the elements included in the window. Both text views are located on the top of the screen and one is the header and the second one is the sub header, which displays an information message of the functionality of the window. The first four buttons are placed in a cross schema and their classical views have replaced with four images of arrows each one representing the direction that NAO can move. The next four, are again placed in a group and provide the functionalities that make the robot able to turn, sit down and stand up. The rest of the buttons are located on the bottom of the screen and their job is to allow user to navigate backwards and forwards between the activities. The Motion Panel is displayed on its final form on both views.
The next step that has been followed was the Speech panel. As it is obvious from its name, was the one that the user could handle NAO’s Speech. The control of this functionality could be succeeding either by the insertion of text manually or by selection of predefined phrases from a list. The user has the option to choose both ways that can make produce a voice message. The window consists of three text view elements, nine buttons, one edit text and one list. A text box provides the space of inserting text manually according to which the robot can produce a voice expression. Three buttons, below the edit text of the manual entry, allow the user to choose if the robot will speak or the manual text in the box above will be inserted into the list and kept for future use or to the list will be cleared of its contents. Again the buttons which have been placed on the bottom of the screen make the navigation through the panels an ease. In the two images below are shown the motion’s views.
Settings panel is the next activity that has been developed for the Android application. The activity consists of nine buttons, eleven text views, two edit texts and one seek bar. The first couple of elements (edit text-button) handle the value that has to do with the name of the robot. The text view is accepted by the user a text manually and the “changeName” button changes the value according to the text provided. The second couple of elements (edit text-button) functions exactly the same, but handles the value of the IP address. The third and last couple which consists of a seeker bar and a button modifies the volume of the sound on the microphones of the robot each time the cursor on the bar is triggered. After the group of these elements, follows another group that consists of seven text views and displays all the information that has been referred above with a more concentrated view. Information is related to the name, the IP address and the volume of the sound. At the bottom of the screen are six buttons which provide navigation between the panels. The portrait and the landscape view of the Settings panel is shown below.
Because the functionality of the demonstrations and the speech recognition hasn’t been developed at the time didn’t prove enough at the end, a proper interface for these two panels has not been designed. Instead an activity consisting of an image and a background image has been implemented to inform the user every time navigates along those two that the functionalities are in the process of construction. The image below shows how exactly this activity looks like.

Figure 4.14 Settings Activity on landscape and portrait mode

Figure 4.15 Under Construction Activity on portrait and landscape mode
4.5.3 Evaluation

The first stage of designing and developing a graphical user interface for an Android application proved, in the end, a very straightforward procedure. The evaluation of this prototype after the completion of all the targets that have been set up at the beginning, was based on the tests took place and had as main purpose to check if all the elements function properly. The procedure followed was to download and run the code into an actual mobile device which supports an Android environment. The device that has been used through all the tests was a Google Nexus 7 tablet provided by ASUS. The first thing that has been checked after the application downloaded and ran on the mobile device was if the positions of the elements were exactly the same as the ones have been displayed on the emulator ran on the computer device. All the elements had the desirable position on the screen, so no any further action taken after the first execution of the application on the device. The next check was to verify if all the functionalities related to the GUI, but not the ones (movement, speech, settings) that are related to the robot, were working properly. All the checks, at the end of the development of the first prototype, guarantee that the element's behavior matches exactly the one has been set up in the designing plan. These tests have been recorded and displayed on the table below.

<table>
<thead>
<tr>
<th>Window</th>
<th>Button Pressed</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Panel</td>
<td>Motion</td>
<td>Motion activity launched</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Speech</td>
<td>Speech activity launched</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Demonstrations</td>
<td>Under Construction activity</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Robotics Lab info</td>
<td>Under Construction activity</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Battery</td>
<td>Battery activity launched</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Settings</td>
<td>Settings activity launched</td>
<td>success</td>
</tr>
<tr>
<td>Speech</td>
<td>Motion</td>
<td>Motion activity launched</td>
<td>success</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------</td>
<td>--------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Main Panel</td>
<td>Main Panel activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Demonstrations</td>
<td>Under Construction activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Robotics Lab information</td>
<td>Under Construction activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Battery</td>
<td>Battery activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Settings</td>
<td>Settings activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Clear List</td>
<td>Empty List</td>
<td>Info message (that the list is empty) displayed</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>List contains items</td>
<td>List view emptied</td>
<td>success</td>
</tr>
<tr>
<td>Add phrase</td>
<td>Empty Text View</td>
<td>Info message (that the list is empty) displayed</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Text View with text</td>
<td>Phrase added to the list</td>
<td>success</td>
</tr>
<tr>
<td>Motion</td>
<td>Main Panel</td>
<td>Main Panel activity launched</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Speech</td>
<td>Speech activity launched</td>
<td>success</td>
</tr>
<tr>
<td>Demonstrations</td>
<td>Under Construction activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Robotics Lab information</td>
<td>Under Construction activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Battery</td>
<td>Battery activity launched</td>
<td></td>
<td>success</td>
</tr>
<tr>
<td>Settings</td>
<td>Settings activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>Main Panel</td>
<td>Main Panel activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Speech</td>
<td>Speech activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Demonstrations</td>
<td>Under Construction activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Robotics Lab information</td>
<td>Under Construction activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Battery</td>
<td>Battery activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Main Panel</td>
<td>Main Panel activity launched</td>
<td>success</td>
<td></td>
</tr>
<tr>
<td>Change Name</td>
<td>Text View’s name value changes</td>
<td>success</td>
<td></td>
</tr>
</tbody>
</table>

*Table 4.3 Table of tests and results regarding the Android's GUI functionality*
4.6 Prototype 6 – Establishing communication between Android app and NAO

4.6.1 Design

After the completion of the GUI everything has been setup on the screen and was provided to the user the opportunity, when the application would connect to the robot, proceed to a direct interaction through a variety of buttons and elements. The sixth prototype was responsible for the establishment of a proper communication between the robot and the application, so the user has the full package of an interface and a communication something that will allow him to handle the robot in different ways. This prototype, from the first moment of this thesis, has been characterised as the most unpredictable. Mainly because it was unknown, before the development stage starts, if the commands via jNAOqi or the direct HTTP requests between the robot and an Android application could work.

4.6.2 Implementation

As it has been explained above, it was difficult to plan which path had to be followed in order the target could be accomplished. The first thought was to find a way to include the jNAOqi library to the Android mobile device and in that way by using the existing modules of the library start building methods that will allow the robot moves speak and function accordingly. This target very fast proved infeasible as the library was not compatible with the 1.14 version of the NAO used. So the next step that has been considered to be tried was to use an intermediate, more specifically an access point, in which both the robot and the application would connect to. With this approach Http requests would be sent from the application to the robot through the wireless access point. The robot also from the same access point could send the responses back to the mobile device. Eventually was proved that in this way the communication could work efficient and after some tests that took place by using Http Requests and Responses provided by Java that scenario confirmed. So has been decided that the connection could be successful through an intermediate if a local network could be set up.

The implementation phase was to set up the network that would include, the NAO robot, the Android device and the modem which allows both of them would connect to. The router used for establishing this network was the NETGEAR DG834G model and could provide to the devices
the option of cordless or wired connection. The image below explains in detail the logic and the relationships that the devices have inside this local network.

![Diagram of local network](image)

*Figure 4.16 Local network providing communication between NAO and Android Tablet*

After all the equipment has been set up, then the connection has to be tested. Http Requests have been created and send through the GUI to the modem in order to check that the communication could be established. The first Http request send to the robot was to make it move right. The format of the command was not exactly the same as the ones have been used in Python e.g. (ALMotion.moveTo (0,-0.2,0)) but needed some changes because browser doesn’t accept all the special characters and needs to convert them beforehand in order to understand them properly and eventually execute them.

```java
HttpGet httpMoveRight = new HttpGet("http://192.168.0.5:9559/?eval"
+ "="ALMotion.moveTo(0%20,0%20-0.2,0%20);"
```

After the successful results that confirmed that the robot and the application could communicate through the HTTP requests and responses another test has to be taken place
regarding the lifetime status of the connection of the robot to the network. For that reason an Http Request created and being sent to check the value of the response sent back from the robot. This functionality would prove very important, for the reason that would be used in the application at frequent intervals to check if the robot is still connected to the network or has been disconnected. The Http request was formatted as below and in case the connection was alive the response was true, otherwise there was no response at all.

```
HttpGet httpConnectionStatus= new HttpGet("http://192.168.0.5:9559/?eval"
    + "=ALMemory.ping()";
```

### 4.6.3 Evaluation

The sixth prototype related to the establishment of the connection between the robot and the application was proved tricky and took much more time that it has planned on the initial schedule. At the beginning, much time spent on finding a way how this type of connection could be accomplished mainly by using the jNAOqi library. The tests have failed as the library was not compatible with the robot’s version 1.14. After the initial tests the second approach that consisted of the Http requests have been great solutions as offered a very straight forward procedure to allow the developer, by just sending simple commands to the browser can handle a wide range of the capabilities of the robot. The approach adopted immediately and the rest of the functionalities were built up on that technique.
4.7 Prototype 7 – Robot moves through commands from Android application
4.7.1 Design

After the successful establishment of a connection between the Android application and the humanoid robot the next prototype that has been created was able to provide all the functionalities related to the movements. From the designing of the GUI has been decided that the actions that the robot could execute would be to move in all directions of the space through navigation buttons and also sit down, stand up, turn left and turn right. Through all the buttons, provided in the graphical user interface of the Motion Activity, each one has been attached to one unique movement. The user by only a single click of the button could make the NAO move in a desirable position. The final view, including all the motion buttons of the Motion panel is shown below.

![Motion panel](image)

*Figure 4.17 Motion panel*

4.7.2 Implementation

Similar to the desktop application the main feature that has been used for achieving the motion’s proper functionality was the ALMotion module. ALMotion provides methods that help make the NAO move along all the directions and also contains commands allowing the user to
manipulate the joint stiffness and the joint angles. The way that the robot can receive commands in order to execute them is through HTTP requests. Each time a button is pressed a request is created and being sent to NAO through the wireless access point. The format of those commands includes the declaration of the IP of the robot, its port number, the module of ALMotion and the relevant parameters that the module needs to contain. All the information needed is included in one single line. Below is an example of the format that these types of commands have when an execution of sitting down movement is declared.

http://192.168.0.5.166:9559/?eval=ALMotion.move("Sit", 0.2)

On the command above http:// is the classical format that all the Http requests have at the beginning. Then the 192.168.0.5 is the IP address of the robot and 9559 the port number. The declaration of the module is the next part of the command and everything is included in the parenthesis are the parameters that passed to the motion module. Depending on the parameters each time a different command is executed.

The user can provide the robot with a desirable action by clicking among a wide range of buttons provided on the interface. The buttons, one by one, have been attached programmatically to a click listener. The listener is separated in cases, as many as the buttons are and each case provides a different behavior that the button can trigger. So every time the user clicks on a button, then a separate thread, which is responsible for the execution of the relevant movement, starts running. The threads based on a runnable interface which contains a method called run and is responsible to create for each case an HTTP request and get an HTTP response. That way a request for making the robot move to a specific direction is sent and then in turn a response is returned back. In the code below is the method of the click listener that is attached to the move forward button and the runnable interface which deals with the threads related to the HTTP request and responses.

On Click Listener

/ * Declares a single click listener which will produce a relevant
* result every time a different action occurs. e.g user presses a button or
* a single click inside an edit text view.
* /

OnClickListener oclButton = new OnClickListener() {
@Override
public void onClick(View v) {
    // Defines the buttons that invoked the listener by their id
    switch (v.getId()) {

        // When button is pressed a new thread is created and starts running
        case R.id.btUp_Motion:
            Thread MoveForwardThread = new Thread(runnable);
            MoveForwardThread.start();
            break;

    }
}

Runnable Interface
/* Creates a runnable interface which includes the method run
* and will allow threads to run when is needed by calling the runnable interface itself
*/
final Runnable runnable = new Runnable() {
    public void run() {
        try {
            //Creating an HTTP client
            HttpClient httpClient = new DefaultHttpClient();
            //Creating an HTTP Post to make NAO move
            HttpGet httpMoveForward = new HttpGet("http://" + addressIP + ":9559/?eval" + "=ALMotion.moveTo(0.2,20,0,200,200)");
try {
    /* A response is declared and executes each time the action(HttpGet) that
    * equals the the selectedAction variable. Afterwards the response is converted to a string
    * and can be used everywhere in the application is needed
    */

    HttpResponse response = httpClient.execute(httpMoveForward);
    String responseBody = EntityUtils.toString(response.getEntity());
    }

    catch (ClientProtocolException e) {
        // writing exception to log
        e.printStackTrace();
    }

    catch (IOException e) {
        // writing exception to log
        e.printStackTrace();
    }

    catch (Exception e) {
    }
}
4.7.3 Evaluation

The prototype of the motion functionality based exclusively on the Http requests and responses. After the point, that proved that it was a straight forward procedure, it was very easy for the rest of the motion capabilities to be added. However, at the end of this stage, in order to evaluate the success of the development that the functions related to the movements had a series of tests taken place. By pressing the buttons and recording the responses on a table is obvious that everything worked exactly the way it meant to be. Below is the table which displays the actions taken, the responses and how successful were, compared to the expected result.

<table>
<thead>
<tr>
<th>Initial robot's position</th>
<th>Button Pressed</th>
<th>Motion</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sitting</td>
<td>Sit down</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Stand up</td>
<td>Moved into the upright position</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move back</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move forward</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move right</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move left</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Turn right</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Turn left</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td>Standing</td>
<td>Sit down</td>
<td>Sat down</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Stand up</td>
<td>No motion</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move back</td>
<td>Moved back</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move forward</td>
<td>Moved forward</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move right</td>
<td>Moved right</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Move left</td>
<td>Moved left</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Turn right</td>
<td>Turned right</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td>Turn left</td>
<td>Turned left</td>
<td>success</td>
</tr>
</tbody>
</table>

Table 4.4 Table of tests and results regarding the Android's Motion functionality
4.8 Prototype 8 – Evaluation by users of Desktop Application

4.8.1 Design

The last prototype was focused on the evaluation of the desktop application by the actual users. The prototype was based on the idea that after the completion of an integrated application, the users could have the time to interact with the environment and express their own experience through that procedure. Eventually the responses would give the developer, a more spherical view of the application’s functionality, but simultaneously a more focused and detailed look on what is needed to change or be added as well regarding the opinion of the users.

4.8.2 Implementation

On the stage of the implementation, the main role played by the sessions that each user has attended for a predefined period of time and the questionnaires that have been created for the purpose to be filled by the users at the end of each session. The questionnaires consisted of thirty eight questions which have been split into ten smaller parts. Each part contained three to four questions that they were related to the different functionalities of the application. Below is shown a sample of the questionnaire, containing the first five questions. The whole document can be found at the end of the thesis in the Appendix section.

![Questionnaire](image)

*Figure 4.18 First page of the questionnaire*
The plan that has been decided to be followed, before the evaluation starts, was the same for all the users. More specifically, each session last twenty minutes, ten minutes would be spent in filling the questionnaire and five minutes at the end for a brief discussion with each user separately in order to retrieve any kind of information that has been missed through the rest of the procedure. It has to be mentioned that no information given prior to the start of the procedure as the purpose of the evaluation was based on the scenario that the user have no clues of how everything works beforehand. Actions such as booting up the robot, establishing an Ethernet connection and running the application’s interface have been set up before each session starts. The user had to connect the robot (providing the IP in the relevant box) with the application by reading a manual that was included in the graphical user interface and then starts interacting directly with the NAO. One of the users, that have attended one of the sessions held during the evaluation process, is shown below.

Figure 4.19 User during the evaluation process interacting with NAO
4.8.3 Evaluation

After the completion of the sessions, thirteen in total, the questionnaires have been collected and analyzed in order to retrieve as much information possible regarding the desktop application for NAO. The answers have been recorded initially on tables and relevant graphs have been deployed. The first questions were of general content, including data related to the age, to the gender and to the relevant experience that each user had with the robots in general. All the participants were between 26-35 years old including MSc students, PhD students, professionals working in the IT field and additionally people with professions from other fields (financial, healthcare). All of them tend to use a computer device in a daily basis with eight out of thirteen for more than ten hours and also at least once a day a desktop application. Regarding the information gathered from the previous experience that the users had with the robots has shown that only two of them have used desktop robotics applications before, so it should have taken into consideration that most of the people haven’t any knowledge or familiarization with the subject.

![Interface’s Layout Preference](image)

*Figure 4.20 Responses to the question “How did you find the layout of the interface”*

After the first part, the next questions related to the appearance of the layout and how easy it was to navigate within the application. Most of the evaluators believed that the design was very good, but the position of the elements had to change because made the navigation a bit confusing. The graph (Figure 4.18) shows the number of people who have declared their opinion for the layout’s appearance.
Figure 4.21 Responses to the questions "How easy is to navigate through the application and "How easy is to use the application's interface and explore it's functionalities".”

However the biggest percentage of people declared that the elements (buttons, labels, image) weren’t placed very wisely on the Main panel and would prefer another arrangement as sometimes made them feel confused, they have answered that the navigation through the application was very easy. The stats (Figure 4.19) appear that nobody has occurred problems to explore the functionalities and navigate through the panels.

Figure 4.22 Responses to the question “How difficult was it to find out how NAO can be connected to the application”.

The next group of questions referred to the connectivity and focused on the level of difficulty that the users have faced when they tried to find a way how to connect NAO with the desktop application. One of the two reasons that no information given at the beginning of the sessions, as
has been mentioned previously was because it would be beneficial to watch and record the behavior of the users when they were trying to succeed this task. The application has been designed and developed in a specific way from the first moment that wanted to guide the user through the steps in order to find on his own what has to do exactly in order to successfully connect NAO. Users could find the information from everywhere in the application if any problems with the communication have arisen. The 46% of the people have said that the procedure was quite easy, the 31% that it was moderately difficult and the 23% that it was very easy. The chart (Figure 4.20) above shows the exact number of answers given by the users.

The very important role of how a proper connection can be achieved played the messages that have been placed on the development stage. A manual showing step by step the procedure and pop-up windows inform the user, when the robot was disconnected, in which place has to move was the supporting material. Referring to the messages, except from the connectivity manual provided, detailed texts have placed on the top of each panel to inform the user about the functionality that could be served in each window. Users asked to answer how helpful these messages proved, in the end, so they could establish a successful connection and explore the functionalities of the application (Figure 4.21).
The next block of questions consisted of the analysis related to the motion functionalities that the robot could execute. The NAO could either move through the keyboard commands or by voice messages. The results of these questions mainly wanted to retrieve the level of difficulty that the users have faced when they needed to use the one or the other method and additionally check the performance of both. It was very obvious, as shown in the relevant chart (Figure 4.22), that the keyboard commands were much more straightforward procedure and haven’t had caused any kind of malfunction at any time the used, instead the voice commands haven’t had the desirable outcome all the times. 85% answered that the robot has always responded through the keyboard with commands to the expected action. Compared to the 8% of the voice commands, the dominance of the keyboard module was obvious. This big difference, between the two methods used, makes it clear that the voice command’s module need to be improved. The main reasons which have been noticed that have caused that very low percentage, was the sensitivity of the microphone on the robot’s head and the non-experience that the users had before they interact with the application.

The hardware issue that the robot had with the microphone’s volume it was known from the early tests that have been taken during the development stage. It was very difficult for the user, to find out from the beginning, how close and how loud should have spoken to the robot in order to accept the command properly. And in some cases, if the right distance has been found and the loudness in the voice has been determined again the robot did not respond always the same.
The second problem that mainly caused most of this undesirable behavior was proved that there was the lack of knowledge of the users. None of the users did not know how exactly the speech recognition module (ASR) works for the robot. In many cases, users did not wait for the robot to finish and were talking on top of it, something that is not working because when the robot uses the speech module cannot keep or accept any voice commands. Firstly has to stop talking, open the reading channel and then is ready to accept any commands. Also, the users did not use the right words every time. Sometimes also have noticed that evaluators were trying to describe what they wanted the robot to do (e.g. Can you go forward and then turn right) instead of providing the already predefined expressions which have been attached to each movement uniquely. In some cases also there were examples that they could not find which word to use. For example, if they wanted to make the robot move forward could not understand if the right expression was the single word forward or move forward or maybe go forward. However, this occurrence’s importance is not that crucial as the robot could understand in such small expressions the key word which is the word forward and execute the command. But still there were cases that the robot proceed with an the action because was confused which expression have been provided. In the statistical point of view (Figure 4.23) 38% answered that the robot did not respond properly, three to five times, 31% one to two times, 23% six to nine times and only 8% that the robot executed the right one every time what is asked for.

The last issue that has seemed to cause some problems was the accent of the users. Because in the tests, took part also people that they do not have as their native language or do not speak fluent English, it was noticeable that the way they expressed the words caused a confusion to the robot sometimes. Compared to the people that they were more fluent it has been recorded that
the problems related to the accent have been decreased in a degree. Although, it was not a major issue, it was clear that a more proper pronunciation could bring better results.

![Voice commands precision RLI vs Motion window](image)

*Figure 4.26 Responses to the question “Has the robot executed the right action anytime you used a word to make it move” both on the Motion and RLI panel*

The speech recognition module except for the motion panel, have been used also in the robotics lab information section. The routine was exactly the same. The user could provide the robot with a key word and get a relevant response back. The only minor difference was that in the motion module the response was a movement of the robot, but in the robotics lab section the response was a voice message that was supporting a direct interaction between the human and the humanoid. However, it was more less the same module, have been noticed that the users have not faced the same high percentages of error propagation (Figure 4.24).

![Motion Panel](image)

*Figure 4.27 Comparison on the question “How many times the robot couldn’t hear which word you used on average and asked you back to repeat the phrase” between the Motion and RLI panel*
One of the reasons for this difference, between the two modules, although the same service is supported is mainly because the users have acquired a small experience as they were going along with the application. The sequence that almost all users have followed from the beginning to the end of each session was by exploring the functionalities in the below order: Manual - Settings - Motion - Speech - Demonstrations - Robotics Lab information. So as firstly, they have come in touch with the voice commands on motion module, helped them to understand better the procedure of how the speech recognition works and how they have to cope with the robot. The percentage of the case that there was not a proper reply between one to two times has been doubled from 31% to 62% and the percentage of the times between six to nine times has been eliminated in 8% (Figure 4.25). The statistics strengthen the opinion that prior knowledge and experience helps the user face the functionality of the speech recognition much better. Experienced users proved, for the second time that they used the module, which had another behavior as they now knew that they had to speak closer to the robot, a bit louder, clearer and also wait for the other side to finish and then start talking.

![Figure 4.28 Responses to the questions "Which part have you found the most easy to use" and "Which part have you found the most difficult to use"](image)

One of the questions placed after the analysis of all the modules, was which function the users found most difficult to understand and use properly. The answers, about the easy one, gave a clear precedence to the functionalities related to the keyboard commands. Either the motion or the speech panel, which executed by straightforward use of the keyboard make everyone feel comfortable. From the opposite side, the hardest module has been voted to be the ones related
to the voice commands. Not specifically, but either the ones are at the robotics lab information or the motion module. The chart below shows how the preferences of the users have listed the functionalities according their level of ease or difficulty (Figure 4.26).

Except for the questions that have given the option to the user to select from a group of predefined choices, also there were questions that provided the space (text boxes) to the user so briefly could express his opinion and write some general comments. Some of them were important to clarify more the weaknesses and the strengths of the application. The first thing that almost all users mentioned was the lack of a button on the UI that could have given the option to the user, from everywhere in the application, navigate back to the Main panel. It was a bit confusing, to understand from the first moment that the return to the Home panel could be done by just closing the existing window. The manual referred to the connectivity instructions, suggested that could be placed in another position on the layout or maybe could be provided inside all the windows and become accessible from everywhere at any time.

Regarding the Demonstrations, there were two main suggestions. The first one was that some more movements and demos could be added and secondly the existence of a button that could make NAO stop at any time that the user wanted.

Lastly were the comments which referred to the interactive communication that the users had in the robotics lab information panel. Some people suggested that they would prefer to hear some more information about the subjects covered from the robot, but there was another group of people who have declared that the robot was speaking too much in which sometimes that ended up very tiring and boring. The opinions shared pretty much equally as the 38% percent supported the view of less information, the 30 that they would like to hear some more information and the rest 32% haven’t commented at all.

The evaluation of the desktop application proved a very important prototype as it has covered two main tasks. Initially could check the performance of the application and secondly reveal all the weaknesses that need to be improved in the near future.
5 DISCUSSION

The project met most of the goals which have been set up at the beginning. The mixture of the theoretical approach and the practical aspect by developing two different types of applications had as a result the remarkable evolvement of my personal development skills and the expansion of the knowledge around humanoid robots.

In the first part the theoretical analysis, placed the fundamentals in which the technical part has been based on. The subjects covered by the literature review included the history of the humanoid robots and how they have been used in the perspective to interact with humans. The following part focused on the analysis of the NAO, which was the core of this thesis, and presented the applications that has been built around the robot, to allow its wired or wireless control. This detailed research was fundamental, as the architecture of NAO has been studied deeply and also the techniques how other developers have approached and developed applications for that purpose. That proved very important in cases that something went wrong from the initial plan because it could provide with alternative paths to think, follow and test for the final objective which was no other of developing a desktop and an Android application for the wireless control of the NAO.

After the completion of the theoretical part, the practical section consisted of eight different prototypes which have been assigned to each one a performance assessment task on a specific date. That assignment was very crucial because kept the project’s timetables on the rails, imprinted in which level each prototype has reached, but simultaneously could show the progress of the whole project. Also, as some implications occurred during the development, the focus on each target has never been lost and any weaknesses appeared have been re-assessed and dealt in a different way.

First, of both applications, has been started the development of the desktop application. A graphical user interface has been designed and implemented in order to support the execution of NAO’s functionalities. Then each unique module has been inserted in the application gradually. The Python language that used for the first time proved a great tool and the Tkinter package was the basic material that has been used for the development of the GUI. The modules of Motion and Speech have been proven completely different things compared to the Speech Recognition. For that reason, that specific module, has been approached in another way in order to be implemented. The Motion and the Speech were a very easy, straight forward procedures which based on the declaration of the ALMotion or ALTextToSpeech modules accordingly and the existence of the relevant command regarding the action (speak or move) that has to be executed each time. From the other side, the Speech Recognition functionality except for the declaration of
ALSpeechRecognition module, needed also a couple of methods to wrap around it in purpose the robot can accept words, process them and exponent the final result. These methods were a bit tricky, because the time that the channel had to be open in order a phrase could be provided by the user, should have been declared in the code at the beginning. That time period was sometimes short and in other cases too long. For that reason many tests taken so the right period of time to be found and the robot could accept the words from the users accurately. However, as this issue has been resolved, other problems have been occurring during the interaction with the users. Problems such as the robot couldn’t always understand the word provided or from the user’s point of view, it was difficult to understand immediately how exactly the module functions in order to increase the percentages of successful responses from the humanoid. The connectivity between the NAO and the desktop application has been implemented without causing any problems, as long as the IP of the robot was known.

After the completion and the tests that have been taken place, for ensuring the full functionality of the desktop application, the next step that was the designing and the development of the Android application. Initially the graphical user interface has been developed and then the functionalities of the Speech and the Motion have been built up one by one. The problem that has been faced on that stage was that the jNAOqi library was not compatible with the version 1.14 of the NAO robot used in the whole duration of the project, so another way had to be found in order the robot could communicate with the mobile application. Some critical time spend, as many tests executed in order a solution to be found, but at the end none of them worked properly. Eventually, the Http requests provided the solution and were the core of the second stage as everything related to the modules of the robot based on them. The two functionalities of NAO that the application includes is the capability of speaking and the one that can trigger the robot to move. As the time was not enough at the end, there was no any effort to integrate the Demonstrations or the Speech Recognition modules, as has been done with the desktop approach.

One of the most important tasks of the dissertation was the last prototype in which an evaluation of the desktop application took place. Thirteen users from different backgrounds and professions, without most of them having prior experience with the robots used and evaluated the application. The reason that people without previous experience and most of them were outside the field of computer science, was basically because the application has been developed with the main purpose to offer an easy way to everyone interact with the NAO. In that way, the audience had a wider range and the application could be tested better by exposing all its negatives and positives. Because the Android has not been developed on time, as it has been set up on the project plan, the users had to evaluate only the desktop one. The whole process proved very significant as revealed weaknesses that have not been discovered on the development stage.
6 CONCLUSION AND FUTURE WORK

This project proved to be interesting and simultaneously extremely challenging from the first moment. The Python language and the development of an Android application have never been implemented in the past. The use and the knowledge of new tools was much more beneficial that it was expected. It has to be admitted, though, that some of the delays occurred and the prototypes were not exactly ready on the due date that has been decided on the project plan, due to the learning curve of the languages that were adopted in this project.

In terms of the outcome of the project, it should be considered a success as most of the goals that have been set up at the beginning have been achieved. Desktop application covered all the functionalities that have been set up in the initial plan. NAO can be connected directly with the application and through the graphical user interface demonstrate a wide range of its capabilities. However the modules of motion and speech produced successful results every single time that they have been used, as the tests revealed after the completion of each stage, the speech recognition feature still doesn't produce the desired outcome every time that is being used. The reasons, that have been explained already, does not permit to the robot respond accordingly all the times. The desktop application has been developed and tested initially in a Linux environment. One of the objectives that has been set up during the development, was the aspect of making the application able to run on the Windows and the Macintosh environments. After the end of the Android’s development stage, both versions have been developed as well. The environments that the application has been tested and ran successfully were a Windows 7 64-bit system and an OS X Yosemite version 10.10.3.

The Android application which has been the second part of the thesis has covered the functionalities of the motion and the speech. Except from the speech recognition module that has not been implemented, mainly because the lack of sufficient time, all the other targets have been accomplished. Although at the initial plan, one of the objectives regarding the mobile was the establishment of a direct communication between the robot and the application, at the end this target has not been achieved. The version 1.14 of the robot was not compatible with the jNAOqi library therefore a middleware has been used in order a connection between the two can be set up. The Http requests have been the key of providing a solution for a successful connection between the device and the humanoid.

The evaluation that has taken place on the last prototype provided a more spherical view of the desktop application and successfully revealed the weaknesses and the strengths of all the modules that have been developed. As has been discussed, most of the targets have been
achieved, but there is still extra work that can be done to extend the functionalities that the current project has covered.

6.1 Future Work

Future work will allow developers to integrate new modules which will make the robot able to demonstrate more of its capabilities. NAO can support many different kinds of functionalities that can be embedded in the existing applications and make them much more functional. The evaluation process except from the weaknesses and the strengths reveal, has also managed to capture the preferences, which automatically converted to suggestions referred to future work and things that the users would prefer to see in the next versions of the applications. Initially, regarding the desktop application, which more features have been integrated comparing to the one of the mobile's device, for sure there are still things that can be added in the future. First of all, more demonstrations can be included in the relevant section so the user can select from a wider range of activities which the robot will execute accordingly. Secondly, a couple of suggestions would be focused on the motion functionality. The user could have the option to select a number of movements, put them in the order that he prefers and instead of single moves the robot could execute a combination of all the selected actions. In that way, except for the classic single movements that are available in the current version, the user could also create unique combinations of movements around the space. For that improvement, one of the perquisites that would offer a more secure environment for the robot and could count as an addition, would be the existence of a button or a function that the user can stop the robot when is moving at any time. Also, another change that could occur in the movements section has to do with the log display. The voice commands are not recorded one by one in the text box on the right hand side, but all together when the interaction with the robot stops. Next for the robotics lab information, in which the robot starts a conversation with the user and some relevant information given, one suggestion could be these details could be split into smaller parts and become much more specific. For example, at the moment when NAO accepts the word “robots” give a general description of all the robots been in the lab. The new functionality could allow the user to ask for each one separately and learn more special characteristics. Another point, which has been suggested from a large number of the evaluators, was that they would like to be informed, except for the information related to the lab with some other kind of information such as the weather forecast or the news retrieved from a website. The last two suggestions for the desktop would be not related to the existing functionalities, but could integrate into the application
as totally new features. The first is related to the use of the front camera installed on the robot's head. Especially, for the motion module another function could be added in order to display the environment around the robot when it moves and allow users record and collect any data needed. The last but not least, would be the creation of a game. That option could be either a quiz or a mind game in which the robot will play with the user and at the end of the process a score would be assigned to each user. The score's data can be held in a text file, other users will compete as well and at the end somebody would be the winner. From the other side the Android application has much more features that can be added as only the Speech and the Motion has been implemented at the moment. All the suggestions above could be part of the features that could be added in future versions of the mobile approach, but initially the proposal would be the development of the Speech Recognition and the addition of the Demonstration's section.
REFERENCES


APPENDIX A - Evaluation Questionnaire

The purpose of this questionnaire is to evaluate and improve the desktop application of Robotics Lab Demonstrator which was created by Panagiotis Filis for the academic year 2014-2015 at Heriot Watt University. Please read the following questions and answer relatively either by choosing one of the given options or giving a short answer.

PERSONAL DETAILS

1. Are you male or female?
   - Male
   - Female

2. Which is your age group?
   - 17 or younger
   - 18-25
   - 26-35
   - 36-50
   - 50 or older

3. Which is your current occupation?

PREVIOUS EXPERIENCE

4. How much time in total do you use computer devices (pc, laptop, tablet, cellphone) in a daily basis?
   - More than 10 hours
   - 5-8 hours
   - 3-5 hours
   - 1-2 hours
   - None

5. How often do you use desktop (Linux, Mac, Windows) applications?
   - Every day
   - Very Often
   - Regularly
   - Rarely
   - None
6. Do you have any experience with robots?
   ○ My job is related to robots.
   ○ I have heard about them and I have used them sometimes in the past.
   ○ Not that much. I have heard about them though.
   ○ I have no idea.

7. Have you ever used an application related to robots?
   ○ Every day  ○ Many times  ○ Sometimes  ○ One or two times  ○ Never

8. If yes, can you please specify?

   APPLICATION'S INTERFACE

9. How did you find the layout (colours, element's position) of the interface?
   ○ Excellent design. Everything on right position.
   ○ A good design approach but elements' position has to change.
   ○ Elements have the right position but the design can be improved.
   ○ Very wrong layout and design simply indifferent.

10. How easy is to navigate through the application?
    ○ Extremely  ○ Very  ○ Moderately  ○ Slightly  ○ Not at all

11. How easy is it to use the application's interface and explore its functionalities?
    ○ Extremely  ○ Very  ○ Moderately  ○ Slightly  ○ Not at all

12. Have you read the subtitle's messages of each window which are responsible to explain the functionality of each window in detail?
    ○ I have read all of them and they were very helpful.
    ○ I read them only once or twice as some window's functionality was unclear.
    ○ I didn't read them as everything was very clear.
    ○ I have not even noticed that they exist.
CONNECTIVITY

13. How difficult was it to find out how NAO can be connected to the application?
   - Extremely difficult
   - Quite difficult
   - Moderate
   - Quite easy
   - Very easy

14. If it was difficult can you write down the steps you followed to find out how to connect NAO? (If you didn’t face any problems leave the box blank)

15. How useful is the included manual window “How to connect Nao” on the main panel that helps the user to connect Nao with the application?
   - Extremely
   - Quite
   - Moderately
   - Slightly
   - Not at all

MOTION FUNCTIONALITY

16. How precisely did the keyboard commands respond accordingly to the relevant actions?
   - Always
   - Often
   - Sometimes
   - Rarely
   - Never

17. How difficult did you find the voice commands functionality?
   - Extremely
   - Very
   - Moderately
   - Slightly
   - Not at all

18. How many times the robot couldn’t hear which word you used on average to give it a command and asked you back to repeat the phrase?
   - More than 10
   - 6-9
   - 3-5
   - 1-2
   - 0

19. Has the robot executed the right action anytime you used a word to make it move?
   - Every time
   - Almost always
   - So and so
   - Rarely
   - Almost never

20. Can you please mention if you faced any difficulties with the motion functionality(both keyboard and voice commands)?

   [Blank space]
ROBOT SPEECH FUNCTIONALITY

21. How clear was the role of each button of the panel related to speech functionality?
   - Totally
   - Very
   - Moderately
   - Poor
   - Not at all

22. Has the robot responded to all the commands that it has to execute?
   - All of them
   - Almost
   - Some of them
   - Almost none
   - Not at all

23. Can you please mention if you faced any difficulties with the speech functionality (both manual and custom phrases)?

   

ROBOTICS LAB INFORMATION

24. Have you found the existence of the buttons of the website's links useful?
   - Extremely
   - Very
   - Moderately
   - Slightly
   - Not at all

25. How difficult did you find the voice commands functionality?
   - Extremely
   - Very
   - Moderately
   - Slightly
   - Not at all

26. How many times the robot couldn't hear which word you used on average to give it a command and asked you back to repeat the phrase?
   - More than 10
   - 6-9
   - 3-5
   - 1-2
   - 0

27. Has the robot given relevant information any time you asked for?
   - Every time
   - Almost always
   - So and so
   - Rarely
   - Almost never

28. Did you expect any other kind of information that hasn't been included to the induction that the robot presented?

   

4
29. Can you please mention if you faced any difficulties with the speech interaction you had with the robot through the induction to the lab that you have attended?

FUNCTIONALITIES

30. Which part have you found most difficult to use? (Please choose more than one answer if needed)
- Motion
- Speech
- Settings
- Demonstrations
- Robotics Lab Information

31. Which part have you found the easiest to use? (Please choose more than one answer if needed)
- Motion
- Speech
- Settings
- Demonstrations
- Robotics Lab Information

32. Can you please define the reason if you have found any of the functionalities difficult or confusing (motion, speech, speech recognition)

ERROR REPORT

33. How often did the application freeze or crash?
- Extremely often
- Very often
- Often
- Just once or twice
- Not at all often
34. If the application have crashed can you please describe in which action that occurred and what exactly happened


**COMMENTS**

35. What you would suggest can be added to improve the desktop application?


36. What you would suggest can be removed from the desktop application as it is not necessary?


37. Other comments


38. Overall, are you satisfied with the application, dissatisfied with it, or neither satisfied nor dissatisfied with it?

- [ ] Extremely satisfied
- [ ] Somewhat satisfied
- [ ] Neither satisfied nor dissatisfied
- [ ] Quite dissatisfied
- [ ] Extremely dissatisfied
APPENDIX B - Link to Python and Java Code

The link below includes all the code that has been used for the development of both (Desktop and Android) applications which covered in this dissertation.

https://github.com/panos5/Panagiotis_Filiotis_Thesis