



MIDDLE EAST TECHNICAL UNIVERSITY
CENG491 - COMPUTER ENGINEERING DESIGN I

Detailed Design Report

A Smart Home Automation and
Energy Efficiency System



v1.0

Ankara 2012

Preface

The design document is prepared according to the “IEEE Standard for Information Technology - Systems Design - Software Design Descriptions - IEEE Std. 1016 - 2009”.

The design document gives information about architectural, logical and user interface viewpoint of the Smart Home Automation and Energy Efficiency System.

The first section of this document includes design document purpose, scope of project, target audience, data dictionary, abbreviation, glossary and definition, and references.

The second section of this document depicts overview of the system

The third section of this document mentions about design considerations in particular design assumptions, dependencies and constraints

The fourth section of this document tries to illustrate a data dictionary. It is provided in order to give a detailed description of the system data objects, their attributes and methods.

The fifth section of this document presents the user interface and its functionalities.

The sixth section of this document tries to illustrate detailed design of the project.

The seventh section of this document gives further information about libraries and tools planned to make use of

The last section of the document gives a brief conclusion about the project

History Table

Date	Version	Description	Writer
December 3, 2012	V1.0	Document is created for the first time.	Team Members
December 27, 2012	V1.1	Document is edited according to new constraints.	Team Members
January 16, 2012	V1.1	Document is revised according to feedbacks.	Team Members

Table of Contents

1. Introduction	7
1.1. Purpose	7
1.2. Scope of the Document.....	8
1.3. Audiences.....	8
1.4. Abbreviation, Glossary and Definition	9
1.5. References	10
2. System Overview	11
3. Design Considerations.....	13
3.1. Assumptions.....	13
3.2. Dependencies Constraints.....	14
3.2.1. Time Constraints	14
3.2.2. Software and Hardware Constraints	14
3.2.2. Control Requirements	144
3.2.2. Security Constraints	145
4. Data Design	16
4.1. Data Description	16
4.1.1. Statistic Table	16
4.1.2. Machine Table.....	16
4.2. Data Dictionary.....	17
4.2.1. Entities	17
4.2.1.1. Entities of HomeUI	17
4.2.1.2. Entities of Authentication.....	17
4.2.1.3. Entities of Machine	18

4.2.1.4. Entities of Schedule.....	18
4.2.1.5. Entities of Alarm	18
4.2.1.6. Entities of Timer.....	18
4.2.2. Methods.....	19
4.2.2.1. Methods of HomeUI.....	19
4.2.2.2. Methods of Authentication	20
4.2.2.3. Methods of Machine.....	21
4.2.2.4. Methods of Schedule	22
4.2.2.5. Methods of Statistic.....	23
4.2.2.6. Methods of MicroController	23
4.2.2.7. Methods of Alarm.....	23
4.2.2.8. Methods of Timer	24
4.2.2.9. Methods of Xbee	24
5. User Interface Design.....	25
5.1. Overview of User Interface	25
5.2. Screen Images.....	26
6. Detailed Design	31
6.1. Graphical UI Framework.....	31
6.1.1. Class HomeUI.....	32
6.1.2. Class Authentication	32
6.2. Raspberry Pi	33
6.2.1. Class Machine.....	34
6.2.2. Class Schedule	34
6.2.3. Class Statistic.....	34
6.3. Pic Controller.....	35
6.3.1. Class MicroController.....	35

6.3.2. Class Alarm.....	35
6.3.3. Class Timer	35
6.4. Xbee.....	36
6.4.1. Class Xbee	36
6.4.2. Class Master Xbee	36
6.4.3. Class Slave Xbee.....	36
6.5. Structural Classes Association Diagram.....	37
7. Libraries and Tools.....	38
7.1. MYSQL	38
7.2. QT.....	38
7.3. C++	39
7.4. C.....	39
7.5. MPLAB X IDE.....	39
7.6. StarUML - The Open Source UML/MDA Platform.....	40
7.7. Eclipse IDE.....	40
7.8. Apache Web Server.....	41
7.9. XBEE- Python API.....	42
8. Conclusion.....	43

Figure List

Figure 1 Abbreviation, Glossary and Definition	9
Figure 2 Deployment Diagram	12
Figure 3 Statistic Table	16
Figure 4 Machine Table	17
Figure 5 Register screen	27
Figure 6 Login screen	27
Figure 7 Machine selection screen	28
Figure 8 Machine options screen	28
Figure 9 Set and schedule screen	29
Figure 10 Statistics screen	29
Figure 11 Current status screen (On)	30
Figure 12 Current status screen (Off)	30
Figure 13 Graphical UI Framework	31
Figure 14 Raspberry Pi Package	33
Figure 15 Pic Controller Package	35
Figure 16 Xbee Communication Package	36
Figure 17 Class Association Diagram	37

1. Introduction

1.1. Purpose

This software design document aims to provide information about the design details of a Smart Home Automation and Energy Efficiency System. The document will present system's physical layouts to show which pieces of the software run, and which type of the Zigbee are used on which pieces of the hardware. Moreover, it will present what type of the communication path is used to connect the system hardware component.

This document will show how the system is composed from. While showing bottom level of elements of the system, document will show their construction purpose, and different kind of static relationship that exist among them. In addition, interaction of the user will be included by describing the properties of user interface and showing representative screenshots of the interface.

The document presents a number of different design views to depict different aspects of the system. It is intended to capture and convey the significant design decisions which have been made on the system.

1.2. Scope of the Document

This document contains complete description of the design of the Smart Home Automation and Energy Efficiency System. It consists of class diagrams, association of the classes, activity diagram to show work flow of the some critical parts of the system, deployment diagram to show physical parts of the system. The design views incorporated are explained in depth and justified for use within this document.

1.3. Audiences

- a) Dr. Onur Tolga Şehitoğlu
- b) Serdar Çiftçi
- c) “CENG 491 - Computer Engineering Design I” teaching assistants
- d) Arçelik A.Ş

1.4. Abbreviation, Glossary and Definition

Abbreviation	Description
IEEE	Institute of Electrical and Electronic Engineers
UML	Unified Modelling Language
CENG	Computer Engineering
GUI	Graphical User Interface
API	Application Programming Interface
IDE	Integrated Development Environment
PIC	Peripheral Interface Controller
MDA	Model-Driven Architecture
AT Commands	Attention Commands
ODBC	Open Database Connectivity

Figure 1 Abbreviation, Glossary and Definition

1.5. References

IEEE Standard 1016-2009 for Information Technology - Systems Design - Software Design Description

Fowler, M., UML Distilled Third Edition A brief guide to the standard object modelling language, Addison Wesley.

Douglass, P. B., Real Time UML, Addison Wesley

2. System Overview

This section provides an overview of the HomeSmartHome Project which aims to control most devices and appliances by households. HomeSmartHome is designed to operate Zigbee enabled devices by allowing the user to control these devices remotely or via schedules/rules with a purpose to lower the house's energy consumption and to ease the daily lives of the inhabitants. It is important to mention that HomeSmartHome will not be a decision making product, control of the HomeSmartHome system is in the hands of the user and the system will only operate within the frame serviced to user

Basically, this project can be differentiated by three main components/technology such as PC, ZigBee wireless technology and PIC 18f4520 microcontroller which play important roles to develop a better, compatible, user friendly and cost efficient smart home system than other existing system

In general, this project consists of two major parts that need to be developed; hardware and software. For the hardware part it consists of Raspberry Pi as the system board which includes PIC 18f4520, Zigbee. HomeSmartHome project has five main functions that are accessible through a user interface: Security, Device Status, Device Schedule, and Device Statistics. These functions are explained in detail in the following chapters as summarized below in the diagram.

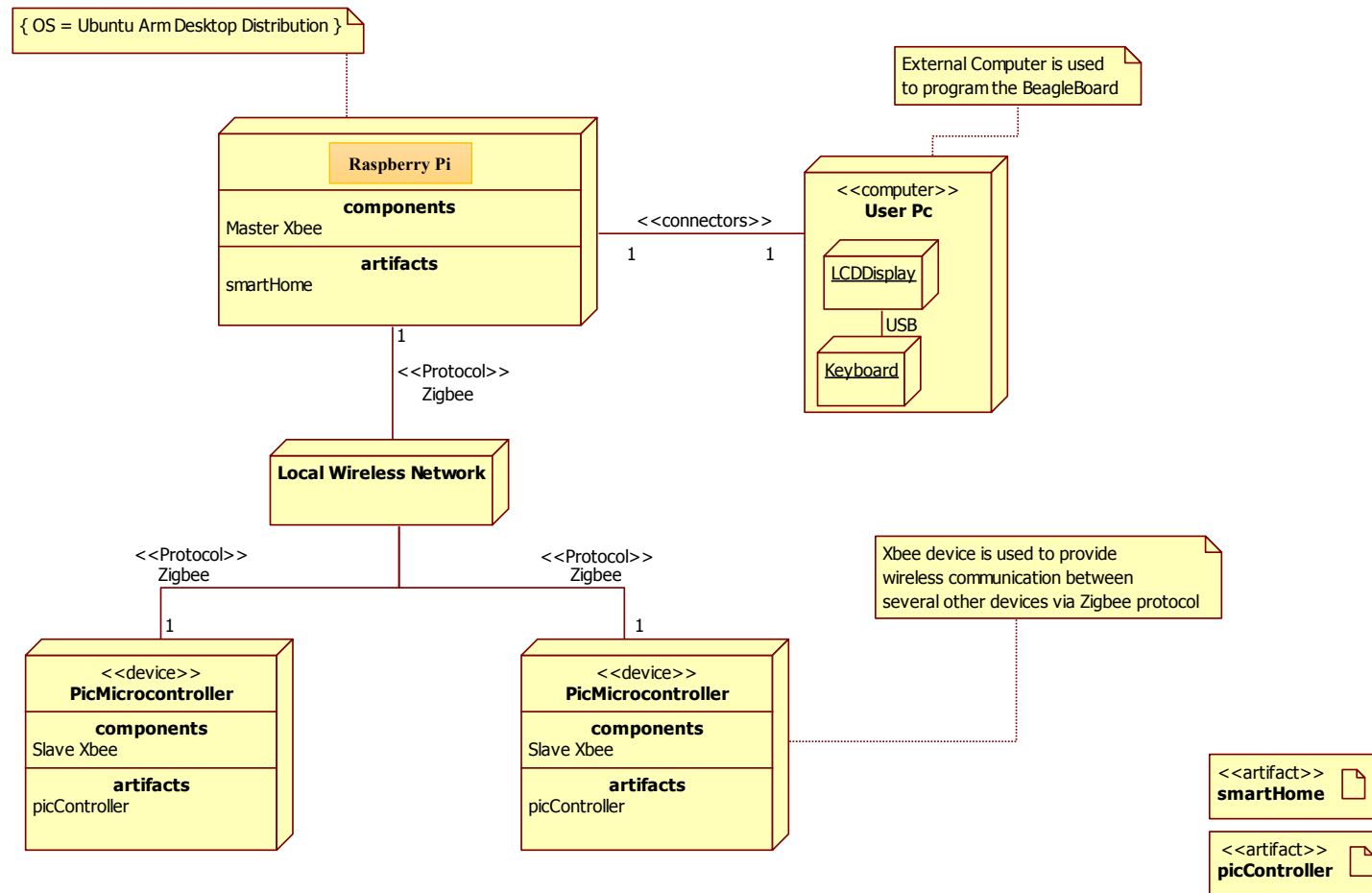


Figure 2 Deployment Diagram

3. Design Considerations

In this section of SDD, design considerations, basic assumptions, constraints and dependencies as well as goals and guidelines about the project are described.

3.1. Assumptions

The HomeSmarthome Control Layer will be designed for the GNU/Linux Operating System and Windows Operating System will be used for interfacing with Web-Server to X-Bee controller.

The Control Layer will contain all data of energy consumption statistics of the Zigbee enabled devices stored in database.

The Device Layer will be designed on Embedded System using PIC microcontrollers, and Raspberry Pi.

The User Layer will be developed and designed on web and can be used any operating system without restriction to control Zigbee enabled devices through System Control Layer.

3.2. Dependency Constraints

3.2.1. Time Constraints

Requirements part of the project is divided into pieces between members of the group HomeSmartHome in order to implement the project efficiently, and complete in less possible time. After each work process, the work done by each member will be combined and checked together by all of the group members. The project is intended to be finalized by the mid of June 2013 and detailed schedule has been made for this purpose

3.2.2. Software and Hardware Constraints

Software and hardware constraints as well as other constraints for the system are defined in detail in Software Requirements Specification Document. In a general manner, it is possible to mention that system must follow a variety of constraints of different phases of it such as operating system - Ubuntu, web-oriented operations - Apache implementations, collection of statistics – MySQL.

3.2.3. Control Requirements

Although home appliances are generally designed for analog usage, they can be managed remotely as well. Basic operations that are performed in analog way are the followings:

- Switch On / Off
- Turn Up / Down

Since Xbee radio modules' integration to home appliances is a requirement, home appliances are expected to have rs-232, rs-435 or external 4 pin outs to adapt Xbee devices to the main system. For the usage of Xbee coordinator it is necessary and sufficient that the controller has a usb connection.

Before sending data, "*data_command data*" command words are preceded to indicate which command will get data as the reply. *Data_command* takes integer 1 to 100 and labels packages according to the classes. To illustrate, 0x20 is used before temperature value to allow the control unit to know data belongs to temperature class.

3.2.2. Security Constraints

In terms of security issues, each and every Xbee module has unique addresses all around the world and it provides facilities that allow home appliances to adapt in a more convenient and efficient way.

Uniqueness of Xbee addresses and address information contained by related packages assures users that communication between end-point and control unit is completely unique and secure. On the other hand, to accomplish end-point communication it is definitely needed to know addresses of end-points. While manual addition of devices, it is automatically asked to enter related Xbee addresses. Information about these addresses are kept in the database and sponsor company is expected to update these files of table in the database.

4. Data Design

4.1. Data Description

In this project, we will keep the information about energy consumption in database management system. We will have two database tables namely “statistics” and “machine”.

From the statistics point of view, Power consumption are measured by the data that the related home appliance send to the system. The data comes to the control unit in a period (to be determined later). Control unit make use of the data it gets to calculate it with the equation, $Wattage \times hours\ used\ per\ day \times number\ of\ days\ used \div 1000$. Afterwards result is recorded to the statistics table in database. Data sending is performed in integer format and via Zigbee

4.1.1. Statistic Table

Statistic table keeps the information about energy consumption statistics data.

Column Name	Data Type	Description
Index	Integer	Primary key
MachineId	Long Integer	Machine unique id
EnergyConsumption	Double	Energy consumption in watt
TimeInterval	Date	Time interval

Figure 3 Statistic Table

4.1.2. Machine Table

Machine table keeps the machine's specific information.

Column Name	Data Type	Description
Index	Integer	Primary key
MachineId	Long Integer	Machine unique id
MachineName	Vartype String	Machine name
Watt	Double	Keeps information how much energy is consumed in one second

Figure 4 Machine Table

4.2. Data Dictionary

4.2.1. Entities

4.2.1.1. Entities of HomeUI

- a) selectedDevice store id of the current selected device via user interface
- b) RegisteredUser hold information about whether any user is registered to the system, or not

4.2.1.2. Entities of Authentication

- a) password hold current read password via login user interface
- b) passwordStoredFile store location of the where the registered user's password is stored

- c) failureCode store authentication failure code

4.2.1.3. Entities of Machine

- a) nameOfDevice store name of the device
- b) idOfDevice stored id of the device
- c) scheduleDataStoredFileName store where the schedule related to the device is stored
- d) status store current status of the device
- e) warning store current warning of the device

4.2.1.4. Entities of Schedule

- a) time store time information in the determined format
- b) date store date information in the determined format
- c) machineStatus showing the current status of the machine

4.2.1.5. Entities of Alarm

- a) AlarmCode store alarm code information

4.2.1.6. Entities of Timer

- a) counter used for calculating passed time from known interval

4.2.2. Methods

4.2.2.1. Methods of HomeUI

- a) HomeUI is used as constructor for creating an HomeUI type object, and for initializing the HomeUI type object's variables
- b) ~HomeUI is used as destructor for eradicating the HomeUI object with its variables
- c) getSelectedDeviceId is used as accessor for getting id of selected device in the UI
- d) initializeScreenUI is used for creating a new user interface with specified width and height, and creating a new user interface with specified colour and format
- e) registerNewUserDialog is used for creating a user interface to let user register to the system
- f) iconComponentManager is used for managing icons being shown near the machine name on the system UI component, and updating icon according to the machine name
- g) updateUI is used for refreshing the screen with new status of the machine, and refreshing the screen with urgent warning message resulting from not correctly using the machine
- h) showLineGraph is used for showing line graph sketched for demonstrating energy usage statistic of a machine to the screen
- i) getCommand is used for accessor for getting command triggered by the user's selection from user interface

4.2.2.2. Methods of Authentication

- a) Authentication is used as constructor for creating an Authentication type object, and for initializing Authentication type object's variables
- b) ~Authentication is used as destructor for eradicating the Authentication object with its variables
- c) searchFile is used for looking for file in which encrypted password of registered user stored in the directory, and determining what action should be done if file is deleted and if the file is changed by unauthorized user
- d) registerNewUser is used for storing new information of the registered user to the hidden file, and determining how the information should be stored in order improve reliability of the system
- e) encryptPassword is used for encrypting the password with salt, and one way hash function
- f) authenticationFailure is used for determining what actions should be done in case of authentication failure
- g) compareEncryptedPassword is used for comparing encrypted password taken from user with password taken from registered user
- h) createFile is used for managing file creation for storing encrypted password, and determining where the file is stored
- i) storeEncryptedPassword is used for storing encrypted password in the file which is already created

- j) readPassword is used for receiving already entered password from user interface, and storing taken password, temporarily
- k) failureMessage is used for accessing already determined failure message according to failure code

4.2.2.3. Methods of Machine

- a) Machine is used as constructor for creating an Machine type object, and for initializing Machine type object's variables
- b) ~Machine is used as destructor for eradicating the Machine object with its variables
- c) getNameOfDevice is used as accessor for getting name of the machine
- d) getWarning is used as accessor for getting warning occurred on the machine
- e) getStatus is used as accessor for getting current status of the machine
- f) getScheduleFileName is used as accessor for getting filename which is used for storing date and time information for scheduling the machine
- g) createScheduleFilename is used as creating a filename according to some reference data of a machine
- h) turnOn is used for turning on the machine
- i) turnOff is used for turning off the machine
- j) getMachineStatus is used for taking current status of the machine

- k) `getAndInterpretWarningMessage` is used for taking warning message and translating it to human readable form

4.2.2.4. Methods of Schedule

- a) `Schedule` is used as constructor for creating a `Schedule` type object, and for initializing `Schedule` type object's variables
- b) `~Schedule` is used as destructor for eradicating the `Schedule` object with its variables
- c) `getTime` is used as accessor for getting time information
- d) `getDate` is used as accessor for getting date information
- e) `setTime` is used as mutator for manipulating time information
- f) `setDate` is used as mutator for manipulating date information
- g) `createScheduleFile` is used for managing file creation for storing schedule information for the machine
- h) `writeScheduleFile` is used for writing date and time information to the file
- i) `readScheduleFile` is used for reading date and time information of the machine to start schedule
- j) `validateSchedule` is used for checking the schedule information with stored schedule information
- k) `finalizeSchedule` is used for finalizing schedule operation

4.2.2.5. Methods of Statistic

- a) Statistic is used as constructor for creating a Statistic type object, and for initializing Statistic type object's variables
- b) ~Statistic is used as destructor for eradicating the Statistic object with its variables
- c) createDatabaseTable is used for creating database table in order to store energy consumption
- d) writeToDatabase is used for storing energy consumption information
- e) readDataFromDatabase is used for taking stored data from database
- f) sketchLineGraph is used for sketching line graph according to data stored on the database

4.2.2.6. Methods of MicroController

- a) main is used as a control mechanism of the machine

4.2.2.7. Methods of Alarm

- a) Alarm is used as a constructor for creating an Alarm type object, and for initializing Alarm type object's variables
- b) ~Alarm is used as destructor for eradicating the Alarm object with its variables
- c) getAlarmCode is used as an accessor for getting alarm code information
- d) announceAlarm is used for transmitting the alarm code to the control component of the system

4.2.2.8. Methods of Timer

- a) Timer is used as a constructor for creating a Timer type object, and for initializing Timer type object's variables
- b) ~Timer is used as a destructor for removing the Timer object with its variables
- c) getTime is used as accessor for getting passed time value
- d) startTimer is used for starting the timer
- e) stopTimer is used for stopping the timer
- f) resetTimer is used for resetting the timer

4.2.2.9. Methods of Xbee

- a) sendData is used for transmitting data
- b) receiveData is used for receiving transmitted data

5. User Interface Design

5.1. Overview of User Interface

In order to present an overview of user interface, firstly, potential users of this system should be considered. It is thought that smart home system will be used by common household, which can include people from different age, background and language. Therefore, this interface should be so generic that one can easily adapt it; and also it should include all necessary functionalities to be used.

HomeSmartHome application will have an interface that lets user easily follow necessary steps. First, with the setup of home automation system, user will need to register to system by providing a username and password (Figure 6). This will enable different users with different privileges. Secondly, in every usage, user will confront with a login screen. After a successful login, all lists of machines that this application can control will be listed as “Home Appliances” (Figure 5). In this interface, user will choose the one that he/she wants to control.

After choosing the home appliance, Figure 8 demonstrates the interface user will confront with that is related to selected machine. Firstly, user will be able to select one of these actions:

- View current status
- Set and/or schedule the appliance
- View statistics

When user wants to display current status (Figure 9), whether the appliance is working or not with the corresponding error will be shown immediately to the user. Secondly, user will be able to set and schedule the appliance on the screen on Figure 7 where shutting down or starting options will be selected with a matching time. For instance, for a currently not-working washing machine, user will select “ON” icon and a time so that the machine will start to work at the provided time. Thirdly, user will be able to see the average and current consumption of the appliance in the statistics screen (Figure 10).

In addition, all necessary interfaces will have a “Back” button, which enables back navigation in the sequence of operations. Finally, all closing actions will end user session considering the software’s necessity for high security.

In order to give technical information about this interface, it must be mentioned that considering usage easiness, at least 600px x 600px of resolution will be considered while designing user interface.

5.2. Screen Images

Some representative screen images of the software can be seen as following:

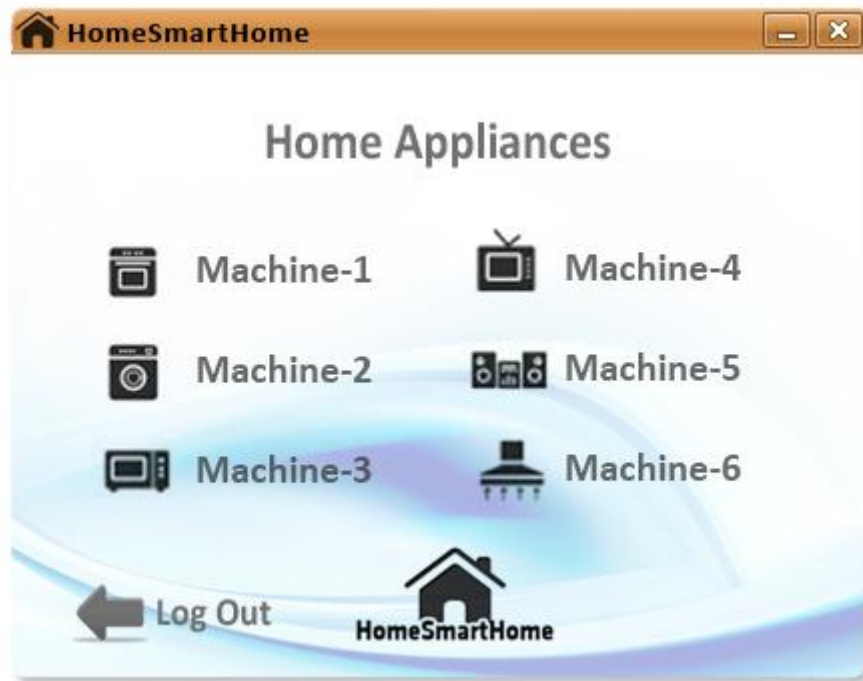


Figure 5 Register screen



Figure 6 Login screen



Figure 8 Machine options screen

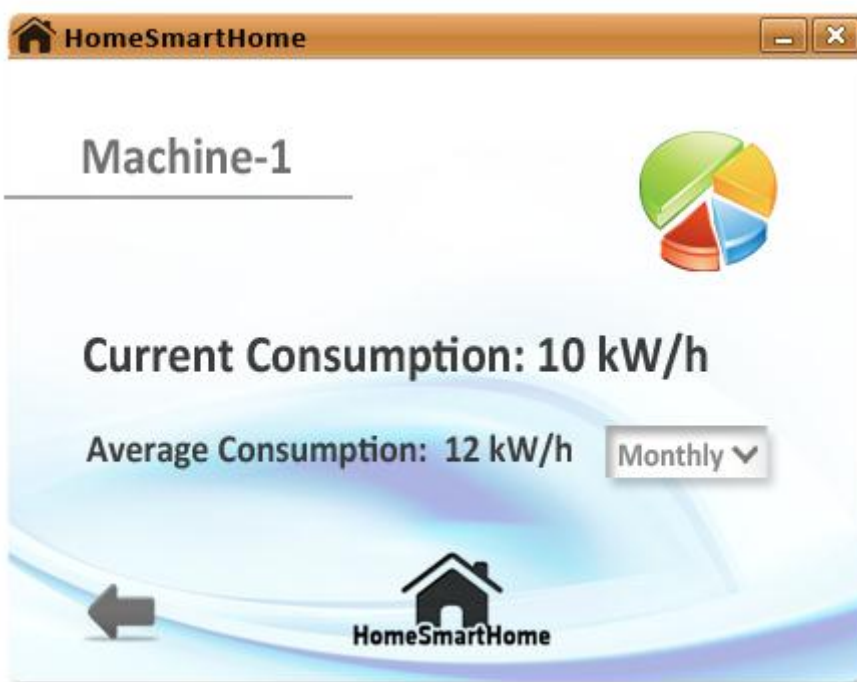


Figure 10 Statistics screen

Figure 11 Current status screen (On)

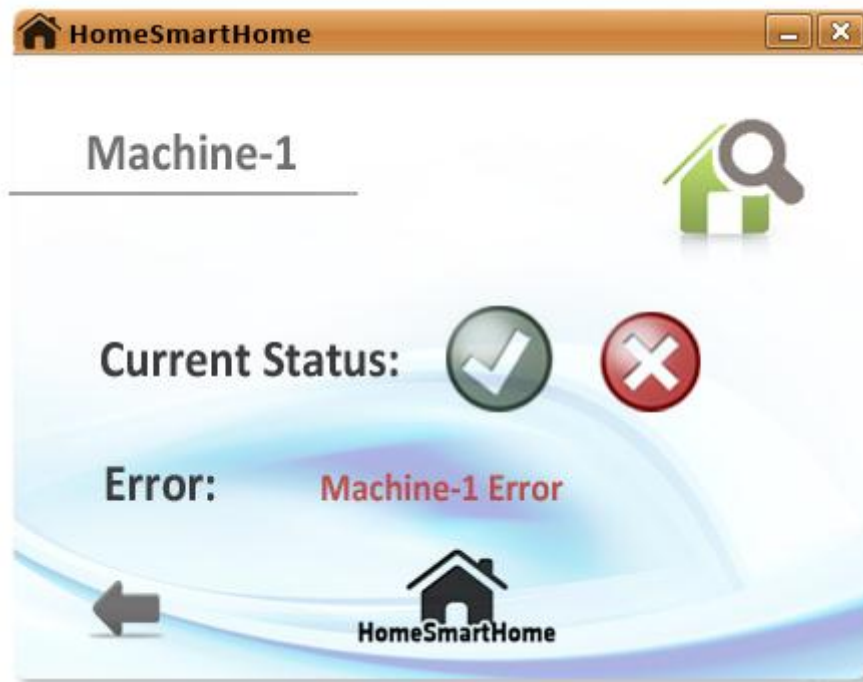


Figure 12 Current status screen (Off)

6. Detailed Design

6.1. Graphical UI Framework

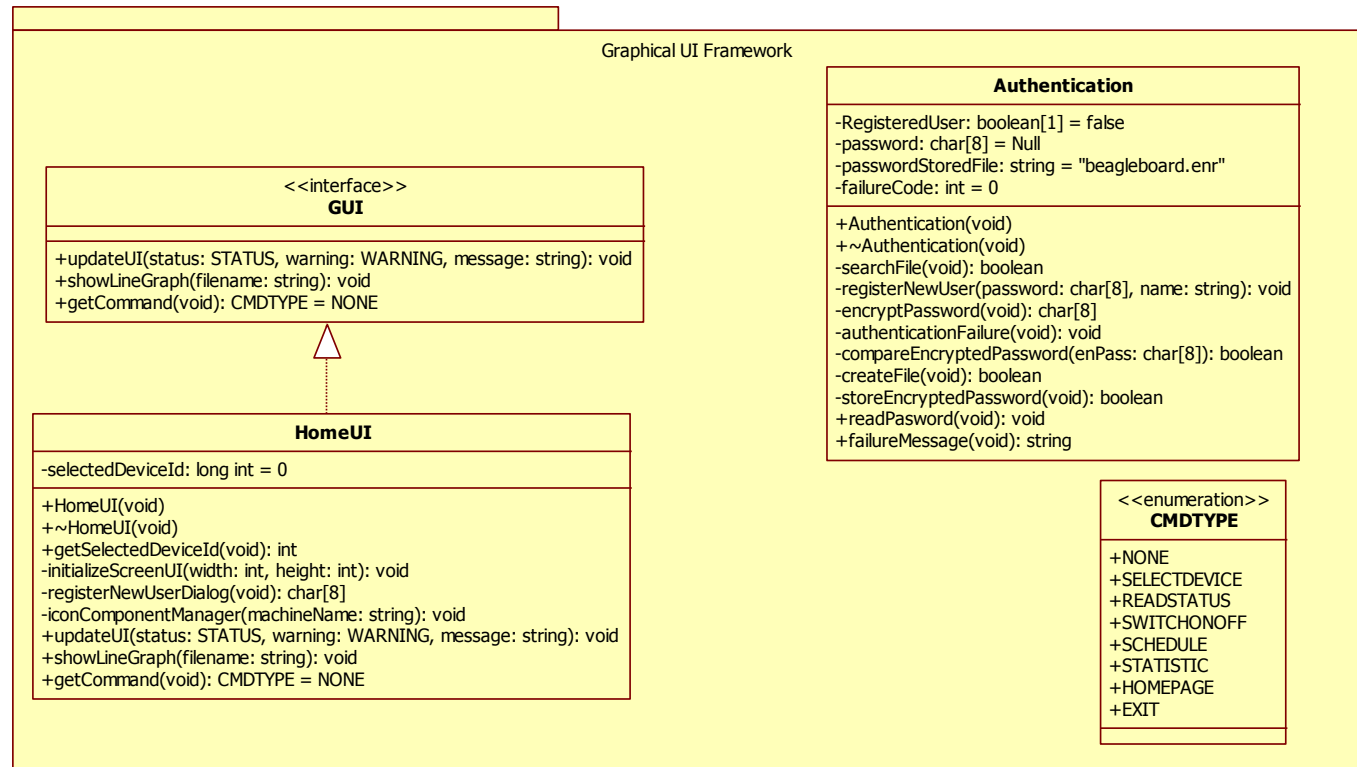


Figure 13 Graphical UI Framework

6.1.1. Class HomeUI

HomeUI class provides system-user connection and enables user to control system properties via interface. It is also provided that status of any machine -casually or in case of any problem- can be observed in addition to the chance of scheduling and getting statistics.

6.1.2. Class Authentication

User and password information of a registered user is kept in this class. It takes a part in encryption of the password entered and matching it with the registered password. In case of wrong password, related precautions planned to be implemented are defined in this class

6.2. Raspberry Pi

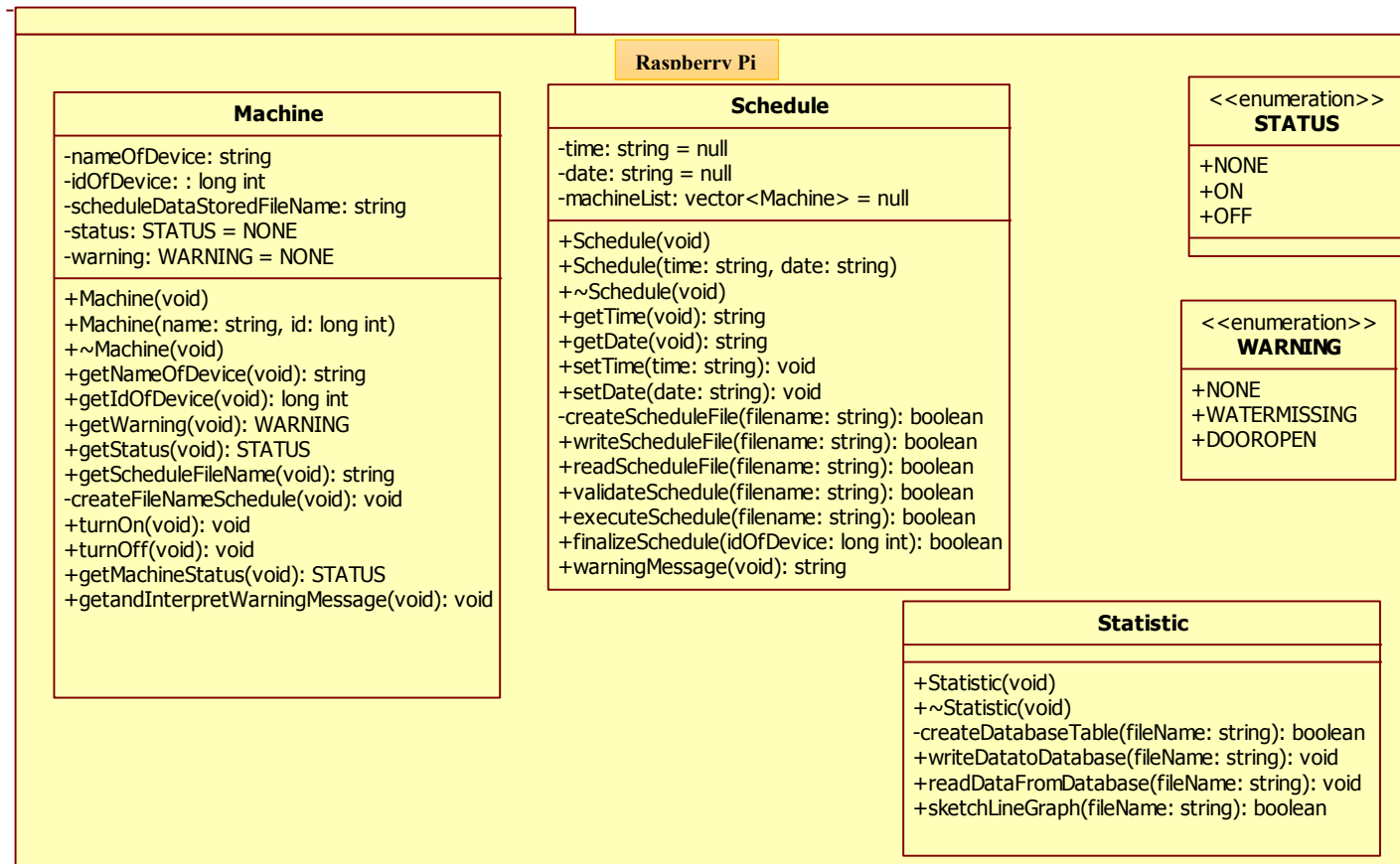


Figure 14 Raspberry Pi Package

6.2.1. Class Machine

It is not only used to control information of machines but to configure current status and to track unexpected issues come up whilst machine is on.

6.2.2. Class Schedule

Schedule and control operations for any machines working currently are handled by this class. Comparison between registered and entered information by user is carried out by class Schedule.

6.2.3. Class Statistic

This is the class that preserves information in database of a currently working machine about computed energy consumption for a unit of time and also processes the information getting from database

6.3. PIC Controller

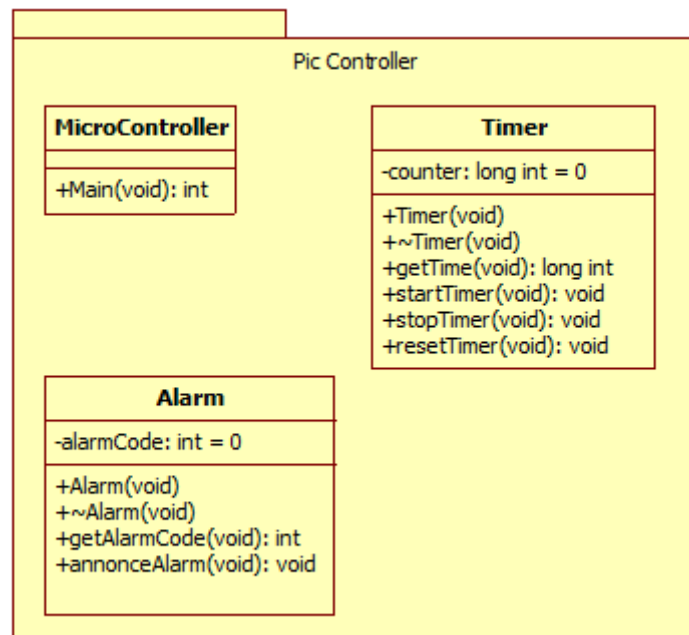


Figure 15 Pic Controller Package

6.3.1. Class MicroController

This class works on the machine and processes external data then responds to system properly.

6.3.2. Class Alarm

Alarm class defines problems that may be occurred in time and report them in the event that such thing occurs

6.3.3. Class Timer

Used for time-critical operations carried on machines.

6.4. Xbee

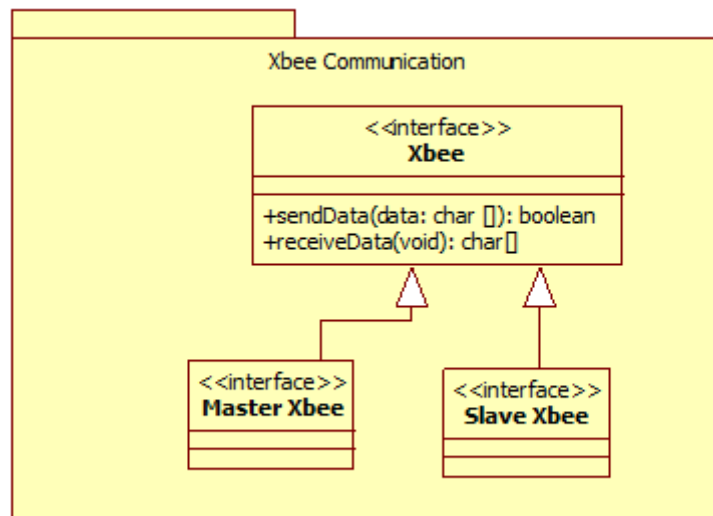


Figure 16 Xbee Communication Package

6.4.1. Class Xbee

Used to provide connection among machines

6.4.2. Class Master Xbee

It is working on Raspberry Pi and used for transmission among other end-points and Raspberry Pi

6.4.3. Class Slave Xbee

It is working on end-points (i.e machines) and plays a role in communication with the main system

6.5. Structural Classes Association Diagram

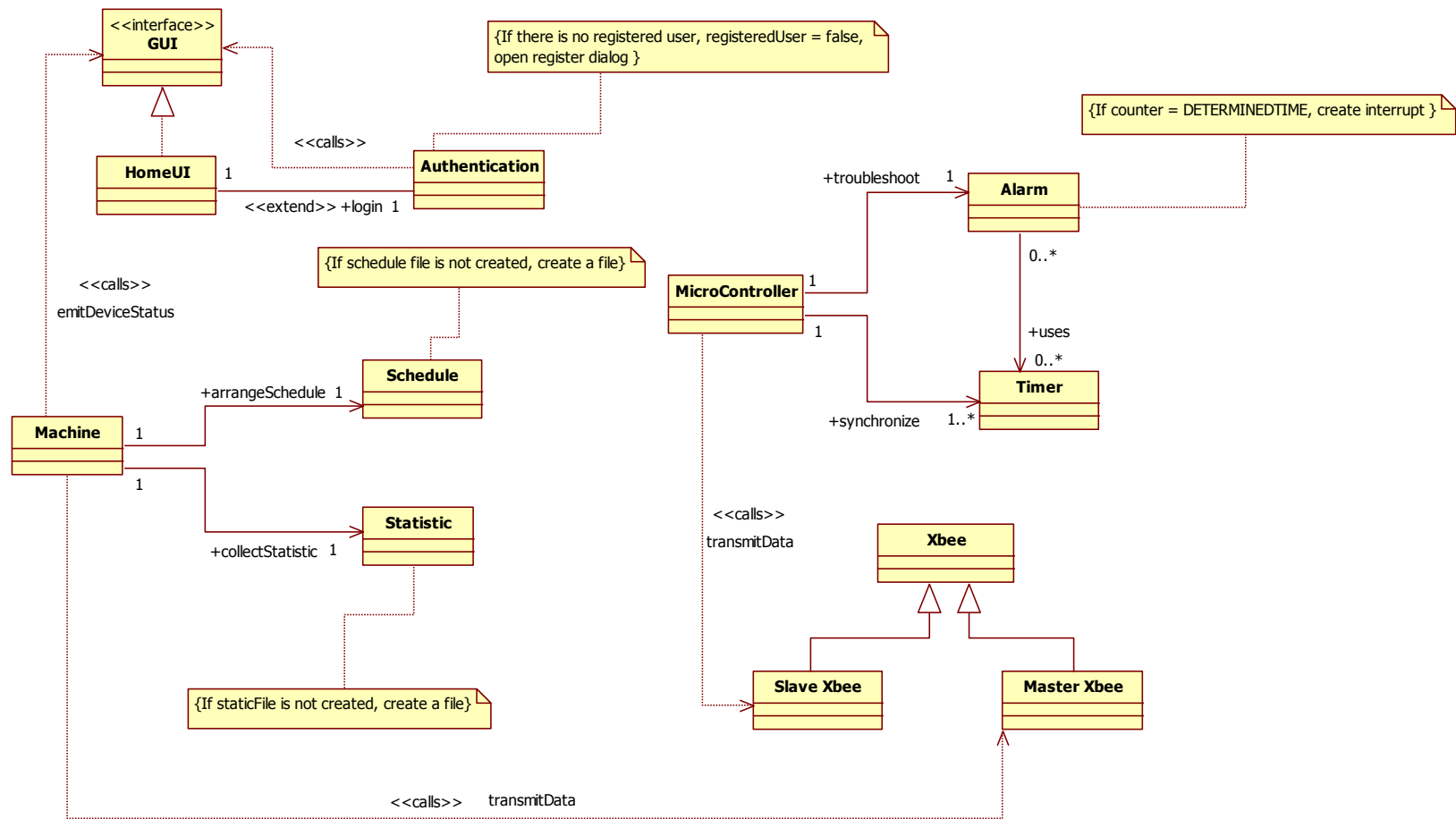


Figure 17 Class Association Diagram

7. Libraries and Tools

7.1. MYSQL

MySQL is a relational database management system that runs as a server providing multi-user access to a number of databases.

In this project, MYSQL enables carrying out data collection and statistics gathering. To illustrate, in the event that user prompts to get statistics for a desired period, MYSQL assists to get the information from database. It also takes a part once charts and diagrams are to be shown in terms of past consumption statistics.

7.2. Qt

Qt is a cross-platform application and UI framework. Using Qt, writing web-enabled applications once and deploy them across desktop, mobile and embedded operating systems without rewriting the source code is possible. Intuitive C++ class library, portability across desktop and embedded operating systems, integrated development tools with cross-platform IDE, high runtime performance and small footprint on embedded are the features of QT.

In HomeSmartHome project, for the two perspectives -application to run on Raspberry Pi, web application to access Raspberry Pi - regarding user interface are planned to be designed using Qt since it provides easiness, effectiveness and simplicity. Interfaces that are shown in the Figures 5 to 12 are to be implemented in Qt as same as possible.

7.3. C++

C++ is a statically typed, free-form, multi-paradigm, compiled, general purpose programming language. It is regarded as a middle level language, as it comprises a combination of both high-level and low-level language features.

In this project, C++ language is chosen to carry out programming process for Raspberry Pi.

7.4. C

C is one of the most widely used programming languages of all time, and there are very few computer architectures for which a C compiler does not exist.

From the perspective of PIC microcontroller, C language is widely used to develop code for it. Therefore, it is foreseen that instead of PIC assembly language choosing C language is more convenient to program PIC microcontroller in a high-level manner.

7.5. MPLAB X IDE

MPLAB X IDE is a software program that runs on a PC (Windows, Mac OS, and Linux) to develop applications for Microchip microcontrollers and digital signal controllers. It is called an Integrated Development Environment, or IDE, because it provides a single integrated “environment” to develop code for embedded microcontrollers.

From the project perspective, the reason behind choosing MPLAB X IDE is that this integrated development environment allows programming PIC microcontroller in a way that communicates with other peripheral devices. In order PIC device to send/ receive data via Zigbee protocol and Xbee devices to/from Raspberry Pi, PIC must be programmed using MPLAB X IDE.

7.6. StarUML - The Open Source UML/MDA Platform

StarUML is an open source project to develop fast, flexible, extensible, featureful, and freely-available UML/MDA platform running on Win32 platform. The goal of the StarUML project is to build a software modeling tool and also platform that is a compelling replacement of commercial UML tools such as Rational Rose, Together and so on.

StarUML is chosen to draw UML diagrams for each and every phase of the project. Class Diagrams shown in the Figures 2, 13, 14, 15, 16, 17 designed in StarUML Platform.

7.7. Eclipse IDE

Eclipse IDE is a kind of Eclipse platform that provides fully functional C and C++ Integrated Development Environment. Eclipse IDE includes many features like support for project creation and managed build for various toolchains, standard make build, source navigation, various source knowledge tools, such as type hierarchy, call graph, include browser, macro definition browser, code editor with syntax highlighting, folding and

hyperlink navigation, source code refactoring and code generation, visual debugging tools, including memory, registers, and disassembly viewers.

Eclipse IDE is very suitable for the project since it eases to integrate cross product operations and provides a convenient environment to develop required modules. Therefore, almost all code developing processes of the project are handled on Eclipse

7.8. Apache Web Server

The Apache HTTP Server is a collaborative software development effort aimed at creating a robust, commercial-grade, featureful, and freely-available source code implementation of an HTTP (Web) server. The project is jointly managed by a group of volunteers located around the world, using the Internet and the Web to communicate, plan, and develop the server and its related documentation. This project is part of the Apache Software Foundation. In addition, hundreds of users have contributed ideas, code, and documentation to the project. This file is intended to briefly describe the history of the Apache HTTP Server and recognize the many contributors.

In the project, besides accessing HomeSmartHome program via user interface on Raspberry Pi, it is planned to design and implement a web user interface connecting through apache web server. Therefore, it is necessary for the project to make use of facilities that apache web server provides since it is thought that web phase of the project, if implemented, would enable the system to gain much more portability.

7.9. Xbee- Python API

Xbee -API is a simple way to use python program that allows you to interact with an Xbee directly, sending AT commands.

In the project, this same API can be used to programmatically interact with Xbees quite easily. Setting this up to run, is just like any other python script. Therefore, it is well worth to make use of it while sending / receiving one Xbee to/ from to the others rather than doing the same thing in C language and C API.

8. Conclusion

Considering the way technology market advances, home automation and energy efficient systems will be used inevitably all around the world in the following years. As a matter of fact, not only for private house applications but also the applications for industry, these systems will gain a major impact for human beings lives. This smart home and energy efficiency oriented project is a very first step to discover new implementation fields and to integrate these devices with the existing ones to ease daily life as much as possible. To accomplish this in a generic way, in our model subsystems are mainly developed with two level of construction. At this point, the results are expected to be:

- A comprehensive checking of model's consistency and safety. The results reveal which model is not conformed with requirements proposed and which model is non-safety model that can lead to problems.
- A comprehensive checking of subsystem's integrity and performance. It can describe in detail the time and space costs of each module and some problems in communication among modules.