ECA BASED CONTROL SYSTEM FOR HOME AUTOMATION

Claudio Tesei
Master of Science
Computer Science
December 2014
School of Computer Science
Reykjavík University

M.Sc. RESEARCH THESIS
ECA based Control System for Home Automation

by

Claudio Tesei

Research thesis submitted to the School of Computer Science at Reykjavík University in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science

December 2014

Research Thesis Committee:

Leonardo Mostarda, Supervisor
Title, University of Camerino

Marta Larusdottir
Title, Reykjavík University
ECA based Control System for Home Automation

Claudio Tesei

December 2014

Abstract

This Thesis is part of the result of a larger and complex work that me and Davide Angelici are carrying out with other members of the team. Consequently, to have a complete view it is necessary to read also the MSc Thesis of Davide Angelici. We claim to have the ability to create a new product that make the Domotic easier and low cost and in this document we provide the evidence to prove that. Through an elaborate Business Plan we touch points as the market, the innovative value, the marketing strategy and some calculation to explain our economical plan; at the same time we introduce our innovative product composed by eight packet of easy installation with the aim of making smarter some components inside the house. After having introduced a clear overview of the system, we will pass at the design phase, in which more detail are given to address the reader towards the implementation. Next the Thesis will be focused on the Central Unit (the main topic of this document), the software employed to develop it and the technique used to write the program. At the end will be taken into account some considerations about the work made until now and what remain to develop to achieve a product ready for the market.
Dedicated to the addictions...
Acknowledgements

Thanks at .......
# Contents

List of Figures xii

List of Tables xv

1 Introduction 1

2 Business Plan 3

## 2.1 Overview

- The opportunity: home automation easy and low-cost 3
- The product 3
- The team 4
- The business model 4
- Roadmap and economic projections 4

## 2.2 The market and the competitive environment

- Home Automation: the smart home 4
- The standards are numerous 5
- A great opportunity 7
- The customer asks for simplicity and flexibility 7
- Feedback is Important 8
- As it has developed so far the market 10
- How it will develop the global market 11
- The main players 12

## 2.3 The product, process or invention: the value proposition

- How has it started 16
- The product 17
- Placing on the market 19

## 2.4 The marketing plan

- Product 20
- Price 22
2.4.3 Sales channels (Placement) ........................................ 23
2.4.4 Promotion ............................................................... 24

2.5 Operational Plan ....................................................... 24
2.5.1 Roadmap ................................................................. 24
2.5.2 Compliance of the system .......................................... 25
2.5.3 Make or buy? ............................................................. 25
2.5.4 Operation characteristics ........................................... 25
2.5.5 Technical Assistance ................................................ 27

2.6 Organizational Structure ............................................. 29
2.6.1 Value chain ............................................................. 29
2.6.2 Organization chart of the company .............................. 29

2.7 The economic and financial plan ................................. 30

3 Software Design ......................................................... 33
3.1 Purpose of the System Design Document ....................... 33
3.2 General Overview ....................................................... 33
3.3 Design Considerations ................................................ 34
3.3.1 Goals and Guidelines .............................................. 34
3.3.2 Development Methods .............................................. 35
3.3.3 Architectural Strategies ............................................ 35

3.4 Hardware Architecture Design .................................... 35
3.4.1 Logical View ........................................................... 35
3.4.2 Hardware Architecture ............................................. 36

3.5 Software Architecture Design ..................................... 40
3.5.1 Central Unit ............................................................ 42
3.5.2 Packages ............................................................... 42
3.5.3 Mobile App ............................................................ 42
3.5.4 Workstation ............................................................ 42

3.6 Internal Communications Architecture Design .............. 43
3.6.1 Communications Hardware Architecture .................... 43
3.6.2 Communications Software Architecture ..................... 45

3.7 DB Design ................................................................. 46
3.7.1 Central Unit Database .............................................. 46
3.7.2 Web Page Database ................................................. 48

3.8 User Machine-Readable Interface ............................... 48
3.8.1 Inputs ................................................................. 49
3.8.2 Outputs ............................................................... 50
4 Central Unit Implementation 51

4.1 Architecture overview ........................................... 51
4.2 Install the environment on the Raspberry Pi .................. 52
  4.2.1 The OS .................................................. 52
  4.2.2 LAMP ............................................... 53
  4.2.3 Java JDK ........................................... 54
  4.2.4 SPI .................................................. 54

5 Control Software 59

5.1 State of the Art .................................................. 59
5.2 Kaalisy Rules .................................................. 61
  5.3 The ECA language ........................................... 61
    5.3.1 ANTLR ............................................ 61
    5.3.2 Lexer and Parser .................................... 62
    5.3.3 The Language in B.N.F. ............................. 63
  5.4 Overview of the components and their requirements ............... 66
  5.5 Server Communicator .................................... 67
  5.6 DataBase Communicator .................................. 68
  5.7 Package Communicator ..................................... 68
    5.7.1 Auto-installation .................................... 69
  5.8 Kaalisy Rules Engine ..................................... 70
    5.8.1 Overview of the component .......................... 71
    5.8.2 The semantic value .................................. 73

6 Conclusion 77

6.1 Future work .................................................. 78
  6.1.1 Improvement ........................................... 78
  6.1.2 Developing ........................................... 78
  6.1.3 Testing ............................................... 79

Bibliography 81

Appendices 85
List of Figures

2.1 The smart home ................................................. 5
2.2 Do you have an home automation system? ............................. 9
2.3 Would you like to have it? ......................................... 9
2.4 Why you do not have it? .......................................... 10
2.5 Home automation system in Italy 2005-2013 ......................... 11
2.6 Expected development for the global market of home automation .......................... 12
2.7 "UnicamSolar" developed for the University of Camerino (2013) .................. 17
2.8 Logic view of "Kaalisy Domotics" system ........................... 17
2.9 Dwelling with a Kaalisy Domotics system .......................... 19
2.10 Differentiation diagram of the products present on the market ......................... 19
2.11 Costs of electronic components required for each package ................. 22
2.12 Selling prices of the packages ................................... 23
2.13 Selling prices of competitor’s products ............................ 23
2.14 Roadmap of the product’s development ................................ 24
2.15 Typical operations of Kaalisy Domotics ............................ 26
2.16 Types of assistance .................................................. 28
2.17 The primary and support activities of Porter’s Value Chain for Kaalisy Domotics ................................................. 29
2.18 Organization of Kaalisy Domotics ................................... 30
2.19 Hypothesis of market penetration ....................................... 31
2.20 Type of configuration expected ........................................ 31
2.21 Plan of sales by product .............................................. 31
3.1 Logical view of Kaalisy system ........................................ 36
3.2 Architecture view of Kaalisy system ..................................... 37
3.3 Raspberry-Pi ....................................................... 37
3.4 Arduino Pro Nano ................................................... 38
3.5 ATtiny85 microcontroller ............................................. 39
3.6 Software view of Kaalisy system ........................................ 41
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.7</td>
<td>Mesh Network view</td>
<td>43</td>
</tr>
<tr>
<td>3.8</td>
<td>Transceiver nRF24L01+</td>
<td>44</td>
</tr>
<tr>
<td>3.9</td>
<td>ZM3102 Transceiver</td>
<td>44</td>
</tr>
<tr>
<td>3.10</td>
<td>E-R Central Unit Database structure</td>
<td>46</td>
</tr>
<tr>
<td>3.11</td>
<td>E-R Central Unit Database</td>
<td>47</td>
</tr>
<tr>
<td>3.12</td>
<td>E-R Web page Database</td>
<td>48</td>
</tr>
<tr>
<td>4.1</td>
<td>Central Unit Software layers</td>
<td>51</td>
</tr>
<tr>
<td>4.2</td>
<td>Raspbian OS logo</td>
<td>52</td>
</tr>
<tr>
<td>4.3</td>
<td>View of Raspbian configuration tool</td>
<td>53</td>
</tr>
<tr>
<td>4.4</td>
<td>Java version</td>
<td>54</td>
</tr>
<tr>
<td>4.5</td>
<td>Serial Peripheral Interface bus</td>
<td>55</td>
</tr>
<tr>
<td>4.6</td>
<td>GPIO pinout Raspberry-pi</td>
<td>56</td>
</tr>
<tr>
<td>4.7</td>
<td>Pinout nRF24L01+</td>
<td>56</td>
</tr>
<tr>
<td>4.8</td>
<td>Arduino&lt; – &gt; Raspbrry-pi communication test</td>
<td>57</td>
</tr>
<tr>
<td>5.1</td>
<td>Antlr logo</td>
<td>62</td>
</tr>
<tr>
<td>5.2</td>
<td>Processing source code</td>
<td>63</td>
</tr>
<tr>
<td>5.3</td>
<td>CSW components view</td>
<td>66</td>
</tr>
<tr>
<td>5.4</td>
<td>CSW loading from DB</td>
<td>66</td>
</tr>
<tr>
<td>5.5</td>
<td>CSW ocuring event</td>
<td>67</td>
</tr>
<tr>
<td>5.6</td>
<td>CSW message from Server</td>
<td>67</td>
</tr>
<tr>
<td>5.7</td>
<td>Central Unit Software view</td>
<td>69</td>
</tr>
<tr>
<td>5.8</td>
<td>Auto-installation sequence diagram</td>
<td>70</td>
</tr>
<tr>
<td>5.9</td>
<td>ECA Rules Engine components view</td>
<td>71</td>
</tr>
<tr>
<td>5.10</td>
<td>Variable Class</td>
<td>72</td>
</tr>
<tr>
<td>5.11</td>
<td>Rule Class</td>
<td>73</td>
</tr>
<tr>
<td>1</td>
<td>Expected Revenue by product</td>
<td>87</td>
</tr>
<tr>
<td>2</td>
<td>Plan of marketing costs</td>
<td>87</td>
</tr>
<tr>
<td>3</td>
<td>Investment Plan</td>
<td>87</td>
</tr>
<tr>
<td>4</td>
<td>Plan of staff costs</td>
<td>88</td>
</tr>
<tr>
<td>5</td>
<td>Income Statment</td>
<td>88</td>
</tr>
<tr>
<td>6</td>
<td>Balance sheet - assets</td>
<td>89</td>
</tr>
<tr>
<td>7</td>
<td>Balance sheet - liabilities</td>
<td>89</td>
</tr>
<tr>
<td>8</td>
<td>Statement of cash flows</td>
<td>89</td>
</tr>
<tr>
<td>9</td>
<td>Ratios</td>
<td>90</td>
</tr>
</tbody>
</table>
# List of Tables

2.1 Standards of communication used in home automation ............... 6  
2.2 List of competitors in Italy ........................................... 14  
2.3 List of competitors in the world ....................................... 15  
2.4 Specification of the packages ........................................... 18  
2.5 Functions of the packages .............................................. 21  

3.1 Raspberry-Pi Technical Specifications ................................. 38  
3.2 Arduino Pro Nano Technical Specifications ......................... 39  
3.3 ATtiny85 Technical Specifications .................................... 39  
3.4 nRF24L01+ Technical Specifications .................................. 44  
3.5 ZM3102 Technical Specifications .................................... 45  
3.6 Central Unit Database Identification .................................. 47  
3.7 Web Page Database Identification .................................... 48
Chapter 1

Introduction

This thesis describes something more than a project, it describes an Idea that aims to become an enterprise. Such Idea is: "Make the Domotic Easy and Low Cost for everyone". Probably at this point the reader is already laughing, and start to imagine which are the solution that will be proposed, but do not worry we are extremely serious, and especially we are conscious of what we will face. Nowadays the home automation market aims to a specific target, the new building. Infinite solutions already exist to make a smart home, of course the smart home market is not new, but all the present products have two points in common: they are invasive and expensive. The ones building a new house does not care about the invasivity, and compared to the total price of the dwelling the domotic system seems even cheap. Thereby we felt that this kind of offers does not fit with the context in which we are, especially in Italy. Fifteen years ago probably was a good business but actually it takes a little part of the people interested in this kinds of products. Then, we decided to create a product addressed to that part of market which no one has considered yet. This was made possible thanks to the evolution of the technology; developing such a solution five years ago was not possible at an affordable price. Our System is composed by eight packages that can be considered plug&play: because of this we can claim the easiness. One Central Unit, that has to be present in every system to connect all the packages with an Interface available on the SmartPhone and the Computer, and other 7 different packages aim to make smart the principal appliances present in the house. Evolve such an Idea to achieve a place in the market is an elaborate route that comprehend several steps, first of all is gather a team. Often happens that an Idea does not see the light, because the people does not arrive at the end of the realization process. However this is not our concern, me and Davide Angelici are the people behind this Idea and we together have group an amazing team composed by another three persons. Anyway just me and Davide have redact the documentation present in this and his thesis that originally was designed
to be just one document. The first part is the Business Plan and in both thesis are identical because was unsplitable and necessary to give a mean at the other part. We decided to introduce the System of Kaalisy Domotic with the BP because the reader have to reach a sufficient knowledge of the market to realize which is the real value of the product that we want to create. The process, that bring an Idea to its realization and future listing, is paved with problems. The Business Plan is addressed to avoid these problem and allow to create a product which are an no end in itself. Then we started with an analysis of the domotic market tapping the trend, the competitors and the buyers. Consequently we begin to introduce our product, but never going too deeply into the explanations explanation and always in a market context. Once that the reader has understood the product, and why we are so convinced that it has market, we will introduce the marketing subject: which are our strategies to let the people know? How we intent to sell it? Everything is described along this part, nothing can be left to the chance. Probably, some paragraph like this may appear a bit offtopic in a MSc thesis of Computer Science, but we felt that the reader has to know all the work that we are doing around this project, not only the implementation part. After the marketing, it will be revealed how we want to organize the work and the structure of the enterprise. At the end of the Business Plan there is some numbers to show the calculation made in the financial plan. Next to the BP should be present the Software Requirements but, as written above, this part is in the MSc thesis of Davide Angelici. However it shows all the functional and nonfunctional requirement that the system must respect. Then, according with what is specified on the SR we begin to design the project of the entire system: this part is described in the Design Requirements, starting from an overview of the system and going deeply until all the hardware utilized in this system has been described. The last part before the conclusion consists in the mere implementation: this is the enjoyable part for the computer scientist point view. To demonstrate that we are not just pulling words out of thin air, we described the implementation of the Central Unit, the most complex package that our offer includes. Now that we clarified what this document is about, we just have to wish the reader an entertaining reading.
Chapter 2

Business Plan

2.1 Overview

2.1.1 The opportunity: home automation easy and low-cost

The home automation systems are capable of controlling lighting, heating, shutters and windows of houses. Kaalisy Domotics aims to bring home automation systems in Italian homes, allowing anyone to be able to realize a customized home automation system at an affordable price. It is not present on the market a solution that offers a service easy to install yet, which does not require work on the electrical system and it also allows you to interact with devices "not smart". Kaalisy Domotics offers an easy to install system which can be purchased at an affordable price and does not require the intervention of specialized technicians. Moreover, with this system is possible to interact with devices "not smart".

2.1.2 The product

Kaalisy Domotics is an innovative home automation system, thanks to its own communication protocol over Wi-Fi, it allows to control easily and intuitively the devices using an app on your smartphone or via an user friendly web interface, which allow the end user to create scenario, for example ("if this happens then do this"). Among the products on sale it is possible to find switches, light adapters, wall plug adapters, thermostat, door opening and sensors.
2.1.3 The team

The entrepreneurial team is both skilled and experienced in Information Technology and Electronics and business development and management.

2.1.4 The business model

The business model adopted is an e-commerce model. The products purchased online will be sent to the buyer by express delivery service carried out by partner curriers company.

2.1.5 Roadmap and economic projections

Before the start of the commercialization of the products a phase of industrialization and standardization of six months is necessary. Moreover, each product must obtain the CE certification. The economic projections, shows that the break-even point is reached during the second year and is followed by a rapid growth in both revenues and profit with ROI and ROE in double digit from the second year.

2.2 The market and the competitive environment

2.2.1 Home Automation: the smart home

Enter at home and turn on the lights, open the blinds, select your favourite music and turn on the TV automatically done by a computer when it recognize your presence, it is no longer a futuristic idea but is now a reality (Figure2.1). Nowadays, home automation systems are able to control lighting, heating, blinds and windows, also acting independently. However, these systems should work together, collaborate, communicate between each other to make life easier for those who use them, increasing comfort and safety, leading to an energy saving and providing modularity and flexibility. The first home automation systems were fairly rudimentary and the final result was really unattractive. Current systems use innovative technologies and provide to the end user a complete and functional product. This development became possible thanks to the introduction of "smart" domestic components, i.e. circuit breakers, thermostats, bulbs, microwave ovens which are able to communicate with the outside world. The terms "smart home" and "home automation" are now commonly used to refer to devices connected to the internet and house’s
equipment able to operate independently. In reality, the full realization of the smart home needs the integration and collaboration of various technologies and services, all working together.

![Figure 2.1: The smart home](image)

On the market there are various intelligent devices, which use different communication protocols. This constitutes a real obstacle in the realization of a fully integrated smart home. In response to this problem, some companies have developed standard communication protocols, in order to simplify the communication between smart devices available on the market.

### 2.2.2 The standards are numerous

The most common standards of communication protocols are reported in Table 2.1. This are currently used for the communication with automation components.

A home automation system compatible with some of those protocols, definitely offers advantages in terms of flexibility, modularity and expandability compared to systems that use only proprietary communication protocols (i.e. that cannot be used by others). For example, Revolv, home automation central unit that is able to communicate with a variety of smart devices, is compatible with a high number of communication protocols (well 7), unlike a product like Juicy iAX, that even if it has a good compatibility, is not able to offer the same standards.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Diffusion</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-Wave</td>
<td>Wireless communication protocol, designed for home automation, in particular for control the smart devices in residential and commercial environments.</td>
<td>There are more than 900 different products certified by the Z-Wave Alliance. These products cover all major sectors market for control devices in residential and commercial environments.</td>
<td>2008</td>
</tr>
<tr>
<td>KNX</td>
<td>Communication protocol exclusively used for building automation.</td>
<td>Open standard, which has now connected more than 300 companies around the world.</td>
<td>2002</td>
</tr>
<tr>
<td>ZigBee</td>
<td>Industry standard for wireless networks.</td>
<td>Association of more than 230 companies that are driving the global development of this technology. First ZigBee products came on the market at the beginning of 2005.</td>
<td>2002</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>&quot;Wireless technology standard for exchanging data over short distances&quot;</td>
<td>Technology used in all modern smartphone, often used in systems Home automation systems of small dimensions.</td>
<td>1999</td>
</tr>
<tr>
<td>X-10</td>
<td>Open industry standard for the communication between electronic devices for home automation.</td>
<td>Although there are alternatives with higher bandwidth, including KNX (Konnex), INSTEON, BACnet and LonWorks, X10 remains popular in domestic environment with millions of units in use worldwide.</td>
<td>1975</td>
</tr>
</tbody>
</table>

Table 2.1: Standards of communication used in home automation
2.2.3 A great opportunity

The currently available solutions used to transform your home into a smart environment are essentially of two types. The first type requires the installation of the smart devices and the use of a "central unit" (e.g. Revolv, Juicy iAX) which allows the interaction with the devices, usually via a mobile app. The main strengths of this solution are the easy installation and the ability to interact with different devices without the necessity to do any wiring. The disadvantage is the relatively high cost of the smart components, compared with the ordinary appliances findable in most of the homes. The second option allows the use of smart devices and house’s appliance without communication protocols, but it is required the modification of the house’s electric system, changing its structure and performing a complete wiring of the devices that you want to "make smart". This solution allows the use of components such as switches, bulbs, thermostats, blinds and shutters that are not necessarily smart, but the invasive intervention required on the electrical system increases the overall cost. Examples of this solution are those offered by companies such as BTicino, AtHome, HiSystem all characterized by a fairly high price. However, a solution that tries to combine the advantages of the previous two solutions by removing their disadvantages, it is not yet on the market. For example a solution that offers a product that is easy to install, does not require work on the electrical system and at the same time it is also addressed to non-intelligent devices. Such a solution would seriously reduce the costs required for the construction of a home automation system and it is precisely this possibility that we intend to achieve with our system.

2.2.4 The customer asks for simplicity and flexibility

From the customer’s point of view, a smart home is much more than a simple high tech gadget. The house is a long-term commitment and a significant investment, and a smart home involves changing the way you live. Previous innovations such as television or personal computer produced significant social changes but the smart home will create even more significant changes. The most of the people are looking for an easy to use system that allows a remote management and the creation of scenarios (e.g. Scenery night: closing the front door, alarm activation, switching off lights). But we observed that there were often too many information that confuse the buyer. Therefore it is preferable to reduce the amount of information and to reduce its complexity in order to create a more accessible system. Too many interfaces to manage the system also confuse the end user, this is why one prefers simple interfaces such as touch-screens, which enable immediately the user to command and activate different scenarios or series of functions. Solutions
based on open standards, flexible and easily programmable systems are required. These provide the ability to the user to change the system or service without the need to modify the electrical system.

### 2.2.5 Feedback is Important

Feedback is a really powerful "tool" to help people to be updated and improve their lack, it can be used also to know how other people perceive specific performances. We retain extremely important know if our solution could in someway satisfy the needs of future potential customers. Then we decide to retrieve these feedbacks adopting the only simple method that all the time works, the questionnaire!

**What to ask?**

To get accurate feedback, first of all the questions has to be simple and clear, avoiding the user to misunderstand the meaning of it. Secondly the questionnaire does not have to cover many topics, few questions referred to few concept. Once that we had clear this few basic rule, we fixed the goal of the questionnaire, in other words, what the result will emerge after have analyzed it. According to the embrional state of the project it was useless try to get a feedback about kaalisy, because we did not have any prototype or examples, then we decide to try to understand if our solution, which is based on four main aspects (non-invasive, easy-to-install, cheap, reusable), can and answer to the problematic of the potential users. We basically wanted to know if they know about home automation system and if they are interested on it and why they do not have it. At the end of the questionnaire we present a custom home automation configuration and we asked how much they would have spend for have it.

We try to target different category of person based on their age in order to be have more realistic data.

**Result**

The first information extrapolated from the questionnaire was the number of people that currently have an home automation system installed at their place.
Figure 2.2: Do you have an home automation system?

As we expected almost nobody has an home automation installed at their apartment, than an obvious question to ask was to see if the would like to have it and has the Figure2.3 shows that a large number would like to have it.

Figure 2.3: Would you like to have it?

Than we asked why you do not have it even if you would like to have it installed at your place? The result coming from this question was really important to understand if Kaalisy was able to answer to their needs. And as we can see from the Figure2.4 the first motivation is because they are expensive, they do not own the apartment and because it is invasive. Moreover we asked how much do they would like to spend for having a system which is able to control all the lights, thermostats, doors and video surveillance. It come out that the are willing to spend around 500 euro.
In conclusion we can be confident with the results obtained from the questionnaire, as we can see the strength of Kaalisy lays on the needs of the users, and this is a really important aspect because it means that we are working in the correct direction. Moreover we are able to present a solution which is non-invasive, reusable and it fits perfectly the budget of the people.

At the end we interviewed almost 151 people of different age, 50% of them were people younger than 35 year.

2.2.6 As it has developed so far the market

The CERP, an organization that gather the research projects of the European Union in the field of Internet of Things, shows the home automation and intelligent buildings among the most promising areas of application. The use of wireless communication technologies (ZigBee, 6LoWPAN, etc.) allows you to connect to each other objects within a building providing two-way communication. In the past, home automation technologies were mainly used in offices and luxury homes, but the technological development and the permanent reduction in costs now make these technologies accessible to a growing part of the population. The features offered are varied and constantly evolving. For example, you can use the temperature and humidity sensors to collect the data needed to optimize the use of heating and air conditioning, ensuring comfort and at the same time reducing energy consumption. An important role is also played in helping the people, given that the home automation solutions can be used to monitor the activities of people within the home, providing assistance to elders or individuals with disabilities in the performance of activities of daily living and, in case of need, make alarm. In Italy, the home automation market developed with the growth trend (data Assodomotica) at around 30% per annum. The trend has been to create home automation systems in new homes or renovated. In our
country about 300,000 homes per year are built and around 700,000 are restructured with the complete renovation of the electrical system. Home automation systems have grown from 10,000 in 2005 to over 100.00 in 2013 (Figure 2.5). The total value of these implants went from 42 million euro in 2005 to 440 million euro in 2013. These figures include both basic electrical systems (normally included in the offers of new buildings), and advanced systems, with high levels of customization. Assodometica identifies four market segments. The first referred to "advanced applications" covers housing and extended luxury homes. The advanced systems require a custom project, which involves, in addition to the installer, designer and architect. The second segment of the market concerns the systems made by the installer directly on the final user. The third segment is the one of "Basic systems" provided in the offer by the construction companies and the fourth is the one regarding applications which are intended for the elderly and disabled. The fastest growing segments are the advanced applications and basic systems. As for the last one, a strong pushes as been given by the growing sensibility regarding energy savings.

![Figure 2.5: Home automation system in Italy 2005-2013](image)

### 2.2.7 How it will develop the global market

According to ABI Research the global market of home automation is trending upwards with significant growth expected over the next few years. With 9 billion USD in 2014, the market is expected to reach 14 billion by 2018. The North American market will remain solid over the forecast period, increasing from 5 billion to 7 billion. North America currently holds 64 percent of the market, but its percentage is expected to drop to 50 percent by 2018 (Figure 2.6). Growth is fueled by a constellation of factors, including a recovering economy and new entrants in the market, according to Adarsh Krishnan, senior analyst at ABI Research. Some of the barriers that limited the growth of the market, such as the lagging economy and high costs, have now been removed. The integration of
security systems with home automation systems is promoting the wider adoption by user, and technological improvements in general, continues to make the market more attractive to user and suppliers. The constant improvement of wireless chipsets, in addition to their decreasing costs, was also a key factor in promoting their adoption. The automation technologies that should enjoy a pronounced growth during the forecast period include contact sensors, motion sensors, wireless sensors, keypads and control panels.

Figure 2.6: Expected development for the global market of home automation

Some of the features that are added to the home for safety reasons are expected to get substantial gains. "We expect to see the most growth in door locks, thermostats and smart plugs", said Krishan. The demand for a universal solution, easy to install and integrated with several subsystems in a home, led to a phase of explosive growth in the industry.

2.2.8 The main players

To understand better the potential of our product we can compare it with the solutions that are available on the market. After a research made on internet it was possible to create a list of competitor, it was decided to divide that list geographically and by type of service offered, so as to highlight both the potential and the gaps of each proposal on the market.
### Direct Competitor

### Indirect Competitor

**In our region**

<table>
<thead>
<tr>
<th>Competitor</th>
<th>Product</th>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seav</td>
<td>SEAV DOMUS</td>
<td>- Web interface design, for home automation, system</td>
<td>- Wiring of the system by a qualified technician</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Compatible with Radio, technology 433.92 MHz</td>
<td>- Proprietary communication protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Invasive modification of the electrical part of the existing</td>
</tr>
<tr>
<td>I.E.T domotica e automazione</td>
<td></td>
<td>- 30 years in the market</td>
<td>- Invasive intervention and modifies electrical system</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Compatibility standard KNX</td>
<td>- Web Page unintuitive</td>
</tr>
<tr>
<td>Domsolution</td>
<td>Smart Solution</td>
<td>- Modularity</td>
<td>- Proprietary communication protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Ability to create dedicated scenarios</td>
<td>- Invasive intervention and modifies electrical system</td>
</tr>
</tbody>
</table>
In Italy

<table>
<thead>
<tr>
<th>Competitor</th>
<th>Product</th>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
</table>
| atHome     | AtHome@ START   | - Modularity  
- Ability to create dedicated scenarios | - Invasive intervention and modifies electrical system  
- It does not support standard as KNX, Z-Wave and ZigBee |
| Auratec    | Auratec System  | - User friendly interface  
- Compatible with the KNX standard  
- Ability to create scenarios  
- Web Page very attractive | - Invasive intervention and modifies electrical system  
- Unique system lacks the presence of individual packages |
| Bpt        | HOME SAPIENS    | - High design  
- Interactive Web Interface  
- Leader nel settore | - Invasive intervention and modifies electrical system  
- The need for a total restructuring of the electric system  
- Shortly versatile for electrical installations already exist |
| BTicino    | MyHome          | - BTcino Configurator  
- High design  
- Modularity  
- My Home Web: remote access for configuration and monitoring home automation system  
- Compatibility with all appliances and devices BTcino | - Expensive  
- Invasive intervention and modifies electrical system  
- Lack of a Web interface  
- Lack of an app (android, iOS)  
- Invasive, direct intervention in the electrical system |
| Vimar      | By-me           | - High Design  
- Easy to Use  
- BUS system | - Invasive intervention and modifies electrical system  
- Invasive intervention in the electrical system |
| Feel3      | Vivimat         | - Availability of different packages: House, Shops, ...  
- ModBus protocol | - Lack of a Web interface  
- Lack of an app (android, iOS)  
- Invasive, direct intervention in the electrical system |
| HiSystem   | HiControl       | - HiVision: Interactive Web interface  
- Compatible with VoIP devices  
- Compatible with standard ModBus, Dmx, Dali, LonTalk, KNX  
- Partnership with RISCO Group and Atec | - Invasive, direct intervention on the electrical system  
- Web Page unintuitive |
| Omniabit   | Juicy iAX       | - Modularity  
- Plug&play  
- Web Page for monitoring and configuring home  
- Compatible with standard Z-Wave, KNX, MyHome Bticino | - Inability to automate electronic devices without communication protocols |

Table 2.2: List of competitors in Italy
In the world

<table>
<thead>
<tr>
<th>Competitor</th>
<th>Product</th>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revolv</td>
<td>Revolv</td>
<td>- Plug&amp;play System,</td>
<td>- Compatible with few smart devices</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Easy to configure,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Low cost (299$),</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Application for Smartphones intuitive,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Availability of a Community,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Nice Design</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Low cost ($99),</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Plug&amp;Play,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Easy to Use,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Strong Community,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Modularity,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Compatible with Z-Wave ZigBee and Wi-Fi devices,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Arduino Compatible</td>
<td></td>
</tr>
<tr>
<td>SmartThings</td>
<td>Hub</td>
<td></td>
<td>- Compatible with few smart devices</td>
</tr>
</tbody>
</table>

Table 2.3: List of competitors in the world
The Table 2.3 it was made collecting all the information regarding one company, than analyse them and comparing with precise parameters. It was decided to divide all the characteristics founded into two main categories (Strengths, Weaknesses). For list the weaknesses and the Strengths it was chosen the following parameters:

- Cost (is expensive?)
- Invasive (it require the modification of the house’s electrical system?)
- Compatibility (it can work with other product?)
- User-friendly (It provides an intuitive user interface?)
- Easy-to-use (it is possible to install it by yourself?)

We use these parameters according to the characteristic of our product, it that way it is possible to have a clear vision of what can do kaalisy Domotics.

2.3 The product, process or invention: the value proposition

2.3.1 How has it started

It all started from our personal desire: to control the lights using a smartphone in the house. Unfortunately, there was no possibility to change and modify the structure of the electrical system. As we did not find any easy and accessible way in the market, we decided to solve the problem by creating our own home automation system that uses open-hardware platforms. Taking note of the satisfactory results obtained, we decided not to stop there, but we continued to add new components in our system to increase the control on the house. Also, we decided to work from the software side, and have an overall improvement in terms of "Human-Computer-Interaction". We have made the system customizable, adaptable to every need of the user. Then, gathering feedback from friends, relatives, colleagues and academics, we found that our needs matched with need of many others so the desire to transform our system into a business idea was born.

We have already had the opportunity to concretely demonstrate our technical skills in a previous project, developed as part of our studies at the University of Camerino. "Unicam Solar" Figure2.7 is a hardware/software system for monitoring power control of photovoltaic panels on web and on mobile devices.
2.3.2 The product

Our system, "Kaalisy Domotics", offers several packages that are easy to install and at an affordable price. This will allow users to create a modular system that fully adapts to their needs and grows with their requirements. There will even be the possibility to integrate the packages of other companies into our system, thanks to the use of the Z-Wave protocol. The initial basic product includes a small number of packages, enough to cover the main control requirements of the housing unit. Later the offering will be expanded and diversified, always keeping an eye on the development of the market and technology to avoid ending up with the product becoming obsolete.

The system (Figure 2.8) consists in a central unit connected to the internet, which allows using and programming all packages inside the house through a user-friendly interface on smartphones (app) or web browser. The packages (listed below) are plug & play, easy to install and it does not require any intervention on the electrical system. Once installed in the house, the packages are automatically detected by the system. Then, by accessing to
the user interface configuration, the user will be able to watch a quick tutorial that will explain how to set-up and use the features of the packages.

<table>
<thead>
<tr>
<th>Installation</th>
<th>Communication</th>
<th>Function</th>
<th>Dimension</th>
<th>Alimentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central Unit</td>
<td>- Connect to the router via ethernet cable</td>
<td>- Wireless via proprietary protocol (used by our packages), - Z-Wave</td>
<td>Server</td>
<td>60x10x20 mm</td>
</tr>
<tr>
<td>Bulb adapter</td>
<td>- Screw to E27 plug before bulb, - Install, connecting the cables of the two phases in case of neon, lampadare, etc.</td>
<td>- Wireless via proprietary protocol (used by our packages)</td>
<td>- Remote switch</td>
<td>30x30x50 mm</td>
</tr>
<tr>
<td>Switch</td>
<td>- Replace the present switch, or, - Cover the present switch</td>
<td>- Wireless via proprietary protocol (used by our packages)</td>
<td>- Switch</td>
<td>110x70x20 mm</td>
</tr>
<tr>
<td>Wall Plug Adapter</td>
<td>- Connect between the wall plug and the appliance</td>
<td>- Wireless via proprietary protocol (used by our packages)</td>
<td>- Monitor appliance consumption, - Remote switch to appliance</td>
<td>40x40x50 mm</td>
</tr>
<tr>
<td>Thermostat</td>
<td>- Replace with the present thermostat</td>
<td>- Wireless via proprietary protocol (used by our packages)</td>
<td>- Adjustment manual / remote / programmed of the temperature</td>
<td>120x60x40 mm</td>
</tr>
<tr>
<td>Door Opening</td>
<td>- Fix to the wall, - Connecting with the lock control cable (it can be usually find in the handset)</td>
<td>- Wireless via proprietary protocol (used by our packages)</td>
<td>- Remote opening door</td>
<td>70x50x20 mm</td>
</tr>
<tr>
<td>Sensor</td>
<td>- Fix to the wall</td>
<td>- Wireless via proprietary protocol (used by our packages)</td>
<td>- Remote Control of the temperature and humidity, - Detects: Gas spill, flood, fire principle</td>
<td>140x40x30 mm</td>
</tr>
<tr>
<td>Surveillance Camera</td>
<td>- Fix to the wall</td>
<td>- Wi-Fi</td>
<td>- Remote video monitoring, - Motion detection</td>
<td>80x80x80 mm</td>
</tr>
</tbody>
</table>

Table 2.4: Specification of the packages
The packages can be controlled ("Switch", "Shut Down", "Open", "Increase", etc...) Smart-phone App (downloadable from the Play Store or the App Store), that also allows to control the system by voice. The scenarios (i.e. "If the camera detects motion then turn on the lights and raise the temperature to 20° C inside the room") are programmable on the simple web interface through the browser. The interface design is very nice, aimed to be easy-to-use.

2.3.3 Placing on the market

Competitors and our solution have been mapped in the differentiation diagram of Figure 2.10 in which the dimensions considered are the method of communication (ranging from wired solutions into completely wireless solution) and the completeness of the product (offer complete - control + components - or only controller).

Wireless solutions that currently exist are not complete as wired systems because not all switches, light bulbs, and appliances can be operated wirelessly. The wireless solutions
available on the market require the purchase of a Central Unit compatible with the most frequently used wireless communication protocols (Zigbee, Z-Wave, etc...). Therefore the research, from different companies, of the components (switches, bulbs locks, appliances, etc...) able to communicate with the central unit purchased, discourages many buyers because it requires technical competence and a great deal of time to use the "do it yourself home automation system. Kaalisy Domotics is placed in a segment of the market of home automation still empty, offering a complete solution easy to install and low cost thanks to wireless technology.

2.4 The marketing plan

2.4.1 Product

The product of Kaalisy Domotics (listed below) is aimed at users that are interested to make their own home "smart" in a simple and economical way. The solution may be purchase completely or partially (one package at a time).
<table>
<thead>
<tr>
<th>Package</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central Unit</td>
<td>Kaalisy</td>
<td>The brain of the whole home automation system, manages all the information from different installed packages and is responsible for the actions taken by the packages.</td>
</tr>
<tr>
<td>Bulb Adapter</td>
<td>Lumus</td>
<td>A device of small dimensions that is connected between the bulb and the lamp holder, allowing to make smart every single light.</td>
</tr>
<tr>
<td>Switch</td>
<td>Lumus Switch Wall</td>
<td>An intelligent switch that, allows to make smart every switch points. It may be inserted in the place of the switch or if you prefer above the breakers themselves.</td>
</tr>
<tr>
<td>Wall Plug Adapter</td>
<td>Wally</td>
<td>A wall outlet where smart appliances, televisions and household appliances are connected. It allows you to:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Turn on and off connected devices, Receive notifications when the consumption exceeds a certain threshold</td>
</tr>
<tr>
<td>Thermostat</td>
<td>Ocio</td>
<td>A compact device that in addition to provide all the function of a thermostat is able to interact with the central unit allowing the user to create scenarios increasingly dynamically.</td>
</tr>
<tr>
<td>Door Opening</td>
<td>Locker</td>
<td>Allows the user to make smart key locks in the home (door entrance, gate of the house).</td>
</tr>
<tr>
<td></td>
<td>Sensy</td>
<td>Allows the user to make smart key locks in the home (door entrance, gate of the house).</td>
</tr>
<tr>
<td>Surveillance Camera</td>
<td>Intrusor</td>
<td>It offers the possibility of introducing a Web Cam, allowing the user to monitor in real time what happens in their own home.</td>
</tr>
</tbody>
</table>

Table 2.5: Functions of the packages
Kaalisy is also compatible with all Z-Wave devices, which permit the customer to use in addition to our packages, which use an owner communication protocol, also any other devices already in its possession and based on Z-Wave. When a package fails the system will note the absence of such package and will inform the user, whom may replace it, free of charge if it happens during the warranty period provided by law. In case of failure on the central unit, every package will go in "Single mode" and will work independently from the others. This will cause the loss of management scenarios and remote management; the rest will be like in a house not smart.

2.4.2 Price

The Low Cost strategy chosen by Kaalisy Domotics it affects the hardware and plastic components used for assemble the packages. The selected components are not the one with highest quality available, but this decision does not implies the malfunction of the system. This because the system does not need to guarantee a precision of 99.9\% during the communication between packages and central unit, even a precision of 90\% (which could then lose 1 package communication sent every 10) it is absolutely reasonable choice because it will be sufficient resend the information, with an acceptable delay of 0.2 ms.

![Figure 2.11: Costs of electronic components required for each package](image)

The Figure2.11 summarizes the costs in the retail market of electronic components needed to assemble the packages provided in Kaalisy Domotics. In a large-scale context is therefore reasonable to assume prices even lower. The Figure2.12 summarizes instead the final price at which we assume to sell packages, according to simulations (see financial plan)
made on the costs of production and on the fixed costs of the company allow to obtain an interesting gain margin.


Figure 2.12: Selling prices of the packages

The Figure 2.13 compares the competitive prices of some packages present in the offer of Kaalisy Domotics against the equivalent best solution (based on sales and reviews) on this site www.smarthome.com.


Figure 2.13: Selling prices of competitor’s products

2.4.3 Sales channels (Placement)

The sales channel chosen is the e-commerce. A web site will provide to the customers all the features and the informations about our company and our products. Moreover the web will be used as point of sale and point of care. This choice allows to reduce the final cost of the product to the buyers by eliminating the increases of price which would be caused by distributors and retailers and reducing the costs of management sales and service. It is a strategic choice made by considering the target market, or people, usually in possession of smartphone and/or tablet and with a certain confidence with computers and electronics, used to seek and buy on the Internet. In addition, this eliminates the territorial barriers and makes possible future expansion on the international market. Products purchased on-line will be sent to the buyer by delivery service carried out by couriers.
2.4.4 Promotion

An aggressive advertising campaign it is planned starting on internet channels (AdWords and social networks) to attract potential buyers on our website. Around the website, using the social media, we intend to create a community of customers and fans of Kaalisy Domotics so that they can communicate and exchange information and suggestions as well as provide new ideas, at the same time helping us to understand the real customer needs. As soon as possible we intend to launch advertising campaigns also on the more traditional channels, to reach those who are less informed in technology development and therefore cannot imagine which the offer became possible thanks to low cost smart solution. We will expect radio campaigns, at least on the local radio stations and press campaigns on free press and publications that can reach our customers, and then the local press. During further growth we would even get to the national press and maybe even to reach the TV channel, initially local and then national. We will not neglect the traditional channels; we intend to participate at the italian fairs about home and home furnishings that are open not only to the industry but especially to the final customers. These occasions can also be used for promotion. We expect partnerships with local companies that operating in the field of plant and electrical installers/designers who could then offer its customers a complete solutions very interesting.

2.5 Operational Plan

2.5.1 Roadmap

What has been developed so far is still at the prototype stage. A standardization and industrialization phase of the production of each component is required. Before to insert the packages in the market it will be also compulsory to obtain the CE certification. The main steps expected, with their timing, are illustrated in Figure2.14 Gantt Overall. The phase of product development is estimated at 6 months. After that the operation phase will begin.

![Figure 2.14: Roadmap of the product’s development](image-url)
Throughout the start-up phase (first 3 years), we identified the following milestones:

**First 6 months:**
- Product development and CE certification

**End of second year:**
- Extension the compatibility to other communication standards
- Development of new packages

**End of the third year:**
- Development of new packages
- Ready to start marketing in Europe . . .

### 2.5.2 Compliance of the system

The buyer should have the security of buying a product which conforms to the current regulations in the field of electronics. In this regard, it will be necessary to comply with Directive 2006/95/EC on Low-voltage equipment, and then get a CE sign. At this stage, we consider to consult a Notified Body, which will deal with the preparation of the EC Declaration of Conformity, and will assist us in filling the Technical Documentation. The contribution of this organization will allow us to speed up this phase and in about two months we will be able to print the CE mark on our products. The costs are around 1,500 to 2,500€ for the consulting and the certification.

### 2.5.3 Make or buy?

When the design of our products will be ready we intend to rely on an experienced company, which uses automated systems and is able to assemble at cost and time absolutely unreachable for us. To simplify the simulations, we assumed economic and financial cost of the assembly as an overhead of 10% compared to the cost of materials.

### 2.5.4 Operation characteristics

Once the production of packages will be standardized, the operations will take place as represented in Figure 2.15
Figure 2.15: Typical operations of Kaalisy Domotics
**Purchase Components**

The electronic components will be purchased from Supplier1 and will be sent directly to the Assembly Company. The cover of each individual packages will be purchased from the Supplier2 and directly shipped to our company.

**Components Assembly**

The components are assembled by the Assembly Company and sent to our Company.

**Final product**

Require the software installation and the application of the cover for every single package assembled.

**Storage**

The final products are stored in a warehouse and ready to be sold. Then it will be important to take care of the management of the inventory, orders and stocks.

**Customer Orders**

The orders will be reported by the website to the shipping manager, who picks up the products from the store, prepare and carry out the shipment.

**Shipping**

It will be provided through couriers to ensure a safe and prompt delivery.

### 2.5.5 Technical Assistance

We are in a start-up phase and the opinion of buyers is essential for the promotion and development of our product. Beyond the possible interactions via social media, we will offer four dedicated instruments to the customer: three acts to assist him during the pre and post sales phases, and one specific for technical assistance after-sales.

**Interuption on the manufacture of the components**

We need to purchase different components with very specific properties, so it is reasonable to take into account a possible interruption in the production line of such components by the producer. This will interfere with our production line since a necessary component to our product is no longer available. Overcome this problem is not always easy and for this reason we will first take it into account in the negotiations phase with the supplier/manufacturer. If this should be not enough, we will be forced to redesign the Hardware
and Software part involved by that component. In the projecting phase we will try to mitigate this risk by trying to develop each part quite independently from the components, especially for the Software part. This will allow us to replace more easily the involved component, even if it was a voluntary choice, derived from the arrival of components considered better of that in the market.

**Computer security**

Security is a delicate aspect especially in the home automation field. With the purpose of offer at the customers a safe product, Kaalisy Domotic opted for a local solution, providing at the buyer the full management system in a single device, connected only to your home router. Such solution allows to avoid the use of an external server (where to install the entire management system), which are more susceptible to attacks by hackers, thereby reducing the risk of intrusion. The proprietary communication protocol as well as permit the different devices to communicate each other adds even an additional layer of security by encrypting communications in order to prevent any intrusions during the communication between the devices and the central system.

**Intellectual property**

Like most of our competitors, we intend to protect the rights of our intellectual property through patents where possible. Not all packets will be patented, for example in the case of the central unit that uses an open-hardware platform will be patented only the software part. The initial cost of patents is around 10,000€, in later years the maintenance cost
will be increased by approximately 20% for annuity starting from 600€ for the first year of maintenance.

2.6 Organizational Structure

2.6.1 Value chain

The value chain of Kaalisy Domotics implements all the primary activities identified by Porter Figure2.17 logistics inputs, operations, logistics outputs, sales, marketing and services. Essential to our company are also the activities (support for Porter) R&D and Procurement. In the organization chart of the company (Figure2.18) are identified the functions responsible for each activity.

Figure 2.17: The primary and support activities of Porter’s Value Chain for Kaalisy Domotics

2.6.2 Organization chart of the company

CDA
Board of Directors. Formed by the 5 founders, who will play the roles of Executive, will be the place where strategic decisions are taken.

CEO
Chief Executive Officer. Responsible for the implementation of strategic decisions within the company and the administrative / legal, personnel management.

CFO
Chief Financial Officer. Head of financial assets.
2.7 The economic and financial plan

To estimate the amount of sales of the devices we started with the data provided by Assodomotica and we have estimated the number of new plants planned in the next 3 years, assuming a reasonable penetration of the market (Figure 2.19) and taking into account that in the first year sales will begin only in the second half of the year.
Figure 2.19: Hypothesis of market penetration

The economic and financial simulations were made based on the configuration type envisaged in Figure 2.1.

Figure 2.20: Type of configuration expected

Figure 2.21: Plan of sales by product

All the other financial analysis are showed on the Appendix 1.
Chapter 3

Software Design

3.1 Purpose of the System Design Document

The Design Document has to be a mean to arrive at the implementation part with the know-how which allow to respecting the requirements. This part has no value if the reader has not readed the Software Requirement Document that is in the Davide Angelici MSc Thesis. In this document we will explain the design of all the System, starting from a general scheme that represent all the component until reach the design of the software. We know from the requirement document which are the property that the system must have, here we will learn how the system intend to meet that specific. It depict a guideline to the implementation. Even if in that phase some change can be done due to some thoughtless problem that occur, the most of notions write here will be respected.

3.2 General Overview

To describe the realization of the system we begin from a logical overview in which is possible begin to recognize all the component present in the Kaalisy Domotic offer, but more important is understand how they are connected together and which role they have within such a system. However, once that we are outlined the architecture we are going to introduce the hardware components that forming it. For each component employed to build the system there is a brief description and a table with its technical specification, in this way if the reader is a bit skilled can understand by himself that the components satisfy the requirements for play that role. Next to the Architecture design is present the Software Architecture in which is explained the System from a software point of view, in
particular the technology that we are going to use to implement the specific parts of such system. Once that the reader has a good overview of the components from both Software and Hardware part it is time to show how the communication works between the Central Unit and the other Package. Then, everything should be clear so we can start to design the Database and delineate the inputs and output of the system.

### 3.3 Design Considerations

In this section there will be listed all the issues which need to be resolved in order to achieve a complete design solution.

#### 3.3.1 Goals and Guidelines

The goal of this project is to create a home automation system which is plug&play and it can be purchased at an affordable cost. Moreover it has to be easy to use and give to the end user an easy way to control it, to provide all of these functionalities the system must be designed respecting some roles:

- The communication protocol should privilege the affidability respect to the velocity.
- An affidavitable structure event condition action (ECA) has to be created to manage the rules (Scenari).
- The software should be as independent from the hardware as possible
- Write a language that allow the user to insert the rules in an easy way
- The central unit has to recognize a new packet in the system and install it
- The User Interface has to be easy-to-use

These are the goals that our system must comply to be in line with the requirements, every goals has its critical part and require a lot of work in the design part to be avoided. The first and second are the most complicated and they will require more effort than every other to be accomplished. To be sure that each packets sended reach the destination we need to implement a really reliable communication protocol. Create an ECA structure and let the user insert the rules means the possibility to have some inconsistent rules that may create a loop or other problem, we need to find a solution to this problem in the design part.
3.3.2 Development Methods

The methods that we are going to use to develop the design of the system and the software are UML and sometimes Object-Oriented to be more precise. A house automation system is composed from several part and the design can become complicated or confused, due to this we have choose UML rather than a textual modelling language. UML allows also to work in more people, the diagram are easy to understand and to modify, even after the develop of the whole system.

3.3.3 Architectural Strategies

We had decided to use Open-Hardware platforms as bases, and develop our system on that. This allows us to spent less time on the design hardware phase. Regarding the software part instead we will employ just a few libraries aimed to the use of the principals features of the Open-Hardware platforms and the most part of the software will be develop by us. Although develop a communication protocol take a considerable amount of work hours it bring a lot of advantages, such as the security or the possibility to balance the velocity and the reliability according with our necessity. The most of the code will be write to program the central unit, the core of the system: the ECA structure, the web server and the database will reside there.

3.4 Hardware Architecture Design

This section outlines the system and hardware architecture design of the system that is being built.

3.4.1 Logical View

In the graph below it is represent an overview of the system, without enter in the specific, it is shown how the main components are connect each other. At this point there is not complication, everything seems simple and understandable.

Components:

Central Unit: Manage the packages and connect everything to Internet through the router.
Package: Provide connection to the actuators and sensors it is the access point to the house environment, it take order from the Central Unit.
Figure 3.1: Logical view of Kaalisy system

**Sensor**: Read information from the environment.

**Actuator**: Take an action on the environment.

**Browser / Mobile Application**: Provide interfaces to the user to command the Central Unit.

The packages may also communicate each other, but in this view it is not shown to keep the graph as clearly as possible.

### 3.4.2 Hardware Architecture

The Figure 3.2 show the precedent view but instead of the components name there the name of the hardware platform that will be used to realize the system. It is possible note that the package will see their implementation on two platform, the use of the ATtiny85 generally is favorite because of the cost, but due to the lack of its GPIO pins some time can be replaced from the Arduino. Of course in the follow description of the platforms we are not going to write about the Smartphone and the Computer.
Figure 3.2: Architecture view of Kaalisy system

**Raspberry Pi**

The Raspberry Pi is a Single-Board Computer, initially designed for schools, but then used to carry out projects of all kinds by programmers from around the world. Introduced in 2011 and entered in the market in February of 2012, Raspberry Pi became popular in every state, and its sales increase day by day.

Figure 3.3: Raspberry-Pi
Raspberry Pi Foundation, the company that developed the device, is a no-profit organization. In conclusion one can say that the choice of Raspberry Pi is not been done only for its excellent hardware, but even for the community that is developed around this device.

<table>
<thead>
<tr>
<th>Technical Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prezzo di offerta:</strong></td>
</tr>
<tr>
<td><strong>SoC:</strong></td>
</tr>
<tr>
<td><strong>CPU:</strong></td>
</tr>
<tr>
<td><strong>GPU:</strong></td>
</tr>
<tr>
<td><strong>Memory (SDRAM):</strong></td>
</tr>
<tr>
<td><strong>USB 2.0 ports:</strong></td>
</tr>
<tr>
<td><strong>Output video:</strong></td>
</tr>
<tr>
<td><strong>Output audio:</strong></td>
</tr>
<tr>
<td><strong>Memoria:</strong></td>
</tr>
<tr>
<td><strong>Collegamenti di rete:</strong></td>
</tr>
<tr>
<td><strong>Periferiche di basso livello:</strong></td>
</tr>
<tr>
<td><strong>Real-time clock:</strong></td>
</tr>
<tr>
<td><strong>Potenza assorbita:</strong></td>
</tr>
<tr>
<td><strong>Alimentazione:</strong></td>
</tr>
<tr>
<td><strong>Dimensioni:</strong></td>
</tr>
<tr>
<td><strong>S.O. supportati:</strong></td>
</tr>
<tr>
<td><strong>S.O. non supportati:</strong></td>
</tr>
</tbody>
</table>

Table 3.1: Raspberry-Pi Technical Specifications

Arduino

Arduino unlike the Raspberry Pi is called Single-Board Microcontroller; the difference is in the integrated circuits: Arduino has a micro-controller, while Raspberry Pi a microprocessor. Arduino is designed to physically interface to a system, through digital and analog pins, for this reason is used in many projects of automation. Designed in Italy in 2005 and sold to the public since 2008, Arduino now is part of several projects carried out for commercial and educational purposes. the Community which is developed behind Arduino is huge, this is probably because is an open-source hardware platform, so anyone can build an Arduino or making any changes.

Figure 3.4: Arduino Pro Nano
Technical Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>ATmega168</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller</td>
<td>ATmega168</td>
</tr>
<tr>
<td>Operating Voltage</td>
<td>3.3V or 5V (depending on model)</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>3.35–12 V (3.3V model) or 5–12 V (5V model)</td>
</tr>
<tr>
<td>Digital I/O Pins</td>
<td>14 (of which 6 provide PWM output)</td>
</tr>
<tr>
<td>Analog Input Pins</td>
<td>8</td>
</tr>
<tr>
<td>DC Current per I/O Pin</td>
<td>40 mA</td>
</tr>
<tr>
<td>Flash Memory</td>
<td>16 KB (of which 2 KB used by bootloader)</td>
</tr>
<tr>
<td>SRAM</td>
<td>1 KB</td>
</tr>
<tr>
<td>EEPROM</td>
<td>512 bytes</td>
</tr>
<tr>
<td>Clock Speed</td>
<td>16 MHz (5V model)</td>
</tr>
</tbody>
</table>

Table 3.2: Arduino Pro Nano Technical Specifications

**ATtiny85**

The ATtiny85 is a great substitute to the Arduino. It can run the most of the Arduino programs, but is cheaper and smaller. As can be seen from the Technical Specification the difference compared with the microcontroller of the Arduino (ATmega328P) is in the amount of flash memory, RAM and specially the numbers of GPIO pins. Though at the role that it has to play inside the Kaalisy Domotic System, it is pretty obvious that this is favorite rather than the Arduino, the price and the size are not comparable. The only disadvantage resides in the greater complexity to program it.

![ATtiny85 microcontroller](image)

Figure 3.5: ATtiny85 microcontroller

Technical Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>ATtiny85</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash (Kbytes)</td>
<td>8 Kbytes</td>
</tr>
<tr>
<td>EEPROM</td>
<td>512 Byte</td>
</tr>
<tr>
<td>SRAM</td>
<td>512 Byte</td>
</tr>
<tr>
<td>Pin Count</td>
<td>8</td>
</tr>
<tr>
<td>Max. Operating Freq. (MHz)</td>
<td>20 MHz</td>
</tr>
<tr>
<td>CPU</td>
<td>8-bit AVR</td>
</tr>
<tr>
<td>Number of Touch Channels</td>
<td>3</td>
</tr>
<tr>
<td>Hardware QTouch Acquisition</td>
<td>No</td>
</tr>
<tr>
<td>Max I/O Pins</td>
<td>6</td>
</tr>
<tr>
<td>Ext Interrupts</td>
<td>6</td>
</tr>
<tr>
<td>USB Speed</td>
<td>No</td>
</tr>
<tr>
<td>USB Interface</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 3.3: ATtiny85 Technical Specifications
Actuator

An actuator may be considered an "active actor" that change in some way the environment in which it is installed: the bulb used to light up a room it is a practical example. In the system there are just two actor that can be considered active: the relay and the transceiver (transmitter part). To command this actuator we do not need any kind of driver circuit, all that is required is embedded in the Arduino or ATtiny85.

Sensor

A sensor, in contrast, can be considered "passive actor", which reads the state of the environment and communicate it to the system without modifying any part. Thermometers, luminosity and humidity sensors, switches, smoke or gas detectors, and many other components are considered sensors. Even if it does not perform actions within the environment are essential to check the status: usually an actuator is activated after the signal (read state) sent by a sensor. The installation of the sensors usually does not depend from the type, their function is to read a number and send it to the Arduino or ATtiny85, which will have to simply translate it into an entire that has meaning for the system and forward it to the Management System.

Other Components

In this document we are not facing some issue as: the supply needed from this platform, the design of the circuits and the PCB. These are not a small issue in which we will never face, on the contrary they may become insidious problem. But this is not the topic of such documentation, it is enough to let know at the reader that we found a perfect power supply (LS03-15B05SR2) that consume less than 1W and is as cluttered as an Arduino Pro Mini. To the circuit until now we rely on the scheme found in Internet that are really well made.

3.5 Software Architecture Design

In this paragraph we are going to introduce the Software Technology that we intent to use to build the system. In the Figure3.6 it is show how they will be employed, in which platform they will be installed. A deeply explanation of the technology will be done in the implementation chapter, here they are just listed and their roles are just outlined.
Figure 3.6: Software view of Kaalisy system
3.5.1 Central Unit

The central Unit is the brain of the entire system, it will contain:
The **Server**: used to allow the communication between the system and the outside devices (smartphones, workstation). It was decided to use Apache http 2.4.10 with PHP5.
The **Databases**: where all the informations are stored, which is accessible from the management software. The databases used is MySQL, because it is open source and is the most used database system.
The **Management Software**: Responsible to change the state of the system according to the input of the user. It is written in Java, because it is a portable language and it does not need to be recompiled in the case of changing type of platform.
The **Configuration Web Page**: It allows the user to interact with the system, the web page is written in HTML5, CSS3 to set up its mockup and structure. JavaScript and jQuery will be used to perform all the access and request from the client to the server and vice versa.

3.5.2 Packages

The packages are the end devices installed in the house. Their function is to perform actions send it by the Central Unit. They are programmed using open source libraries which will be installed into open hardware components.

3.5.3 Mobile App

The Mobile application is used to interact with the system by smartphone. It is developed in Android or iOS depending on which operating system is installed. Its main function is to perform the classic GET/POST request from the Server. Moreover the application, thanks to the Google speech recognition libraries, it is able to perform user inputs by voice commands.

3.5.4 Workstation

It simple consists to redirect the user to the Configuration Web Page, where it can change settings and creates scenario, all the modifications are saved into the system and accessible through mobile app.
3.6 **Internal Communications Architecture Design**

In order to satisfy the requirements of the system the topology of the network as a Mesh, it is necessary. This kind of network it is difficult to implement and configure, but it gives important and essential advantages: fault tolerance, easy to troubleshoot and the range of the signal is as large as possible. Each package communicate with all the others (within the range of the signal), this create an overload on the packets sended and more in general in the traffic over the network, but this overload is insignificant compared with the advantages that takes this type of topology.

![Mesh Network view](image)

**Figure 3.7: Mesh Network view**

Furthermore, we want that the system is capable to understand when a new package enters for the first time in the network and auto-install it, in order to allow the auto-recognition of the packages we have to properly implement the communication protocol.

3.6.1 **Communications Hardware Architecture**

To be in line with the goals of the project, we need a cheap transceiver, with a signal that has a good range and a restrained power consumption, careless the bandwidth and speed of the signal. Thus we opted for two solutions, one already standardized with his communication protocol and another cheaper but devoid of any protocol, both use a low-power RF radio. The first is the well-known Z-wave will be installed only on the central unit to provide compatibility at all the Z-wave packages already in the market and for the our packages a nRF24L01+ transceiver will be used.
Technical Specifications

| Power consumption: | 900mA deep sleep mode 

11.3mA Radio TX at 0dBm 

13.3mA Radio RX at 2Mbps on-air data-rate |
|-------------------|-------------------------------------------------|
| Radio specifications: | 2.4GHz ISM band operation 

GFSK modulation, 1 or 2MHz bandwidth 

0, -6, -12, and -18dBm programmable TX output power 

Configurable on-air data rate of 250kbps, 1Mbps or 2Mbps 

-94dBm RX sensitivity at 250kbps 

-82dBm RX sensitivity at 2Mbps 

-85dBm RX sensitivity at 1Mbps 

Excellent co-existence performance 

Compatible with a 16MHz 660ppm crystal |
| Interfaces: | SPI, up to 10Mbps |
| Power supply: | Internal linear voltage regulator 

1.9 to 3.6V supply range |
| Temperature range: | -40 to +80 ºC |

Table 3.4: nRF24L01+ Technical Specifications

**nRF24L01+**

**Z-Wave**

There are different transceivers with the capability of implementing the Z-wave standard, the one that we are going to use in this project is the ZM3102.

Figure 3.9: ZM3102 Transceiver
### Technical Specifications

**Power consumption:**
- TX@-5dBm = 24mA
- TX@0dBm = 36mA
- RX = 23mA
- Power down/sleep mode = 2.5 µA

**Radio specifications:**
- Freq for 9.6kbps / 40kbps EU: 868.42 / 868.40MHz
- High Sensitivity (-102/-98 dBm)
- FSK Modulation
- 9.6kbps/40kbps Data Rate
- -22.0dBm to -2.0dBm Output power (ZM3102N) (EU, US)
- -23.5dBm to -3.5dBm Output power (ZM3102N) (HK, ANZ)

**Interfaces:**
- 10 General Purpose I/Os
- Two Interrupt Inputs
- Serial UART SPI Interface
- Triac Control Interface
- PWM Output
- Four multiplexed 12/8 bit ADC inputs

**Power supply:**
- Internal linear voltage regulator
- 2.1 to 3.6V supply range

**Temperature range:**
- -40 to +85°C

Table 3.5: ZM3102 Technical Specifications

### 3.6.2 Communications Software Architecture

The main goal of the nRF24L01+ transceiver is to allow all the devices to share and exchange information. All the transceivers are programmed taking into account that they have to be efficient and reliable, than it was decided to let them spend almost all their life cycle in sleep mode, in order to decrease the energy consumption, and wake it up only when they have to perform/process some actions. To accomplish the exchanges of informations between the package and the central unit, an own communication protocols is realized. It is responsible to:

- synchronize the data flow
- detect and prevent the loss of informations exchanged
- ensure the security of the system

The system is also able to communicate with Z-Wave devices, thanks to the Z-Wave protocol. This protocol is working according to the specification of the Company; therefore our work it consist to modify few parameter which allow the Z-Wave module to communicate with the devices supported.
3.7 DB Design

In our system there is the necessity to have two separate databases, one dedicated for the web page used to purchase the product and the other one, installed in the central unit, used to control the home automation system.

3.7.1 Central Unit Database

In order to have a clear and efficient way to get the information inside the database, it was decided to divide it as following:

![E-R Central Unit Database structure]

Figure 3.10: E-R Central Unit Database structure

The structural organization of the database respect the way how the system is organized in the real environment. This decision will be the first step to provide at the end user and easy approach to the system. To maintain this easy organization, all the entity of the databases must be able to provide the informations necessary to guarantee the efficiency of the system.

The entities and the informations that reside in the database give to the system the following characteristics:

- Strong adaptivity in case of changing requirements
• Simplicity to maintain and update
• Reduction in population of the database

The way of the datas are organized (Figure 3.10), increase the system’s ability to process the inputs of the user and perform the desirable actions, ensuring the usability and efficiency of the entire system.

Central Unit Database Identification

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Attribute name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id_Room</td>
<td>Identification Room</td>
<td>Univoc number that identifies which room is it</td>
</tr>
<tr>
<td>Id_Package</td>
<td>Identification Package</td>
<td>Univoc number that identifies which package is it</td>
</tr>
<tr>
<td>Id_Central_Unit</td>
<td>Identification Central Unit</td>
<td>Univoc number that identifies the central unit</td>
</tr>
<tr>
<td>Status</td>
<td>-</td>
<td>Inform the user about some malfunction</td>
</tr>
<tr>
<td>Name</td>
<td>-</td>
<td>Show the type of room (Livingroom, bedroom, ...)</td>
</tr>
<tr>
<td>N_Accesses</td>
<td>Number Accesses</td>
<td>Count the number of access, from the user, in that room</td>
</tr>
<tr>
<td>Type</td>
<td>-</td>
<td>Show which type of package is it (lighting package, sensor package, ...)</td>
</tr>
<tr>
<td>Value</td>
<td>-</td>
<td>Show the value of the end device (On, Off, 23°C, ...)</td>
</tr>
<tr>
<td>Action</td>
<td>-</td>
<td>Show actions relative to the rule selected</td>
</tr>
<tr>
<td>Condition</td>
<td>-</td>
<td>Show the condition relative to the rule selected</td>
</tr>
<tr>
<td>Img</td>
<td>Image</td>
<td>Icon relative to the package or the scenario configured</td>
</tr>
</tbody>
</table>

Table 3.6: Central Unit Database Identification
48 ECA based Control System for Home Automation

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Attribute name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id_customer</td>
<td>Identification Customer</td>
<td>Unique number to identify the customer</td>
</tr>
<tr>
<td>Ctr_name</td>
<td>Customer name</td>
<td></td>
</tr>
<tr>
<td>Ctr_surname</td>
<td>Customer surname</td>
<td></td>
</tr>
<tr>
<td>Ord_date</td>
<td>Order Date</td>
<td>Date relative to the order made by the customer</td>
</tr>
<tr>
<td>Ship_place</td>
<td>Place where send the order</td>
<td></td>
</tr>
<tr>
<td>Price</td>
<td>Total price of the order</td>
<td></td>
</tr>
<tr>
<td>Prod_name</td>
<td>Product name</td>
<td>Description of the product</td>
</tr>
<tr>
<td>Prod_status</td>
<td>Product status</td>
<td>Status of the product: available or not available</td>
</tr>
<tr>
<td>Prod_price</td>
<td>Product price</td>
<td>Price of the single product</td>
</tr>
</tbody>
</table>

Table 3.7: Web Page Database Identification

3.7.2 Web Page Database

The main goal of the web page is to show and sell our products. For the purchasing process the system has to guarantee, to the end user, security and affidability. To maintain the web page understandable, secure, affidable and adaptive a simple and clear data organization is needed. To achieve this result it was decide to reduce at the minimum the number of the entities used, in order to decrease the complexity of the database.

![Figure 3.12: E-R Web page Database](image)

The Figure3.12 shows how the data are organized into the database of the web page. Is it important to specify that this is only the structure of the data regarding the purchasing phase.

Central Unit Database Identification

3.8 User Machine-Readable Interface

Only two types of users will interact with the system, the Admin and the Normal User. The first one has the permission to install the packages, the rooms, the scenario and change
the status of the packages. At the first use of the System it is duty of the Admin to configure all the rooms of the home and sequentially all the packages that he want to install. On the other hand the Normal User it is just allowed to interact with the system through the Mobile Application or Browser.

3.8.1 Inputs

The system receive two different kind of inputs: one from the Users and the other from the Environment. The first one are all the inputs performed by the user, using a Web Browser or Mobile Application. Some of them are commands given by pressing the Virtual Button on the interface. The second one are different because the inputs are automatically sended by the packages to the system, like the temperature sensors or the bright sensors. The system can handle several kind of inputs, each of them leads to the execution of different actions.

User Inputs:

Rule \Scenario
Input made exclusively by the admin, using the control page he can access to the scenario section and then, through a list item, containing all the commands and all the packages installed into the domotic system, he is able to create dedicated scenario.

Switch
This kind of input is present both in the control page and the mobile application. It consist of a button that can be pressed by the user and change the status of the package associated to it. This input it can be sended both from the user and from the admin.

Virtual button
That kind of button it appear to the user (control page and mobile app) when a group of packages were selected and associated to only one button. In that way the user can control several packages through a single interaction. It has the same characteristic of the Switch with the difference that the command is sended to several packages.

Text Field for Integer
It consist of a text box displayed on the control page and on the mobile app, where the user or the admin can insert integers to modify the status of with specific packages (thermostat package, Ê)
Vocal
The user and the admin can perform most of the input explained above (switch, text field for integer) by simply using their voice. This feature is present only on the mobile application and it consist in a widget, when the user select this widget he has to give a specific command that it will be parsed and sended to the central unit.

Environment:

Sensor
It consist of a automatic input sended by the sensor package to the system, the interval from two automatic input of one package is configured by the central unit and it can not be modified by nobody.

Camera
Another environment input is the one coming from the camera, which continuously sends informations to the system. The user or the admin can access, through the control page or the mobile application, to the surveillance area and check if everything is under control.

3.8.2 Outputs

The outputs that the system performs according to the inputs received are of two types

Physical Outputs When an input is performed by the user the system will execute it and will modify the status of the selected package. The relative output can be:

- Turn off the package 1
- Set the value 28 to the package 2
- Activate the scenario 3

Visual Outputs
Most of the outputs modifies the status of several packages, to maintain the system updated with all these changes it is necessary to update also the interface of the control page and the mobile application. Then the system when it performs a physical outputs it performs also a visual outputs, in order to maintain the interfaces synchronized with the status of the packages.
Chapter 4

Central Unit Implementation

In the Design Document are described several part to implement, here we will focus just on the topic of this Thesis that is the ECA Rules Engine, and the others part will be face more lightly or not addressed at all. Starting from analyze the architecture, by way of the libraries and IDE used up to explain the source code of the application.

4.1 Architecture overview

As can be seen in Figure 4.1 the Central Unit is composed by two Hardware platform: The Raspberry Pi aimed to manage the logic part and the Internet communication and the Transceiver used to communicate with the other package.

Of course the first things did was concern to set the environment, then we started looking for an OS on which would be possible install all the suite essential for us, that is the yellow box represented on the Figure 4.1. Afterwards, still in the Raspberry Pi, we choose to install Apache as Server, MySQL as DBMS and PHP as Scripting Language, altogether
these software form the so-called bundle LAMP, we are not going too deeply in the explanation of this bundle since this part it is covered by the Thesis "Web user interface and mobile application for an home automation system" of Davide Angelici. Regarding the control and management part we opt to write the Control Software in Java to facilitate the portability, for this reason a JVM and the Java libraries was required. To reach this purpose Oracle offer a Java SE aim to be installed in the ARM platforms that fit perfectly with our necessary. The Control Software it is the core of the system in fact it took a lot of working hour and the most of the thesis discuss about its structure and its implementation. Last but not least we written a program used to let communicate the Raspberry Pi with the Transceiver which have the task to send the message coming from the Control Software to all the other packages installed in the system, to reach this goal we used a C++ library that allow as to communicate through the Serial Peripheral Interface (SPI).

4.2 Install the environment on the Raspberry Pi

This subchapter it is write as a tutorial, describe the software and explain how to install it might seems boring and unnecessary, but from my point of view it should be read, to fully understand the process that bring to the implementation of the final product.

4.2.1 The OS

Since the launch of the Raspberry Pi many Operating System (previously born to run on a ARM Architecture) are been adapted to take advantage from this platform, most of their are Open-Source project that would never reach that goal without the community thats works behind on it. At the present state there is an OS that has overtake the others by considering the use of resources and the compatibility, it is the Raspbian OS.

![Raspbian logo](image)

Figure 4.2: Raspbian OS logo

Born from an unofficial porting of the "GNU/Linux Debian wheezy armhf", it has as a goal "to become the leading OS choice for all user of the Raspberry Pi". Unfortunately, just the Architecture independent ("Arch all") packages are compatible between Debian and Raspbian (due to the hard float[nota]) but this is not such a bad thing due to large
offer of software available to the Raspbian, it count now more than 35000 packages (pre-compiled software bundled in a nice format for easy installation on your Raspberry Pi). For us it is easy to say that fulfill our requirement, since it provide an easy installation of the Java/C++ environment and Lamp bundle, moreover optimizing the usage of resources that are really limited on ARM Architecture.

The installation of this OS is extremely easy, the image of Raspbian is available on the Raspberry Pi website, once you got the image it is sufficient write it on a SD card and boot the Raspberry with the card inserted. The first screen that will appear it is the **raspi-config** tool on which we need just to enable the SPI module to communicate with the Transceiver successively.

![Raspberry Pi Software Configuration Tool (raspi-config)](image)

Figure 4.3: View of Raspbian configuration tool

### 4.2.2 LAMP

The explanation of this bundle (Linux Apache MySQL PHP) can be found on the Davide Angelici Thesis, in this section we will focus only on the installation of the software on the Raspberry Pi.

Then, a kernel Linux it is already installed, is on the Raspbian OS. To assure to have a system upgraded and does not risk to create inconsistency prompt:

```
apt-get update
apt-get upgrade
```

Now, thanks to the precompiled software, install Apache, PHP and MySQL can be done prompting a command:

```
apt-get install apache2 php5 mysql-client mysql-server
```

After have inserted some password for the DB and Server the system it is ready to work.
### 4.2.3 Java JDK

Until two years ago run Java on the Raspberry Pi was not so easy, there were two possibilities: the ArchLinux OS that fully support the OpenJDK (many people was already working to port its compatibility also to the Raspbian OS) and the Java SE for embedded by Oracle that was developed for a Linux Kernel over a ARM Architecture. Nowadays both version are perfectionated to the Raspbian OS, so there is the possibility to choice between two excellent Environment: OpenJDK JRE and Oracle Java SE. Both version have the right requirements to be used in this project, so I took the decision to use the Oracle version looking at the benchmark available on Internet. The Oracle Java SE compared to the OpenJDK JRE seems to compute the calculation faster and with less waste of resources.

There is nothing to do to install the Java SE since in September 2013 the Raspbian images start to came with the Oracle Java by default.

![Java version](image)

**Figure 4.4: Java version**

### 4.2.4 SPI

In the pile of Figure 4.1 the mean of SPI might be misunderstood. Looking at the left part we have a Speaker above the SPI above the C++, of course this cannot be correct but we used this definition so the reader can easily understand the environment in which we are working. Before to explain the implementation of this part a focusing on the SPI would help the reader to richly understand what we are talking about.

The Serial Peripheral Interface is a communication system, to be precise in the OSI Stack we can individuate it in the Physical Layer and Data Link Layer. Designed and Developed by Motorola it is became almost a standard, largely used to allow the communication between the microcontroller and the other integrated circuits often in the same PCB. In the SPI Architecture there is only one Master and more Slave, in our implementation the Master is the Raspberry Pi and the Transceiver take the part of the Slave. The SPI bus is: Serial type, Synchronous and full-duplex.
Figure 4.5: Serial Peripheral Interface bus

It need 4 channel to send the signal: SerialCLocK, Master Output Slave Input, Master Input Slave Output and Slave Select (in the transceiver it is called CSN Chip Select Not).

Luckily, to manage the nRF24L01+ on Raspberry through the SPI bus there is a library developed by Stanley Seow that comes to our aid. It is written in C++ so we need to install a C++ environment (Builder, Linker, Compiler) on Raspberry, and furthermore to write a C++ program aim to deliver the message coming from the Control Software to the Transceiver. Now that is explained which is the right interpretation in the Figure4.1, it is time to move on to implementation of this part, starting from the installation of the C++ environment.

As for the other tools also this can be install just prompting:
```
apt-get install build-essential
```
that will provide to set all the environment that permit to compile in C++ and in particular the compiler g++. Once all this tools are installed, it is the turn of the Stanley library, it can be downloaded using GitHub through this command:
```
git clone https://github.com/stanleyseow/RF24.git
```
compiling and install:
```
make
make install
```
Inside this library there are also some example that are just what we need after wiring to test its functionality, and be sure everything it is done in the right way.

At this point the environment should be ready but it is necessary start wiring to make some test. To understand how to connect the Transceiver to the Raspberry Pi through the SPI, it is necessary a scheme showing the pinout of the GPIO on the Raspberry (it is important to pay attention at the model, the GPIO are different in each version) and the Hardware Interface on the nRF24L01+.

The Transceiver has 4 more pins that does not belong to the SPI, two of that are for the power (GND and VCC), CE is used to control data transmission and reception when in
Figure 4.6: GPIO pinout Raspberry-pi

Figure 4.7: Pinout nRF24L01+
TX and RX modes, IRQ is the interrupt pin, and is active-low. Also the CSN (Chip Select Not) pin is active-low, and is normally kept high. When this pin goes low, the 24L01 begins listening on its SPI port for data and processes it accordingly. According with this we can use just the CSN pin without the IRQ and connect it with the CE0 on the Raspberry (this depends from the Stanley’s library), now the other connection can be easily individuate. Finally to test everything, we used another Transceiver attached to an Arduino UNO, both was running the example called "Getting started_CallResponse" written by Stanley.

![Figure 4.8: Arduino< – > Raspbrry-pi communication test](image)

Yuppi Yo Yuppi Yeah the transceiver works well, with this magnific news we can consider finished the first part, the environment is perfectly set!

---

1 There are three internal interrupts that can cause this pin to go low when they are active. Each of these bits can be masked out such that when the bit’s respective interrupt becomes active, the status of the IRQ pin is not changed.
Chapter 5

Control Software

This chapter represents the main part of this thesis, develop the software to manage and control the system. It is done carefully and considering all the possible way to do it, of course at the end we implemented the one that fit more with our system. A similar version of this software was already developed by Claudio Tesei for his bachelor thesis, but that can be considered a raw version that never has seen a true implementation.

5.1 State of the Art

What we want is develop a software that can manage all the packages and in the same time permit to the user to program Scenarios. This can be done in many way, but the more efficient and easy (from the user’s point of view) is use a Rules Engine. To have a better understanding of the all chapter it is necessary spend some world about the Rules Engine. It is a tool that provides a different way to program something, we are accustomed to use a Declarative Programming but Rules Engine is different, in the first you say "How to do it" in the second "What to do". It offer a different view from a more common imperative model, providing a list of production rules instead of the commands in sequence. The rules are usually described by a condition and action. Thinking at the solution of Kaalisy Domotics, it is easy to observe how bad can be the idea of program the house using an imperative model. We claim the simplicity of our System and we want reach it even through the use of a Rules Engine.

There are two main categories of engines, forward and backward chaining. Forward chaining is the most used type nowadays, and it can be further divided in other two category based on what they process: Inference Rules or Event-Condition-Action Rules. The
first category indicate the production rules, they represent the "behaviour" (of an object for example), composed from a Condition and an Action. They can be easily interpreted as "IF condition THEN action". In the second category instead we can identified the reaction rules, when an event happen something is triggered and the System will react according with the rules, these are the ECA rules. "WHEN event IF condition THEN action" is the right interpretation. Return to what was said before there is a kind of engine that is called backward chaining, it use the information rack up during its life cycle to reach the goal prestabilish. We can say that it know how to extrapolate the rules from a goal takes as input. Usually this kind of engine use a Inference Rules.

Now it is clear that ECA rules are suitable with our System, and we need an Engine that process that kind of entries. From about 2002 to the present, plenty of engine are developed, most of they based on three Algorithms that are distinguished from the others for their efficiency: RETE, TREAT and the mix of this two Gator. The two most famous and used engines use the RETE algorithm, they are: "Jess, The Rule Engine for the Java Platform" closed-source/ public domain project developed by Sandia National Laboratories and "Drools, a Business Rules Management System" open-source project developed by Red Hat and licensed ASL 2. Furthermore, both of this engine have the support to the ECA rules. In addition to this, as said before, there exist others engines (usually developed to a specific purpose) open and close that can be found. Most of their are just written on a paper others have seen their implementation, but in any case they aim to the area of Semantic Web and Rule Markup Languages (RuleML, RIF,...) then they are not helpful for us. Exists also the possibility to implement the Rules Engine in the DataBase, for example it is possible build a Rules Engine with Microsoft SQL. We discarded this option because our Engine as the requires to do something more than just write an entry on the DataBase according with some rules (after in this reading this will became more clear).

It might be reasonable build our engine with the support of another engine as Drools, but this is not what we did. We consider carefully this possibility however at the end we opt to create our Rules Engine. Of course it is a forward chaining with ECA rules, yet the syntax is written by us and even the algorithm. Undertaken this way does not seems easy and it is not, probably even the performance of the system can be compromised, but we believe that this choice may does not carry immediate result now, however sooner or later it will demonstrate to be the right move. This is due to the fact that develop a customized engine to the system allows to integrate better the other components, as result we have that probably the Engine is not efficient as the one that adopt the RETE algorithm, yet considering all the system the result could be reversed. Under a programmer views, the
path taken is for sure hardest in the first part since we have to build the engine, anyway these efforts are paying off during the implementation of the others components.

5.2 **Kaalisy Rules**

As we mentioned before the ECA Rules Engine that we created is the base of the entire system, then it is very important understand well how it work and how it interpret the packages. We know that the rules are defined as Event Condition Action, but what we do not know it is how are defined the Events, the Conditions and the Actions? Starting from understand how it is organized the system it is explained what the ECA is. It should be clear that Kaalisy Domotics is composed by Packages compound from sensor and actuator, the sensors are capable to "read" from the system and the actuators can "write". Then it is logical to interpret they as variables, so each package has at least one variable. Thus, moving on to what we advert before, we can say that the rules has this form: "(Event) WHEN variables (Condition) IF bool_exp (Action) THEN variable = state", which means: "WHEN the state of at least one variable change then check IF the boolean expression (composed from variables) it is true, whether is true change the state of the variable".

This was just an overview of the ECA structure useful to understand that the rule are defined over the variables and the system can be govern as a set of variables.

5.3 **The ECA language**

In the Kaalisy Domotics System also a scenario could be seen as a rule, then it is easy to see how everything is governed by the rules, the most important are set by default, but other may be inserted from the user to meet more specific requirement. To satisfy this exigence and to manage the command sent from the Interface (App or Browser) we created a specific programming language used to define rules and variables, it can be considered the front-end of the Rules Engine.

5.3.1 **ANTLR**

Creating a programming language may be very challenging without the help of any tools, not impossible but it can take a lot of time, in fact we did not opt to write the ECA
Language by hand but we decided to generate the lexer and parser with ANTLR (ANother Tool for Language Recognition).

![ANTLR logo](image.png)

*Figure 5.1: Antlr logo*

Created by Terence Parr (and others) in 1989 and released the first time in 1992 under BSD License, now it is widely used to implement a grammar due to its powerful, flexibility and ease of understanding. This tool permit to generate a lexer and a parser starting from a defined grammar in ANTLR format (similar to the EBNF and YACC) or in a special abstract syntax to the AST (Abstract Syntax Tree parser). Then, ANTLR is something more than a simple definition language grammars, the provided tools allow to implement the defined grammar creating automatically lexer and parser in either Java or C++. ANTLR implement a pred-LL(k) parsing strategy and allow an arbitary length lookahead (called k) to disambiguate in cases of ambiguity. In other word it is the perfect tools to create our domotic programming language, but before to explain the grammar and its implementation it may be helpful explain what the lexer and parser are.

### 5.3.2 Lexer and Parser

The programming language are composed from key-words and well defined constructs which are sequentially interpreted to extrapolate the command that the machine have to perform. The two programs that perform this process are called Lexer and Parser.

A source code is sended as a flow of chars to the lexer from an input interface. The lexer works consist on pack this meaningless chars that will take meaning when will be elaborated from the parser. Either char or group of chars collected in this way are called tokens, they are components of the programming language as key-word, identifier, symbol and operator. Usually the lexer remove comments and spacing from the program, and all the other contents that has no semantic value to the program interpretation, then convert the flow of chars on a flow of token that have an individual mean dictated from the rules of the lexer. This flow of token generated from the lexer are then sent as input to the parser.
Despite the lexer collected a sequence of char that recognize has semantic value on the flow, it does not consider their value inside the context of the program because this role belong to the parser. The language are described through grammars that determine exactly what define a given token and which are the tokens considered valid. The parser organize the tokens in admitted sequences defined from the grammar of the language. If the language will be used exactly as defined in the grammar, the parser will be able to recognize the pattern that constitute specific structure and group them together properly. When the parser meet a meaningless sequence that does not match with any defined structure it will signal an error. The parser generates one or more symbol table that contain information about the token met. If the token is the name of a procedure or has some specific value, the symbol table are employed on the control of the type (for example to check that an integer will not be interpreted as a String). ANTLR use the symbol table to speed up the token matching.

An LL parse is a left-to-right, leftmost derivation parser that uses k tokens of lookahead when parse a sentence, it is in the category of top-down parser. It takes into account the input from the left to the right and try to build a leftmost derivation that is: from the first symbol and continually expanding the leftmost symbol up to the target string, the passed symbol are called nonterminal.

### 5.3.3 The Language in B.N.F.

Once that ANTLR and its environment is setted it is time to write down the ECA Language\(^1\). We spent a lot of time working in this part, before to define the grammar and after to give a semantic value to the grammar. The most difficult challenge is to create something that would be easy to understand without loss in expressiveness. In fact the

\(^1\) A first version of the follow language was create by Claudio Tesei during his Internship
ECA Language would be employed from the user (helped by the interface) to program the System (creating scenarios) and in the same time it has to be capable to manage the packets in all the possible way that means it has to be as expressive as possible. In addition the Language constitute also a point of contact with the Server, then the user interface, indeed the Server talk with the Control Software using this structure. Respect all this property it is not a walkover, and reach a point that can be considered good or acceptable could take years, because in this phase to obtain a good achievement it has to be paid attention at the testing part and especially its interpretation.

To leave no doubt in the reader, beneath there are some examples about what the language should be capable to say:

"When the temperature falls below 16 degrees and it is later than 11pm but earlier than 8am turn on the heating at 20 degree"

"When the power consumption has reached 3000 Watt turn off the wall plug2"

"Turn on light1"

As explained in the precedent paragraph the grammar will be composed from specific key-word, identifier, symbol, operators and structures. :

\[
\text{⟨prog⟩} ::= \text{‘rule'}\langle \text{rule} \rangle\langle \text{SEMI} \rangle \\
| \text{‘define'}\langle \text{define} \rangle\langle \text{SEMI} \rangle \\
| \text{‘delete'}\langle \text{delete} \rangle\langle \text{SEMI} \rangle \\
| \text{‘modify'}\langle \text{modify} \rangle\langle \text{SEMI} \rangle
\]

\[
\text{⟨delete⟩} ::= \langle \text{IDE} \rangle
\]

\[
\text{⟨modify⟩} ::= \langle \text{IDE} \rangle \text{‘value’} \langle \text{INTEGER} \rangle
\]

\[
\text{⟨define⟩} ::= \langle \text{IDE} \rangle \text{‘:=’} \langle \text{PACKAGE} \rangle \langle \text{REALNAME} \rangle \text{‘where’} \langle \text{bool_exp} \rangle
\]

\[
\text{⟨rule⟩} ::= \langle \text{IDE} \rangle \text{‘:= when’} \langle \text{bool_exp} \rangle \text{‘then’=} \langle \text{command} \rangle \text{‘else’} \langle \text{command} \rangle ?
\]

\[
\text{⟨command:⟩} ::= \text{‘{’} \langle \text{IDE} \rangle \text{‘=’} \langle \text{bool_exp} \rangle \langle \text{SEMI} \rangle \text{‘}’
\]

\[
\text{⟨bool_exp⟩} ::= \langle \text{int_exp} \rangle \langle \text{OPBOOL} \rangle \langle \text{int_exp} \rangle \\
| \langle \text{OPREL} \rangle \\
| \langle \text{NOT} \rangle \langle \text{int_exp} \rangle
\]

\[
\text{⟨int_exp⟩} ::= \langle \text{sum} \rangle \langle \text{PLUS} \rangle \langle \text{sum} \rangle \\
| \langle \text{MINUS} \rangle \langle \text{sum} \rangle
\]
Using this grammar we can write again the previous "phrases", that became:

"rule heating := when temp<16 && (time>23 or time<8) then heating = 20;"

"define PC := none virtual value wallp1 + wallp2 + wallp3 + wallp4;"\(^2\)

"rule monitor := when PC >= 3000 then wallp2 = off;"

"modify light1 = on;"

Actually the language has a good expressiveness but not so easy understandability, indeed an interface that simplifies the "talking" with the CSW it is required and will be always

\(^2\) The explanation of the virtual variable is in 6.8.1.1 Variable
required. In the actual state as is possible to note reading the MSc thesis of Davide Angelici the only "phrase" that the user has to write it is to enter is a rule, the others like "define", "modify", "delete" are generated automatically by the interface that let to the user just the filling of some text field or just press a button.

5.4 Overview of the components and their requirements

The CSW would perform different roles than just provide an ECA Rules Engine. The whole software is composed from 4 part, the core of course is the ECA part but the other 3 are still important, they provide communication with the DataBase, the Server and the Package.

![CSW components view](image)

Figure 5.3: CSW components view

In the Figure5.3 are described the 4 components of the CSW: Server Communicator, Package Communicator, DataBase Communicator and the Event Condition Action Rules Engine. It is easy to see how they works together, each communicator can send and receive information from the ECA that is the mind of this system, it must be underlined that sometimes (for example when the SC detects a new package installed within the System) can happen that some components talks together without passing to the ECA . The data flows that we have instead are represented on the following figures.

![CSW loading from DB](image)

Figure 5.4: CSW loading from DB

On the Figure5.4 it is shown the flow when the system is started, the ECA load the data from the DataBase, compute the calculation and send the output at the packages and the Server. The DB is readed from the ECA just in this case, in all the other case the DB is written immediately after that an event occur and change some data. Then we can observe that the ECA has also the task of maintain the data consistent between the DB, Server and Packages.
When an event occurs from a package, the ECA is notified and provides updates before the Database and after the Server. The Database is considered more important since a fault happens and the system should be restarted; the starting configuration is stored in the DB.

The last case is triggered by the user. Through the browser or application, they send a command to change something in the actual configuration. In this case, the ECA updates before the packages rather than the Database; this is preferred because we want a reactive system.

### 5.5 Server Communicator

To send the message between the Server and the CSW, we opt to use a socket. The CSW is written in Java and the communicator part on the Server is written in PHP. Both are in the same OS, so there are different ways to communicate: pipe, FIFO, through the DB. In addition, every OS has a loopback interface, so it is possible to use a socket. The reasons behind this choice are simple: sockets are easy to implement, are bidirectional, have a good performance, and the feature that is more attractive for us is the possibility to communicate inside a network. Thought a change of the architecture where the CSW is brought out in a different hardware from the Server, a socket would be perfect for communication. The Server Communicator is always waiting for a message coming from the server, reading constantly the InputStreamReader, barely a message comes it sends the data to the ECA engine and waits for an answer that will be forwarded to the Server. Instead from another point of view, the SC provides a method
used by the ECA engine to send data directly at the Server. For example when a new package is detected, the engine will send the information about that packet to the Server that will just notify the user and through a guided walkthrough will furnish to him an easy way to define the new package in the System. This method take as input just the String to forward to the Server. Of course the SC is not the main component of the CSW but despite its easy implementation its role is very important.

5.6 DataBase Communicator

As each respectable Domotic System also Kaalisy Domotic has its own Database where store the data about the user, the environment and the system. What concern the CSW in the Database are just two table Variables and Rule (to a better explanation of the DB see the thesis of Davide Angelici), the CSW must keep updated this table according with the wishes of the user and especially with the values of the actuator and sensor (which are precisely our variable in the DB). To communicate with MySQL, that is the DBMS installed on the Central Unit as we saw in the chapter 5, we make use of JDBC Java DataBase Connectivity. It actually is an API from Oracle Corporation that provide an interface for querying and updating data in a Database. Therefore, we use the DC class as an Interface to reading and writing from the Database not only but especially for the engine. It provide different methods to write, read, delete and modify a variable and write, read and delete a rule. The use of this class could have been avoid just coding a connection with the Database directly from the engine and the other part of the CSW that use the DB, instead we decided to create a class first for the clarity of the code and second, as before, because if a change of DBMS happened will be sufficient change the connector in this class to solve the problem.

5.7 Package Communicator

Send a message to the packages it is a bit more complicated than send it at the Database or the Server. In the chapter 5 was introduced the Speaker, a Software in C++ that allow the CSW to communicate through a transceiver using the SPI. As shown also the Speaker is locate within the Central Unit and even for it can be done the same considerations made out to the SC. In fact due to the same reason also in this class we use the socket to communicate with the Speaker. Aside this, there is a more complex part that is intrinsic to the message, because the Speaker will just forwards the String sent from the PC, it has also
the task to format the message properly to made it understandable from the package. Then the PC has to traduce the message according with the communication protocol used by the packages. Furthermore another task is been assigned to the PC, detect the new packages and signal it to the ECA Engine, this feature is deepened on the follow paragraph. The Package Communicator is always listening for a message from the packages, because if some sensors change its value the engine should promptly warned. Finally it provide some method to allow the ECA Engine to communicate with the packet, for example when a condition in a rule became true, it is duty of the engine make sure that the action will be performed.

5.7.1 Auto-installation

When a package is inserted for the first time, it will send a message content its package number (univocal for all the packages) and type to the Package Communicator which immediately will check whether this packed exist or no in the DB, in both case it have to respond with a "Welcome" message. Afterwards and notify the SC (skipping the ECA in

3 A strength protocol has to be implemented yet, until now we use a simple protocol that allow us to communicate with the packages.
this case) of this new package. The Server Communicator will notify the Server which will provide an interface to let the user set the package. Once that the user has finished to set up the package it is time to the system to define a new variable and let know at the user that all went well.

![Auto-installation sequence diagram](image)

Figure 5.8: Auto-installation sequence diagram

Above it is shown an example of auto-installation of a Lumus Package. The user settle in a new Lumus Package at home. The package is detected and the system follow the procedure as exposed, supposing that the user call the variable light1 and set it on, the generated String from the UI that the ECA Rules will receive would be: "define light1 := 0234 1 value 1;". At this point the engine do what it have to do and send the command to the PC, that will traduce it thanks to the help of the DB to obtain the package and variable number, sequentially the Package Communicator will send the command to the package and will wait until a confirm message arrive. If the "OK" message does not arrive means that something it went wrong so the PC will send again the command message until an answer arrive or the time expires (an time limit has set by default).

### 5.8 Kaalisy Rules Engine

Everything it is clear now, except a small part, how work the Rules Engine? Well, it is time to reveal the secrets of this mystic engine. This paragraph is focused on how the ECA engine play his role inside the Control Software. We understand how and from whom it takes the input, the mechanism concealed behind a command, how is administrate the message traffic, but how it transform the String: "modify light1 = 1" in a command to deliver?
5.8.1 Overview of the component

Even this time it is necessary understand the environment that rotate behind the engine, by doing so it will be possible pass at the next paragraph and understand the semantic value of each structure in the language.

As can be seen from the Figure 5.9 the main structures are two Hashmap that contain the link to the Variable and the Rule. The last one it is not a Java object but is a Thread that is running the Rule. A lot of structures could have been used instead of the HashMap, but we considered that for our purpose store a Variable or Rule based on the key that can even be used to retrieve the item later it is just what we need. In this phase the point is to use less resources when it is possible, running on the Raspberry Pi the CSW has to be really light if we want to have a good performance. For this reasons even the method employed to pull out an item from a data structure (an hashmap in our case) it is significant.

Variable

We know that a variable in the engine correspond to an actuator or a sensor in the domotic system, then we needed to insert values that can be used to represent any type of actuator or sensor. To address they there were no problem because each packages has an univocal identification String and each actuator or sensor has an univocal "realname" inside the package. Represent the status instead was not so easy, in a first version we had generalized and use boolean and integer value. Boolean to all the variable that can be turn on or off and integer to all the other. This way to storage the status works pretty well but became more complicated during the defining phase and means also another value to store and pass during the call aimed to identify the type. Due to this in the following version every
status it is represented with an integer that will be decodified in a boolean or an integer inside the package.

To distinguish between actuator and sensor, we used two String "read" and "write". Actually there is also a third type of variable, it has versus "virtual" means that does not physically exists in the system, in fact it is used to express value that does not came from a package. An example may explain better which is its role.

"define PC := none virtual value wallp1 + wallp2 + wallp3 + wallp4;" In this case the variable PC does not represent an actuator or a sensor but the value of four variables, so it can be defined as "virtual". In the Figure 5.10 it is shown also another member, an ArrayList of String called rules, that list contain the name of the rules in which the variable appears in the condition. Another possibility to achieve the same goal (link the rules with the variable present in their conditions) could have been insert a list of the name of the variable inside the class rule, at first sight seems even more elegant as solution, but further it will be clear the reasons of that choice.

**Rule**

The rule is the most complex concept in this document, at this point the reader should have understand the major feature of the rule, but we have just mentioned at the beginning of this subchapter about its implementation. As we know a rule must to be reactive,
as soon as the condition of the rule changes, the system has to perform the consequent actions.

In the Figure 5.9 it is shown that the rules are represented as a Thread, this is not completely correct, the rules contained on the HashMap are Objects of the class shown in Figure 5.11 that have a Thread as member. Of course that Object can be interpreted as a Thread, but was necessary to be clear underline this difference.\(^4\) Returning to what was said before, Thread are optimum to our purpose. We can start a thread for each rule, and make a cycle inside it to check the condition of the rule, whereas the condition is true or false it will decide which action perform. Such a method is easy to do, but it is pretty obvious that wastes a huge quantity of resources. Consider that the system has to run 20 rules, means that 20 threads have to cycling together, it is unthinkable that such a solution can works, specially on the Raspberry Pi. A patch for that solution can be put a wait of 10 seconds for example, but in this way the engine would not be reactive. Then remains just a way, make the thread wait and advise it just when the variables present on his condition changes. In this way all the requirement are satisfied, a waiting thread does not afflict the performance of the CPU it weighs only on the memory, but considering that the number of thread are not so elevated this is an acceptable compromise. The reasons whereby there is a list of rules on the variables members, it is for such purpose: when a variable changes status it has to notify all the Thread that are checking the condition in which it is present, the Object Rule inside the Rule HashMap it is just an handle to allow the variable to reach the Thread and notify it.

### 5.8.2 The semantic value

In the paragraph 5.3.3 was presented the grammar of the Language, and then the following paragraph has been utilized to describe the components and the structures of the Control

\(^4\) Since the call stop has been deprecated was impossible put directly the Thread on the Hashmap and all this was necessary.
Software and specially the ECA Rules Engine. Thereby the role of each component was identified and now should be clear how they work. But to be more specific and does not skip any part we are going to show how the parser play its role. To resume what it is written in the chapter 6.3.2, the parser has the task to transform the token received as input from the lexer in specific command, doing so they will take a semantic value. To know exactly the mechanisms that lets work the parser it is better see the source code, indeed below they are described with a pseudocode and a brief explanation, of course such code it is not the same that can be found in the Language.g file, but has the same semantic value. The program will always run the input string tokenized as start. To be shorter we used the key-word of the grammar to identify the value that they assume on the input.

```
start
    if (input == "rule") perform rule;
    if (input == "define") perform define;
    if (input == "delete") perform delete;
    if (input == "modify") perform modify;
```

This is the start point, each input will be matched as start. Here the program choose which kind of input is and will perform the respective part of the program.

```
modify
    if (VariableTable.getVersus(id) == "write") pc.write(id, value);
    VariableTable.modify(id, value);
    VariableTable.notifyRules(id);
    dbc.write(id, value);
    sc.write("OK");
```

As we can see from The Language the input string modify has two fields to fill: the name of the variable to modify and the value of the status which has to assume. Then, the program according with the requirements, check if that variable is an actuator. In a positive case it will first update the value of that variable in the Package through the Package Communicator, otherwise it will start changing the field value of that variable in the Variable HashMap. Now that the value of a variable is change it must notify all the Rules in which it appears on the condition. At the end will be updated the Database through the Database communicator and the Server Communicator will indicate at the UI that everything went well.
if (RuleTable.contain(id)) {
    RuleTable.kill(id);
    RuleTable.delete(id);
    VariableTable.deleteRulesFromVariable(id);
}

if (VariableTable.contain(id)) {
    if (VariableTable.hasRules(id)) {
        sc.write("error");
    } else {
        VariableTable.delete(id);
    }
}

dbc.delete(id);
sc.write("OK");

If the input was a delete command it contain just the name of the rule or variable to cancel, once established whether it is a rule or a variable, it will take two different actions: -in the case it is a rule name, the thread that is running that rule will be killed, its entry containing the handler and the identifier in the HashMap will be delete and also the references present in the variable which was present in its condition will be delete. -in the case it is a variable name, first it must be checked if this variable is in any rule condition, to prevent a disaster, if is not the program will proceed to delete the variable from the Variable Table.

define
VariableTable.insert(id, pack, realname, value);
if (VariableTable.getVersus(id) == "write") pc.write(id, value);
dbc.insert(id, pack, realname, value);
sc.write("OK");

To insert a variable must be specified as well as the name and the value also the package identifier and the realname inside the package, these data are not asked at the user, but insert automatically from the UI. Thus, the parser will first insert the variable in the Variable HashMap after send the command to change the status at the Package and at the end as every time update the DB and notify the UI.

rule
Rule handle = new Rule();
RuleTable.insert(id, handle);
Thread thread = new Thread()
    run()
    while(true){
ck = bool_exp
if (ck==1){
    command trueaction;
} else{
    command falseaction;
}
Thread.wait();
}

handle.setGuard(thread);
dbc.insert(ide, condition, trueaction, falseaction);
sc.write("OK");

Define a rule is a bit more complicated than define a variable. First of all the parser has
to create an Object of the class Rule that will be employed as handle to reach the Thread.
Then, once that the handle is created it will be put in the Rule HashMap with the name of
the rule as a key. At this phase, a new Thread is created in which will be run a cycle that
play always the same role: check the condition of the rule, perform the consequent action
and wait. Now a link between the rule handle and the thread is made, the DB update and
a confirm it send to the UI.

command
if (VariableTable.getVersus(ide) == "write") pc.write(ide, value);
VariableTable.modify(ide, value);
VariableTable.notifyRules(ide);
dbc.modify(ide, value);
sc.write("OK");

Perform an action means change the state of one or more variables, then in the input must
be specified the name of the variable and the value which they have to assume. As each
time that the program need to change a variable, before check the type to know if it is
necessary change its status in the package. After, it will update the value in the Variable
HashMap, notify the rule, update the Database and a confirm it send to the UI.

bool_exp

return value_of_expression;

To receive the value of the expressions, the program effect several call depending from the
complexity of the expression, it is here that the variable "ide" present on the expression
take a value. Observing the language starting from bool_exp, it is easy understand the
flow that transform an expression from a string to an integer value.
Chapter 6

Conclusion

Certainly the reader now it is so passionate at the reading, that did not notice that the thesis is finished and he was expecting other chapters in which would like to see the implementation of all the packages, the testing phases and the reaction from the market when this new product will be introduced, but unfortunately beautiful things come to an end. I begin this thesis claiming that the subject of the document is evolve an Idea to become an enterprise, maybe in the introduction whom was reading thought that it was just an arrogant exclamation, but I hope along the lecture he changed his mind. Through the chapters it is possible to see which are the basis and the references that allow us to say:" We can make the Domotic Easy and Low Cost to everyone". In the Business Plan we have got to verify that our Idea is innovative and competitive compared with the products already present in the market. Obviously, not all that glitters is gold, we acknowledge the presence of strong competitor, and especially we have discovered the existence of a similar products overseas. Then taking in to account the consideration made in the Business Plan, we pass at the design of the product. Reading in this document, design the structure of the system may appear an easy challenge, but it is not. Despite we already knew all the requirements, write down the solution that you have in mind, concretize an Idea, it never is an easy work. However the most difficult and interesting part in the same time is always the implementation, it is largely described in this thesis. By what we write the reader should achieve such a knowledge of the system, that would be able to develop by himself the product. We know that can be observed the lack of the testing phase in the thesis, but sadly we did not developed other packages yet, then the testing phase will be a future work. To sum up, we are a great team full of hope in what we are doing. This document summarizes a part of the work done so far, not only with the goal of write a thesis but with the goal of use what we learned during the studies to create our future. Lot of efforts has been done until now to reach this step, and for sure we will face with
many other obstacles during the route addressed to build an enterprise, but this does not discourage us, we will go forward as we always have done.

6.1 Future work

We write about what we done so far, but the route we intent to do is still long and full of pitfalls. In this section it is described the pending work that we have to do to develop this product, and some improvement that can be done at the work already done.

6.1.1 Improvement

Some part has not been developed as write in the Design Documentation because was not concern with the topic of this thesis, and due to the lack of time we have given priority to other but this does not mean that we will skip that part.

Communication protocol
We developed a protocol that just allow the communication between the packages but does not furnish at all a security layer. The protocol utilized can be easily hacked with a MITM (Man in the Middle) technique, indeed it is sufficient to masquerade as another package to be capable to communicate with the system. Due to this reason we want to implement a security layer in the communication protocol.

Group of variables
In the actual state of the system, if the user want to use a variable to describe the behavior of a group of variables have to do a tedious work with the rules. What we want is to esemplificate this feature with a new command that should be capable to express a similar concept: "When light_group is turned off turn off all the light".

6.1.2 Developing

Talking about software and hardware not everything it is been developed, some part has to be designed and implemented yet. This phase will require a consistent hours of hard work but has to be done to be inline with what we want to produce.

Packages
Until now we developed just the Central Unit, the other packages have been omitted. Their design and development will not be for sure as complex as it was to the Central Unit however it is an important work to do as soon as possible. Actually we already have a prototype of some packages but it is really raw and we need to redesign it to make it ready to the market.

**Check the presence of inconsistent ECA rules**

Probably the reader has noticed that the use of the rules can create several problems. If the user does not care about the correctness of the rules that insert into the system, an inconsistency between those rules may appear. To be more concrete, imagine a rule that opens a door when a light is switched on and another rule that switches off the light when such door is opened, obviously it is impossible to switch on the light due to an inconsistency. But even worst case may happen; it can be even formed a cycle if the rules inserted are not correct. To avoid this problem we need to develop a checker that controls the correctness of the rules inserted compared with the rules already present in the system. This is a difficult work, but it is indispensable if we want to create a good product.

### 6.1.3 Testing

The last phase in which we will go through it is the Testing. When all the package will be ready, before to celebrate the accomplishment of the route it is necessary do several tests. The test will be aimed to stress the component and punish the software, doing so we will discover also the limit of the product. Probably will even let it test by some possible future user, that should represent the target category at which we are aiming.
Bibliography


Appendices
### Figure 1: Expected Revenue by product

<table>
<thead>
<tr>
<th>Italian market</th>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central Unit</td>
<td>47,920</td>
<td>119,600</td>
<td>299,500</td>
</tr>
<tr>
<td>Lumus</td>
<td>167,200</td>
<td>418,000</td>
<td>1,045,600</td>
</tr>
<tr>
<td>Lumus Switch Wall</td>
<td>89,040</td>
<td>222,600</td>
<td>558,500</td>
</tr>
<tr>
<td>Walby</td>
<td>124,280</td>
<td>303,200</td>
<td>758,600</td>
</tr>
<tr>
<td>Oco</td>
<td>18,320</td>
<td>45,800</td>
<td>114,500</td>
</tr>
<tr>
<td>Locker</td>
<td>9,520</td>
<td>23,800</td>
<td>59,500</td>
</tr>
<tr>
<td>Sensy</td>
<td>111,680</td>
<td>279,200</td>
<td>698,000</td>
</tr>
<tr>
<td>Infuser</td>
<td>95,760</td>
<td>239,400</td>
<td>598,500</td>
</tr>
<tr>
<td><strong>Total revenue Italian market</strong></td>
<td>660,720</td>
<td>1,641,800</td>
<td>4,129,600</td>
</tr>
<tr>
<td><strong>Annual growth rate</strong></td>
<td>15.00%</td>
<td>15.00%</td>
<td></td>
</tr>
</tbody>
</table>

### Figure 2: Plan of marketing costs

<table>
<thead>
<tr>
<th>Italian market</th>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fairs and Exhibitions (+mission)</td>
<td>20,000</td>
<td>20,000</td>
<td>20,000</td>
</tr>
<tr>
<td>Advertising on press</td>
<td>40,000</td>
<td>50,000</td>
<td>100,000</td>
</tr>
<tr>
<td>Radio advertising</td>
<td>20,000</td>
<td>50,000</td>
<td>100,000</td>
</tr>
<tr>
<td>Internet advertising</td>
<td>20,000</td>
<td>50,000</td>
<td>100,000</td>
</tr>
<tr>
<td>TV advertising</td>
<td>0</td>
<td>50,000</td>
<td>200,000</td>
</tr>
<tr>
<td><strong>Total revenue Italian market</strong></td>
<td>100,000</td>
<td>230,000</td>
<td>540,000</td>
</tr>
<tr>
<td><strong>Impact on sales Italy</strong></td>
<td>15.44%</td>
<td>13.03%</td>
<td>51.08%</td>
</tr>
</tbody>
</table>

### Figure 3: Investment Plan

<table>
<thead>
<tr>
<th>Investments materiali</th>
<th>Anno 1</th>
<th>Anno 2</th>
<th>Anno 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offices</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Laptop</td>
<td>6,000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Tablet</td>
<td>1,600</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Printer</td>
<td>1,000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PC</td>
<td>2,600</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total investments in tangible assets</strong></td>
<td>8,809</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Investments immateriali</th>
<th>Anno 1</th>
<th>Anno 2</th>
<th>Anno 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Establishment Expenses</td>
<td>2,600</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Certification CE Expenses</td>
<td>4,600</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Patents Expenses</td>
<td>30,000</td>
<td>50,000</td>
<td></td>
</tr>
<tr>
<td>Purchase of new software</td>
<td>5,000</td>
<td>1,000</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total investments in intangible assets</strong></td>
<td>51,600</td>
<td>31,000</td>
<td>51,000</td>
</tr>
<tr>
<td><strong>Total investments</strong></td>
<td>45,009</td>
<td>31,000</td>
<td>51,000</td>
</tr>
</tbody>
</table>
Figure 4: Plan of staff costs

<table>
<thead>
<tr>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales</td>
<td>660,720</td>
<td>100,0%</td>
</tr>
<tr>
<td>Sales Abroad</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Total Sales</td>
<td>660,720</td>
<td>100,0%</td>
</tr>
<tr>
<td>Opening Balance</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Purchase of Commodities</td>
<td>338,479</td>
<td>51,2%</td>
</tr>
<tr>
<td>Change in inventory</td>
<td>52,172</td>
<td>10,0%</td>
</tr>
<tr>
<td>Cost of goods sold</td>
<td>398,935</td>
<td>59,4%</td>
</tr>
<tr>
<td>Subcontracting</td>
<td>25,278</td>
<td>3,4%</td>
</tr>
<tr>
<td>Direct material</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Total variable manufacturing costs</td>
<td>295,573</td>
<td>40,6%</td>
</tr>
<tr>
<td>Gross Profit</td>
<td>375,147</td>
<td>55,6%</td>
</tr>
<tr>
<td>Profit</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Transport</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>variable Commercial Costs</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Net profit</td>
<td>375,147</td>
<td>55,6%</td>
</tr>
<tr>
<td>Management costs of technical production</td>
<td>151,200</td>
<td>15,3%</td>
</tr>
<tr>
<td>Administrative Management</td>
<td>100,000</td>
<td>15,1%</td>
</tr>
<tr>
<td>Total Administrative costs</td>
<td>151,200</td>
<td>15,3%</td>
</tr>
<tr>
<td>Total operating costs</td>
<td>574,800</td>
<td>86,7%</td>
</tr>
<tr>
<td>Profit from Operating activities</td>
<td>-4,653</td>
<td>-0,7%</td>
</tr>
<tr>
<td>Interest on capital</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Interest on capital</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Borrowing Costs</td>
<td>0</td>
<td>0,0%</td>
</tr>
<tr>
<td>Earnings Before Tax</td>
<td>-4,997</td>
<td>-0,8%</td>
</tr>
</tbody>
</table>

Figure 5: Income Statement
### Figure 6: Balance sheet - assets

<table>
<thead>
<tr>
<th>Assets</th>
<th>Year 1</th>
<th>%</th>
<th>Year 2</th>
<th>%</th>
<th>Year 3</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash</td>
<td>0</td>
<td>0.0%</td>
<td>7,428</td>
<td>0</td>
<td>695,774</td>
<td>59.5%</td>
</tr>
<tr>
<td>Immediate cash</td>
<td>0</td>
<td>0.0%</td>
<td>7,428</td>
<td>0</td>
<td>695,774</td>
<td>59.5%</td>
</tr>
<tr>
<td>Receivable trade</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0%</td>
</tr>
<tr>
<td>Receivable abroad</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0%</td>
</tr>
<tr>
<td>Cash deferred</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0%</td>
</tr>
<tr>
<td>Closing balance of commodities and FG</td>
<td>70,173</td>
<td>66.7%</td>
<td>175,396</td>
<td>70.2%</td>
<td>438,450</td>
<td>36.6%</td>
</tr>
</tbody>
</table>

| Current assets              | 70,173 | 66.7%  | 182,833 | 73.3%  | 1,134,223 | 92.1%  |
| Non-current assets          | 6,600  | 7.7%   | 5,200   | 2.5%   | 3,800    | 0.3%   |
| Financial assets            | 36,000 | 39.3%  | 35,200  | 26.7%  | 87,600   | 7.9%   |
| Net capital assets          | 45,400 | 100.0% | 249,433 | 100.0% | 1,231,823 | 100.0% |

### Figure 7: Balance sheet - liabilities

<table>
<thead>
<tr>
<th>Liabilities</th>
<th>Anno 1</th>
<th>%</th>
<th>Anno 2</th>
<th>%</th>
<th>Anno 3</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bank overdrafts</td>
<td>11,493</td>
<td>15.9%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bank overdrafts - current</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Suppliers of raw materials</td>
<td>39,952</td>
<td>69.9%</td>
<td>37,775</td>
<td>65.7%</td>
<td>179,947</td>
<td>46.7%</td>
</tr>
<tr>
<td>Other short term liabilities</td>
<td>9,150</td>
<td>15.9%</td>
<td>21,267</td>
<td>39.4%</td>
<td>49,050</td>
<td>12.9%</td>
</tr>
<tr>
<td>Current liabilities</td>
<td>65,659</td>
<td>69.9%</td>
<td>92,468</td>
<td>39.9%</td>
<td>251,997</td>
<td>10.2%</td>
</tr>
<tr>
<td>Non-current liabilities</td>
<td>15,975</td>
<td>12.2%</td>
<td>65,593</td>
<td>26.2%</td>
<td>132,760</td>
<td>10.8%</td>
</tr>
<tr>
<td>Share capital</td>
<td>50,000</td>
<td>39.9%</td>
<td>50,600</td>
<td>20.0%</td>
<td>50,000</td>
<td>4.1%</td>
</tr>
<tr>
<td>Net income</td>
<td>4,967</td>
<td>5.9%</td>
<td>47,124</td>
<td>15.9%</td>
<td>789,939</td>
<td>63.4%</td>
</tr>
<tr>
<td>Revenue and net income of the year</td>
<td>54,933</td>
<td>44.9%</td>
<td>52,987</td>
<td>17.9%</td>
<td>881,976</td>
<td>70.5%</td>
</tr>
</tbody>
</table>

| Liabilities + Equity      | 115,573| 100.0%| 249,433| 100.0%| 1,231,823| 100.0%|

### Figure 8: Statement of cash flows

<table>
<thead>
<tr>
<th>Cash flows</th>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating income</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Operating expenses</td>
<td>-1,000</td>
<td>-1,000</td>
<td>-1,000</td>
</tr>
<tr>
<td>Net operating income</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Addition of receivables</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Addition of receivables paid</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Addition of current liabilities to suppliers of current assets</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Addition of current liabilities to suppliers of capital</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Change in net working capital</td>
<td>-100,000</td>
<td>-100,000</td>
<td>-100,000</td>
</tr>
<tr>
<td>Cash flows from operations</td>
<td>-10,118</td>
<td>80,298</td>
<td>709,796</td>
</tr>
<tr>
<td>Investment in long term asset</td>
<td>-10,000</td>
<td>-10,000</td>
<td>-10,000</td>
</tr>
<tr>
<td>Trade receivables</td>
<td>-10,000</td>
<td>-10,000</td>
<td>-10,000</td>
</tr>
<tr>
<td>Financial liabilities</td>
<td>80,000</td>
<td>80,000</td>
<td>80,000</td>
</tr>
<tr>
<td>Net balance</td>
<td>-10,118</td>
<td>80,298</td>
<td>709,796</td>
</tr>
<tr>
<td>Borrowing cash</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Proceeds and losses</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cash flows</td>
<td>-10,118</td>
<td>80,298</td>
<td>709,796</td>
</tr>
<tr>
<td>Opening balance</td>
<td>0</td>
<td>-10,118</td>
<td>7,410</td>
</tr>
<tr>
<td>Closing balance</td>
<td>0</td>
<td>-10,118</td>
<td>7,410</td>
</tr>
</tbody>
</table>

Claudio Tesei
Figure 9: Ratios