

---

---

# VOICE CONTROL BASED HOME AUTOMATION SYSTEM

---

---

16:332:567 SOFTWARE ENGINEERING COURSE PROJECT  
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

FALL, 2013

EDITED BY

LI XINYU

GU YUE

GUO JIAQI

NAYYAR VIDUR

SHU CHANG

ZHANG LUFAN

*The State University of New Jersey  
Rutgers*

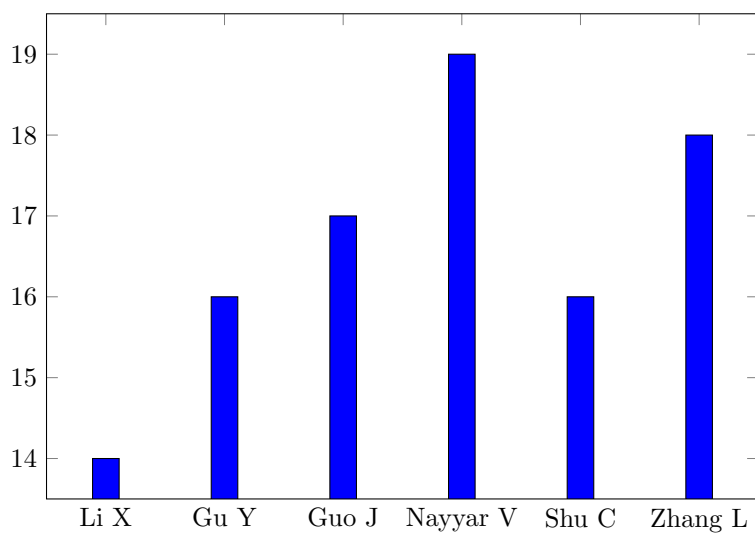
FOR DETAILS

[HTTPS://SITES.GOOGLE.COM/SITE/VCHOMEAUTOMATION/](https://sites.google.com/site/vchomeautomation/)

## 1 Contribution Breakdown

Interaction Diagrams									
aspect	UML diagrams	Prose Descr of Diag	Alt. Soln Description						
Score	10	10	10						
Li Xinyu									
Gu Yue			partly(6')						
Guo Jiaqi	Partly(17')								
Nayyar Vidur	Partly(3')								
Shu Chang			partly(4')						
Zhang Lufan									
Class + Specs									
Sts arch & Design									
Hardware design									
aspect	Class Diag and Description	Signatures	styles	pkg diag	database	map hardware interfacing	hardware requirements network & communication protocol [OTHERS]		
Score	5	5	5	2	3	2	3		
Li Xinyu									
Gu Yue									
Guo Jiaqi									
Nayyar Vidur									
Shu Chang									
Zhang Lufan									
Alg's & data struct									
User interface									
Testing Design									
Project management									
Ref									
aspect	Alg's & data struct	appearance	Prose Description	Testing Design	Doc merge	Proj coord	Plans	Ref	
Score	4	6	5	12	11	5	2	-5	
Li Xinyu									
Gu Yue									
Guo Jiaqi									
Nayyar Vidur									
Shu Chang									
Zhang Lufan									

Figure 1: contribution breakdown



# Contents

<b>1</b>	<b>Contribution Breakdown</b>	<b>2</b>
<b>2</b>	<b>Interaction Diagrams</b>	<b>4</b>
2.1	UML Diagrams . . . . .	4
2.2	Alternative solution description . . . . .	13
<b>3</b>	<b>Class Diagram and Interface Specification</b>	<b>16</b>
3.1	Class Diagram . . . . .	16
3.2	Data Types and Operations Signature . . . . .	18
3.3	Traceability Matrix . . . . .	30
<b>4</b>	<b>System Architecture and System Design</b>	<b>32</b>
4.1	Architectural Styles . . . . .	32
4.2	Package Diagram . . . . .	34
4.3	Database . . . . .	35
4.4	The thread algorithm of the Server . . . . .	36
<b>5</b>	<b>Hardware Development</b>	<b>38</b>
5.1	ARDUINO . . . . .	39
5.2	PIC Microcontroller . . . . .	41
5.3	ATtiny-85 . . . . .	43
5.4	Part List detail and their role . . . . .	50
5.5	Communication between the hardware . . . . .	56
5.6	Working of Infrared Communication . . . . .	57
5.7	Working of Radio-Frequency Communication . . . . .	57
5.8	Star Network Topology: . . . . .	58
5.9	Multiplexing, Modulation, Algorithms and Protocols . . . . .	59
<b>6</b>	<b>Algorithm</b>	<b>62</b>
6.1	The keywords extraction algorithm . . . . .	62
6.2	The smart control algorithm . . . . .	62
6.3	The reminder algorithm . . . . .	63
<b>7</b>	<b>User interface design and implementation</b>	<b>65</b>
<b>8</b>	<b>Design of tests</b>	<b>69</b>
<b>9</b>	<b>Project Management</b>	<b>74</b>
<b>10</b>	<b>Reference</b>	<b>75</b>

## 2 Interaction Diagrams

### 2.1 UML Diagrams

#### Use Case 1-User Case 4: Controlling and Modification of Elements

#### Use Case 12: Control Information Feedback

User Case 1-User Case 4 allow the users to control and modify the light and music player by their voice messages. These are the main functions of the home automation system. The diagram below shows the general process of the system. There are 4 actors involved in the process: User, Server, Arduino, and Elements. The user selects the control function on the main UI, and UI controller displays the referred control UI to the user. After the user sending the voice message to the server, the server will process the message with the core algorithm and coding the message with the communication protocol then send to the Arduino and the elements. This process involves the feedback information between server and user via cellphone. When the server identifies the message failed, the feedback function will be activated to send the error notification to the user interface.

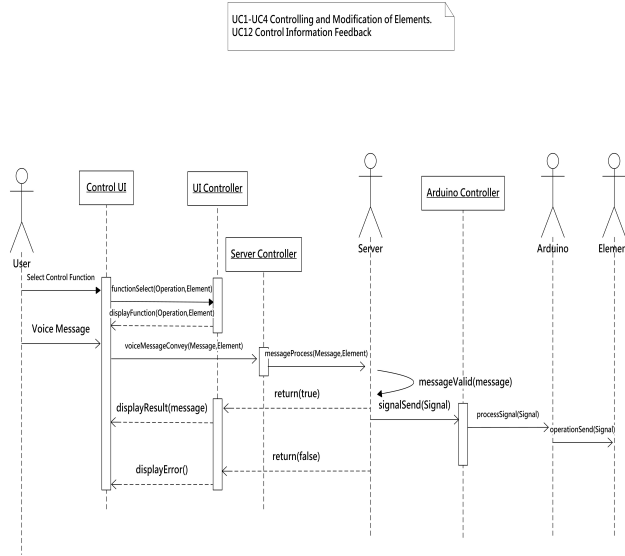


Figure 2: UC1-UC4,UC12

#### Use Case 5: Elements Status Check

Use case 5 allows the user to check the current status of selected element. The purpose of this use case is to monitor the system and make sure all the elements



are working normally to guarantee the security. This process will be achieved by the communication between user interface, server, arduino and elements like below. When the user selects the function, the cellphone will send a request instruction to server. And the server also sends the request instruction to the arduino and elements after coding the message. When the elements receives the request, they will check the status of themselves and send the feedback information to the server. Then the server can read the information and process it and send back to the user interface.

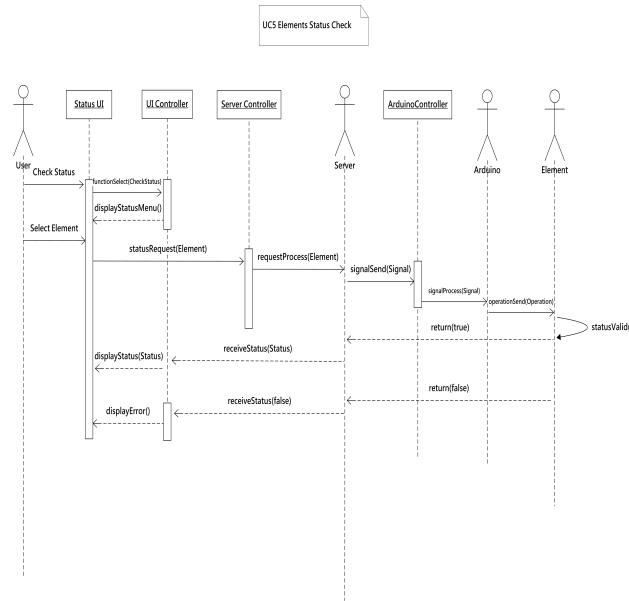


Figure 3: UC-5

## Use Case 6-User Case 11: User Operation and Management

In these series of use cases, they involve several operations for the user and administrator including log out and login action, authority management, create new account, remove account, and check the information of the users. The users can be classified into general users and administrator. There is only one administrator in the system who can modify the authority of users and remove users. Also the administrator has the full-permission of the operating to the elements. The general users have limited authorities to access the elements and database. The database located in the server has played an important role in this series of use cases. All the information of users has been stored in the database and the program should access the database to do the operation of users. The separated diagram of these use cases will be showed below.

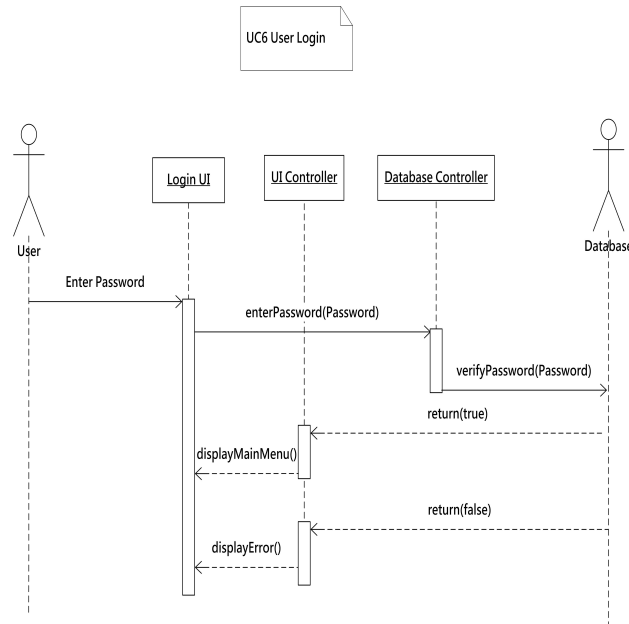


Figure 4: UC-6

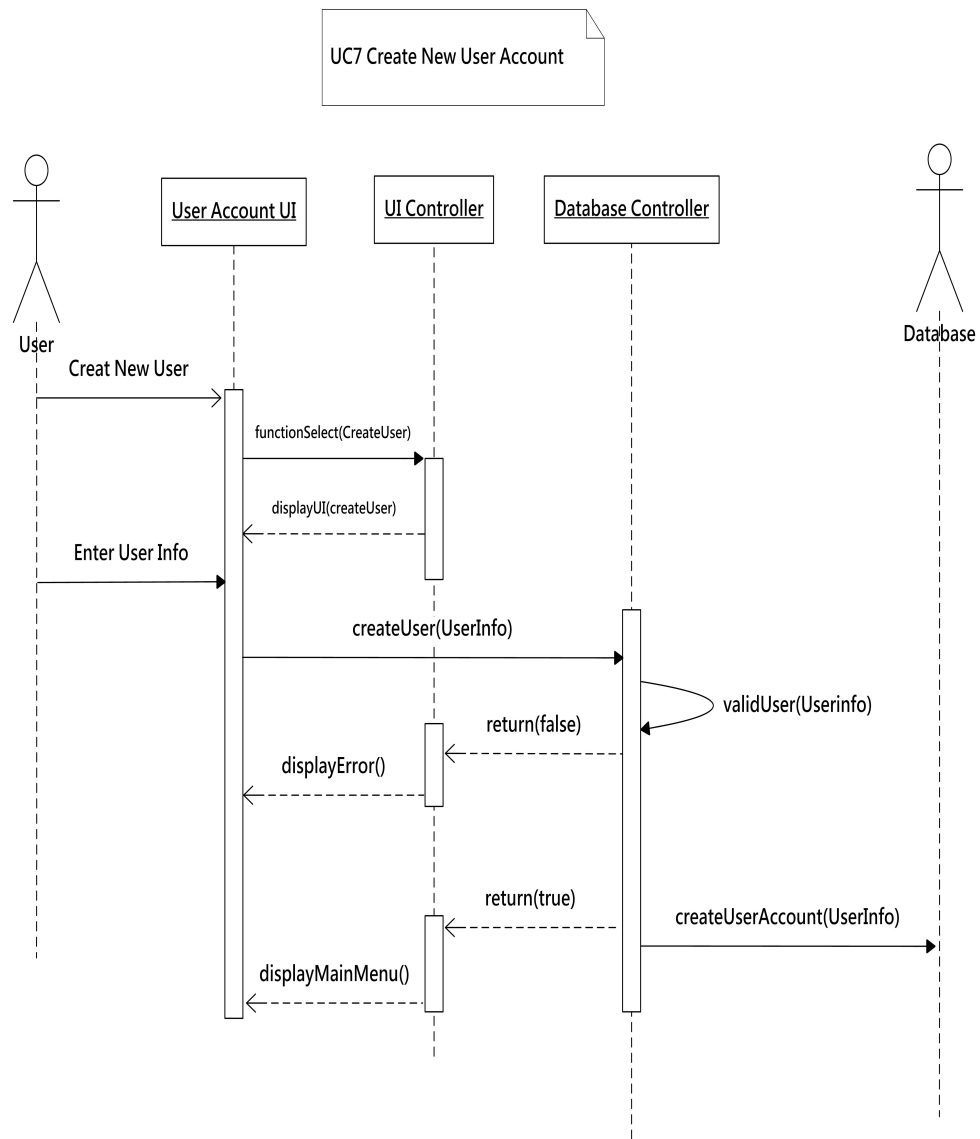


Figure 5: UC-7

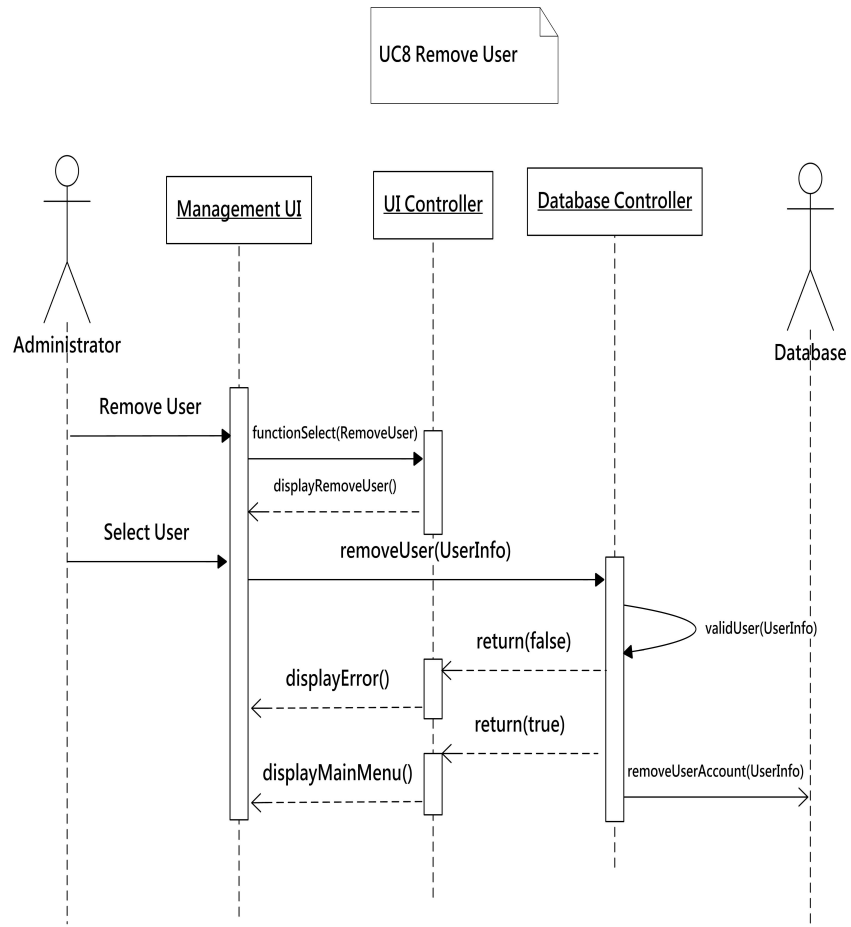


Figure 6: UC-8

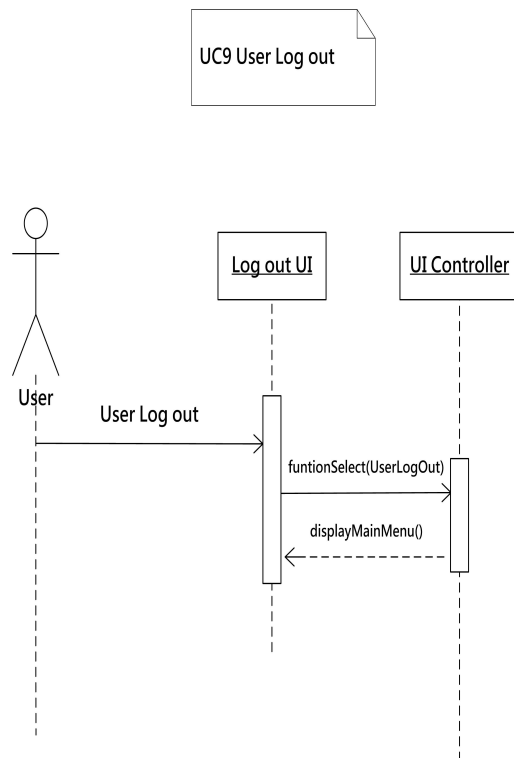


Figure 7: UC-9

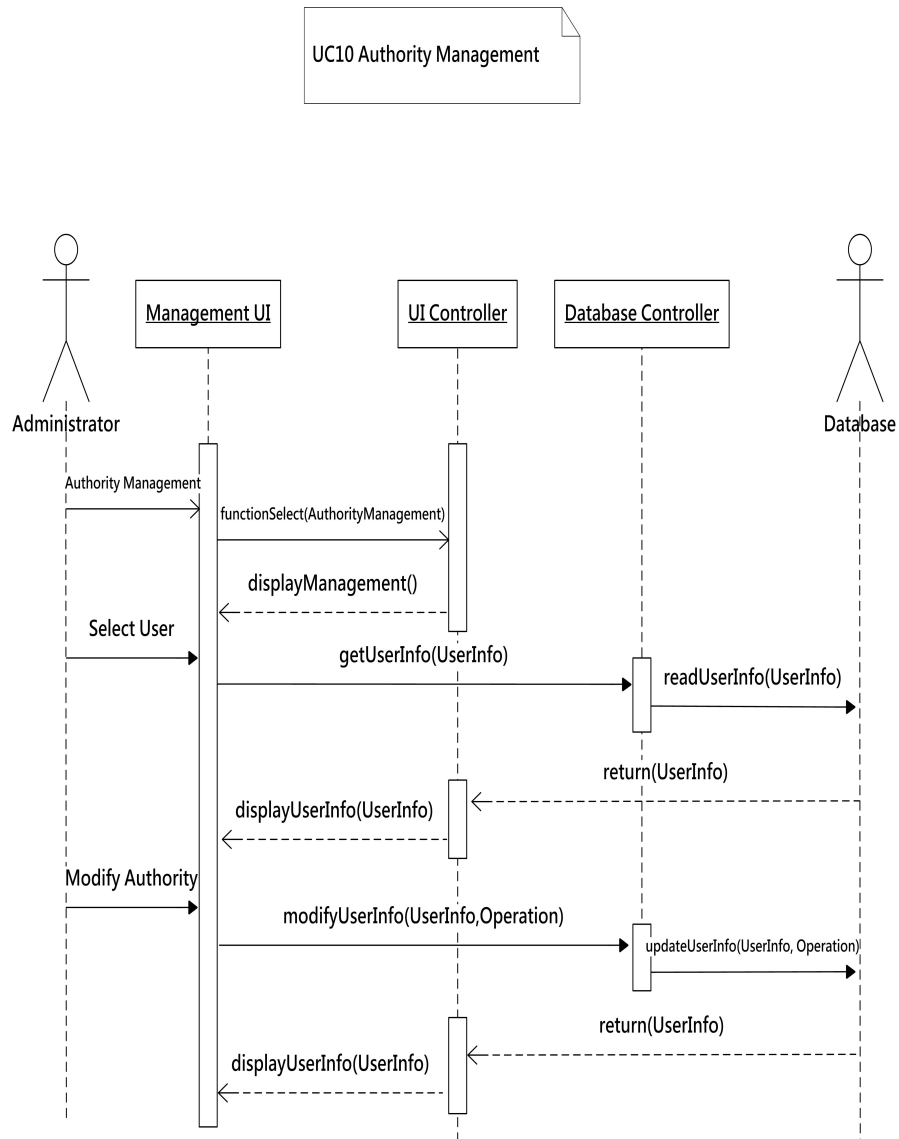


Figure 8: UC-10

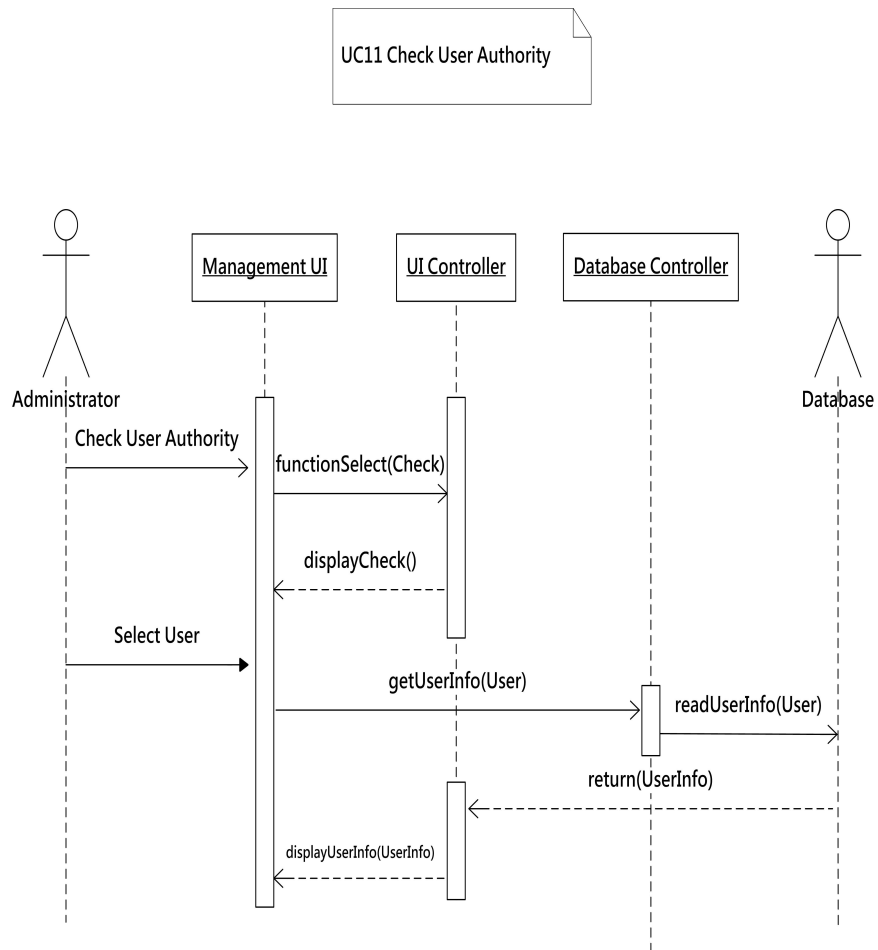


Figure 9: UC-11

### Use Case 13: Network Error Notification:

UC-13 offers a function of checking the network status for users. Because the communication between cellphone and server uses wireless internet, it can be unstable with several external reasons. So it is necessary for users to check the network whether connected. If the network has been interrupted, the system will send a notification to the user interface to remind the users and other operations based on the network will be paused. When doing this function, the cellphone will send a request instruction to the server. And the server can check the connection by some algorithm and send the feedback information to the cellphone. If the connection failed, the error notification will be pushed to the user interface.

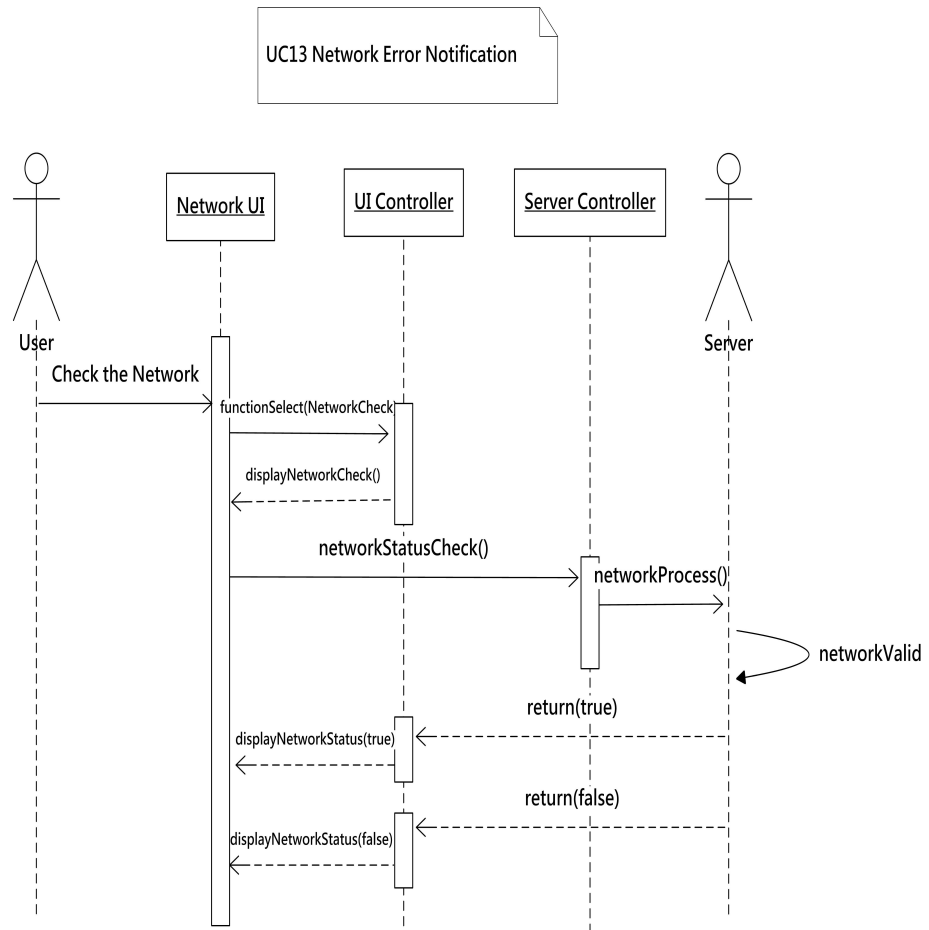


Figure 10: UC-13



## 2.2 Alternative solution description

**c.2.1 User Case 1— *Light Controlling*** The actor goal of this part is to turn on or turn of the light, hence, the basic requests from users is to control the light switch. There may exist a lot of alternative solutions to help us solve this problem, for example, we can implement button to realize these functions (considering the extra demands from customers and In order to avoid voice control problem, we add the button function into our system). However, compared with the voice control, it may cost more time and resource to complete the light controlling by button, we need more than tree buttons to complete just on instruction, such as turn on the light or turn off the light, but for voice, we only need one sentence to solve such problem, customers and users do not need remember and care about button function, couple of words or a sentence enable complete the whole requirements. This method extremely reduces the elements of the external world which may interact with the system-to-be.

**c.2.2 User Case 2— *Music Controlling*** Just as the alternative solution mentioned above, voice control possesses a great amount of advantages than the button, however, in this part, we will explain the reason why we add the buttons into our system. As a part of our system, button also play an important role. At first design, we do not include the button design, because for a voice based system, button seems useless for the implementation. However, for music controlling part, we decide to add buttons, because music may influence our voice recognition and make our system not available. For these reason, in this part, button is a good design for the system compared with other alternative solutions (the only voice control system plays as an alternative solution in this part).

**c.2.3 User Case 3— *Modify the Brightness* / c.2.4 User Case 4— *Modify the Volume of the music*** For these two user cases, there is an alternative solution can be used for both of them. At the original design, we decide to use text message and keyboard to control the brightness and volume. However, compared with the voice control, this method may need too many interactions between the system-to-be and the users; as for the developers, they also need define many actions to realize the users requirements. Voice control, which bases on Andriod api by using the google voice, enable the developers save a lot of time and resource to realize the system and complete the requirements from customers.

**c.2.5 User Case 5— *Elements Status Check*** Elements Status Check is a basic request from the customers, at the original design, we provide several alternative solution to realize this function. For example, adding buttons to enable the users select which status they wish to check (checking the music player status or checking the light status), however, in the end we choose no button in this part, which means when the user logs into the system, they can check the information directly on the screen. Compared with former solution,

this may save customers time and also the resource of the system, at the same time, users enable decide what they will do by checking the status directly.

**c.2.6 User Case 6— User Login / c.2.7 User Case 7— Create New User Accounts** For these two parts, we provide a solution at the beginning of the project, username and password are the basic elements and we just design these two elements in the login system and create new user accounts system at the original program. However, as for a great application in the cellphone, the update and push notification are also the necessary parts. Hence, the username and password are not enough for the systems development and the communication between the users and the developers, for these reason, we add email information request into the new user account. Compared with the original alternative solution, this solution may help the developers and users save time and convenient to both of them.

**c.2.8 User Case 8— Remove Users / c.2.11 User Case 11— Check the User's Authority** Alt. solution: Server delete the all the account information but the administrator. The cost of this alternative solution is cheap; however, it is not practical. The server has no idea which user information should be deserted or is illegal, so it can only delete all the information and leaves the administrator only. Then the login in to the accounts will become inconvenient, for the accounts other than the administrator has all been deleted, which is obviously not our original intention. And it will not work when change the authorities of users.

**c.2.9 User Case 9— User Log Out** Alt. solution: To log out from the system (without special protection to personal information). In this case, the user information would not be deleted, which will possibly result in insecurity. The information of the user will be unsafe. And whats more, the abundant information of users will be stored in local storage or the storage of the server. If the data is stored in local storage, it will be a waste of the room. And it will be more serious if the data is stored in the server, because the user information should first be sent to the server, so along with the problem of storage consuming in the server, the workload of the whole communication will enlarge.

**c.2.10 User Case 10— Authority Management** Alt. solution: Cellphone doesnt send the modified information to the server, instead it changes the authorities of this user directly. In this case, the server will not be able to modify the information in database. As long as the fail of updating in time, the whole system does not have an synchronous database, which will result in later disorder of data. Another result from this is the fail of checking the authority security.

**c.2.12 User Case 12— Control Information Feedback** Alt. solution: Only have the server and the cellphone connected during the feedback query,

resulting in asynchronous of other parts of the system. It is not bad to updating the other parts later when the whole networking is complete, however, the result may turn out to be disturbing to the UC13, which is notify the network error. Because if the information of the whole system could be updated during the network error, then it will no longer be a network error, and thus the network error will be difficult to classify.

*c.2.13 User Case 13— Network Error Notification* Alt. solution: give notification whenever the communication between two parts of the whole system is interrupted. This would not be effective because the links between the parts of the system without the cellphone will not be frequent. And notify their network error will not be useful and it will take too much time.

### 3 Class Diagram and Interface Specification

#### 3.1 Class Diagram

dig.png

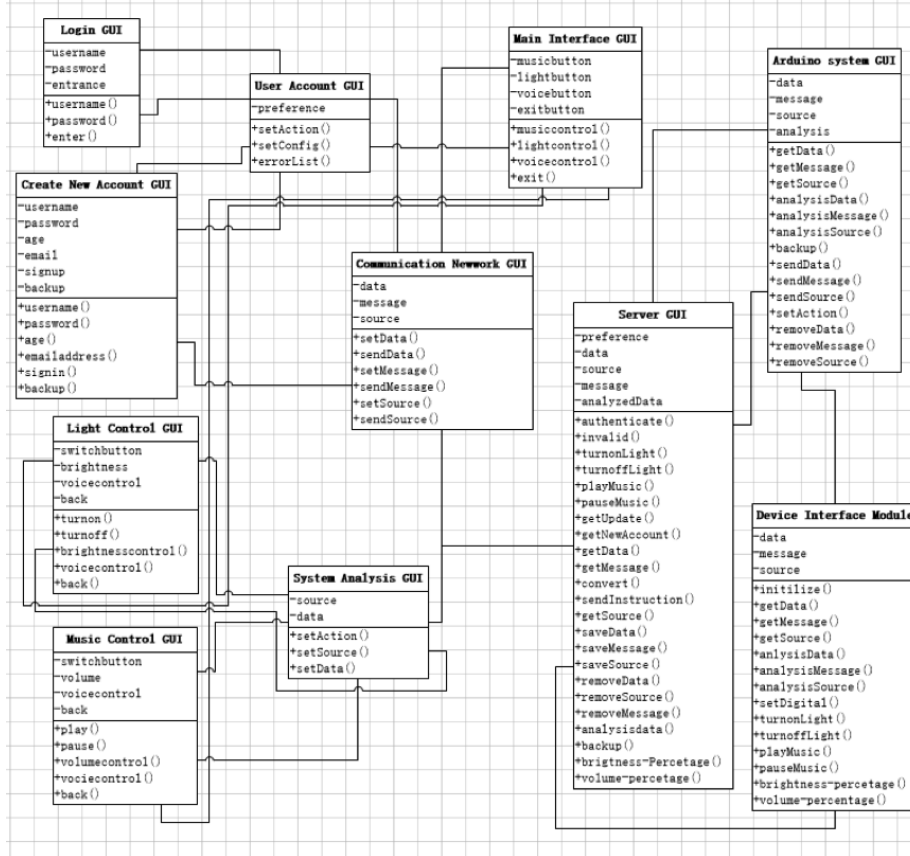


Figure 11: Class Diagram

From figure, we know that there are mainly three parts in the class diagram, the first part we call it UI part, in this part, there also have a lot of small GUIs, such as login, create new account, user account, main interface, light control, music control, and system analysis, the class diagrams of this part is mainly about the UI function. The second part is server part, this part also can be divided into two small parts: server GUI and communication GUI, this part is about the server operations and its data information. The last part is the Arduino, which is designed for the hardware, and Arduino also has two small parts: Ardiuno system GUI and device interface module GUI. Each class in the

figure connect with at least two other classes, and we will show the details for these classes in the following content.

As for the UI part, we can also divide it into two parts to introduce it, the first is the noncontrol part. This part contains login, create new account, user account, and main interface, the function of this part is to help the users to use the system, login GUI enable user type their usernames and passwords, and then user account enable get the information and begin to check the username and password, if it is correct, the user will go to the main interface part, at same time, if the user wish to create new account, the create new account GUI also provide the possible for the users to help them sign up. The main interface GUIs function is to help the user to choose which part they wish to control, it is the center part between noncontrol part and control part. Control part contains three parts: light control, music control and system analysis. The first two GUIs are mainly for the control function, users requirements, such as turn on or turn off the light, play or pause the music, brightness of the light, and volume of the music, can be satisfied by these GUIs, and the system analysis will analyze the source and data which come from the first two parts. The server part is the bridge between the UI and Arduino, the communication network enable the data and resource from UI part to be sent to the server, and the server will analyze all information and then send the corresponding information to the Arduino and the UI part.

The Arduino also can be divide into analysis part and module part, the first part will get the data and message from the server and then analyze and convert it to the instructions for the module, after that the module will execute the corresponding instructions. The system will work well when the whole parts work together, and also we should know that not all of the operations information will be sent to server, some application operations, such as back up and next, do not necessary for the server, only useful data or information will be translated into the server (control of the light and music, create new account). We will show the partial diagrams and more details in the next section.

### 3.2 Data Types and Operations Signature

In this part, we will present all types and signatures for the above data and operations. The class will be divided into several subclass.:

#### 1 .Login GUI

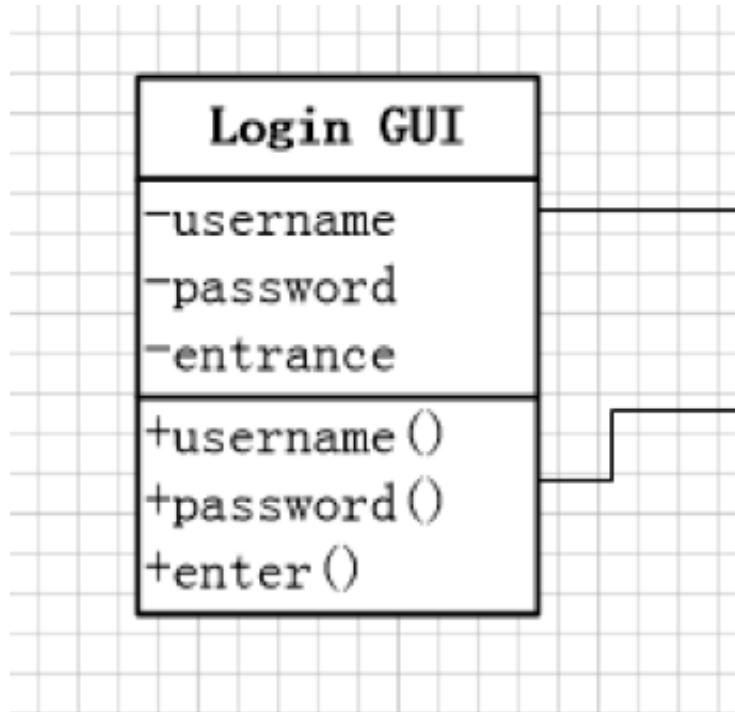


Figure 12: Login GUI

The data types and operation signatures: Username: java:string

Password: java:string

Username():void

Password():void

Enter():void

In this part, the users enable type their usernames and passwords to enter the whole system. When the user type username and password and click the enter button, the login GUI will send the corresponding information to the user account to check the user state. If the username and password correct, user will enter the system; if not, the user will get the error message.

## 2 .Create New Account GUI

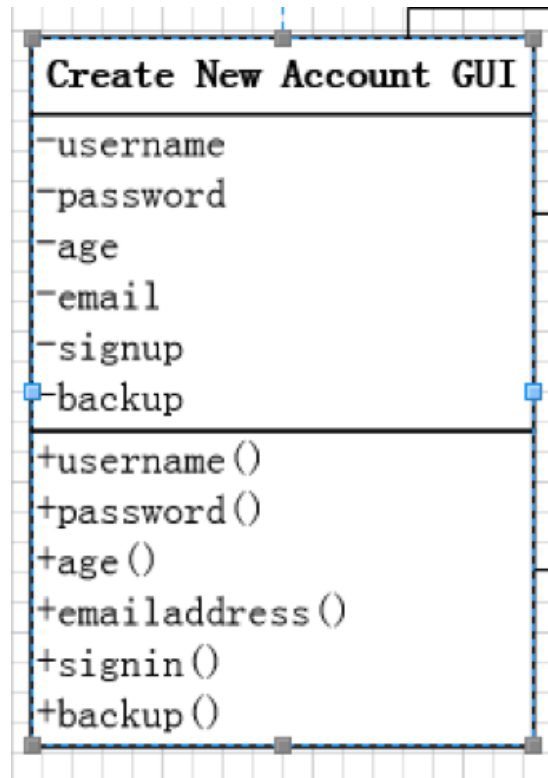


Figure 13: Create New Account GUI

Username: java:string  
Password: java:string  
email: java:string  
Username(): void  
Password():void  
Age():void  
Emailaddress():void  
Signin():void  
Backup():void

As for the create new account GUI, the user enable register new account in this part. The users should type new username, password, age, and email information into the system, and then click signup button to complete the process, and they also enable get back to the login state. When they finish the register, the create new account GUI will send these information to the server by using communication network GUI.

### 3 .User Account GUI

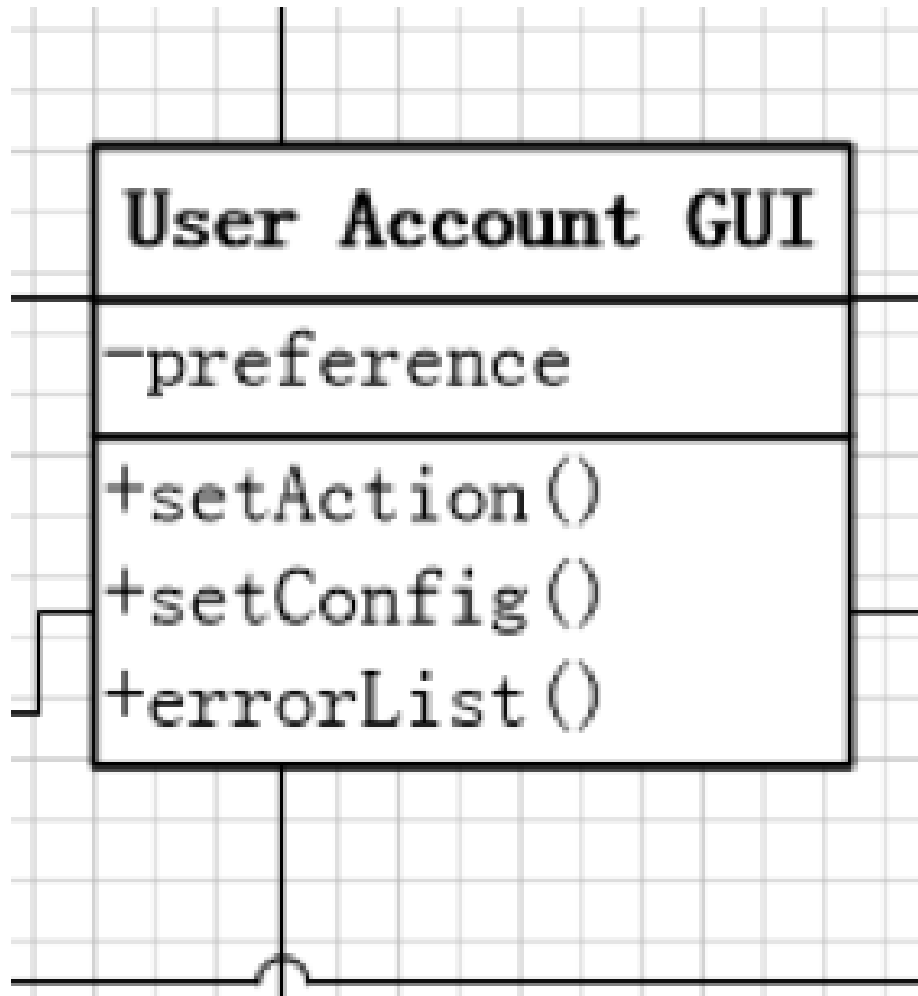


Figure 14: User Account GUI

```
Preference:java:string
setAction():void
setConfig():void
errorList():void
```

When the user creates a new account, the create new account GUI will send the new information to the server to store it, and then back up to the login interface. As for the username and password, the user account GUI will check them, the user will enter the system if they correct or providing error information if not.



#### 4 .Main Interface GUI

The data types and operation signatures:

Musicbutton:java:button

Lightbutton:java:button

voicebutton:java:button

backbutton:java:button

musiccontrol():void

lightcontrol():void

voicecontrol():void

back():void

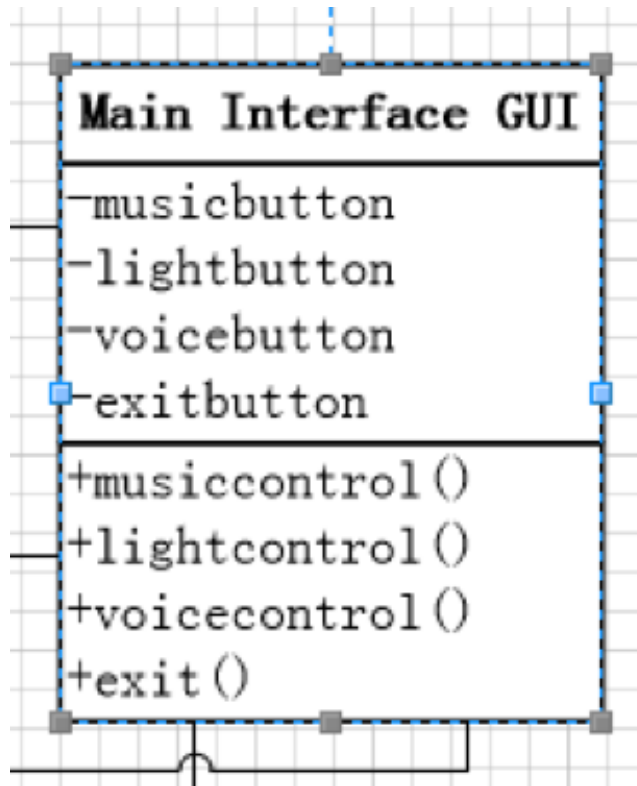


Figure 15: Main Interface GUI

The main interface GUI actually has three main functions, the first and most important part is the voice control. Users can use voice to control the system when they press the voice button. The second part is music button and light button, users enable choose to enter the music control interface or light control interface. The last part is exit part, users can exit the system by this button.

## 5 .Light Control GUI

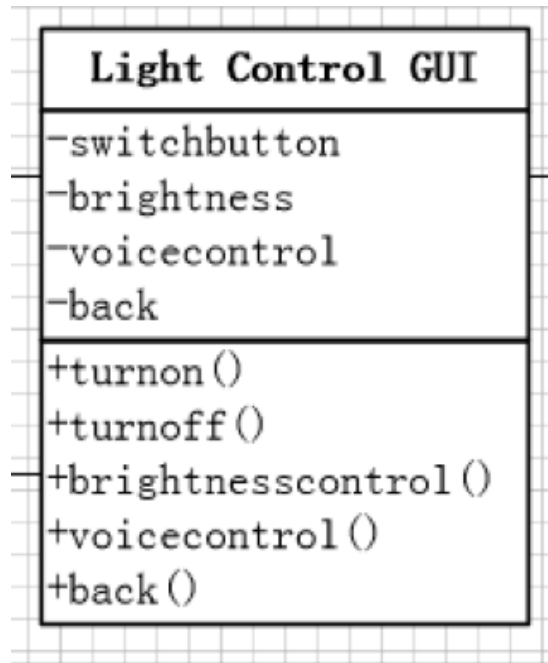


Figure 16: Light Control GUI

The data types and operation signatures:

Switchbutton:java:button

Brightness:java:button

Voicecontrol:java:button

back:java:button

turnon():void

turnoff():void

brightnesscontrol():void

voicecontrol():void

back():void

This part is mainly for the users who prefer button control, and at the same time, we also provide voice control here, the switch button enable turn on or turn off the light and the seek bar enable change the brightness of the light. Besides that back button enable the users to go back the main interface. The information the users required will be sent to the system analysis GUI in the end.

## 6 .Music Control GUI

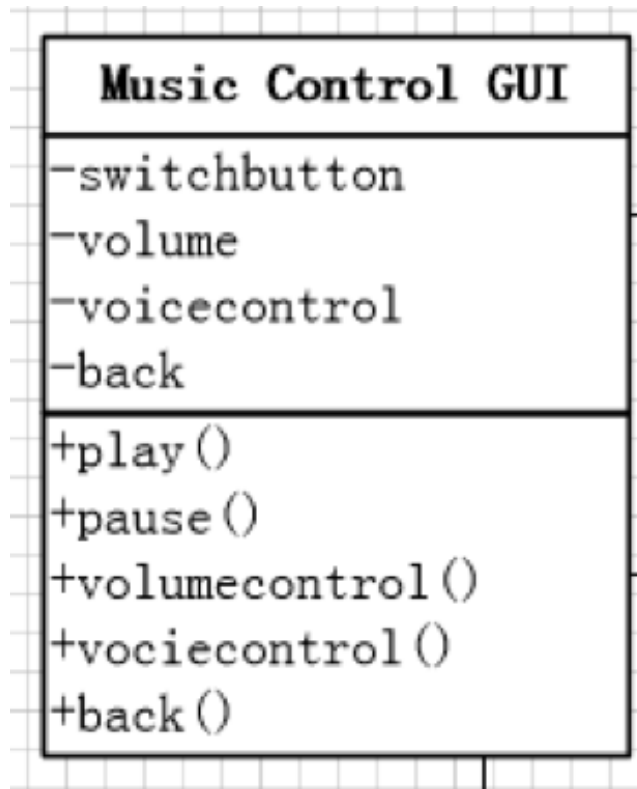


Figure 17: Music Control GUI

Switchbutton:java:button  
volume:java:button  
vociecontrol:java:button  
back:java:button  
play():void  
pause():void  
Volumecontrol:java:button  
Voicecontrol:java:button  
back():void

In this part, user can use voice to complete the action and also we provide button for them, switch button enable lay or pause the music, the volume can be adjusted by the volume button. At the same time, back button can help user go back to the main interface. All the data and information generated by this part will be sent to the system analysis GUI.

## 7 .System Analysis GUI

The data types and operation signatures:

source: java:string

data: java:float

setAction():void

setSource():void

setData():void

The system analysis GUI will analyze all data and source which generated by the music control part and light control part, these information will be set and send to the server part by using the communication network.

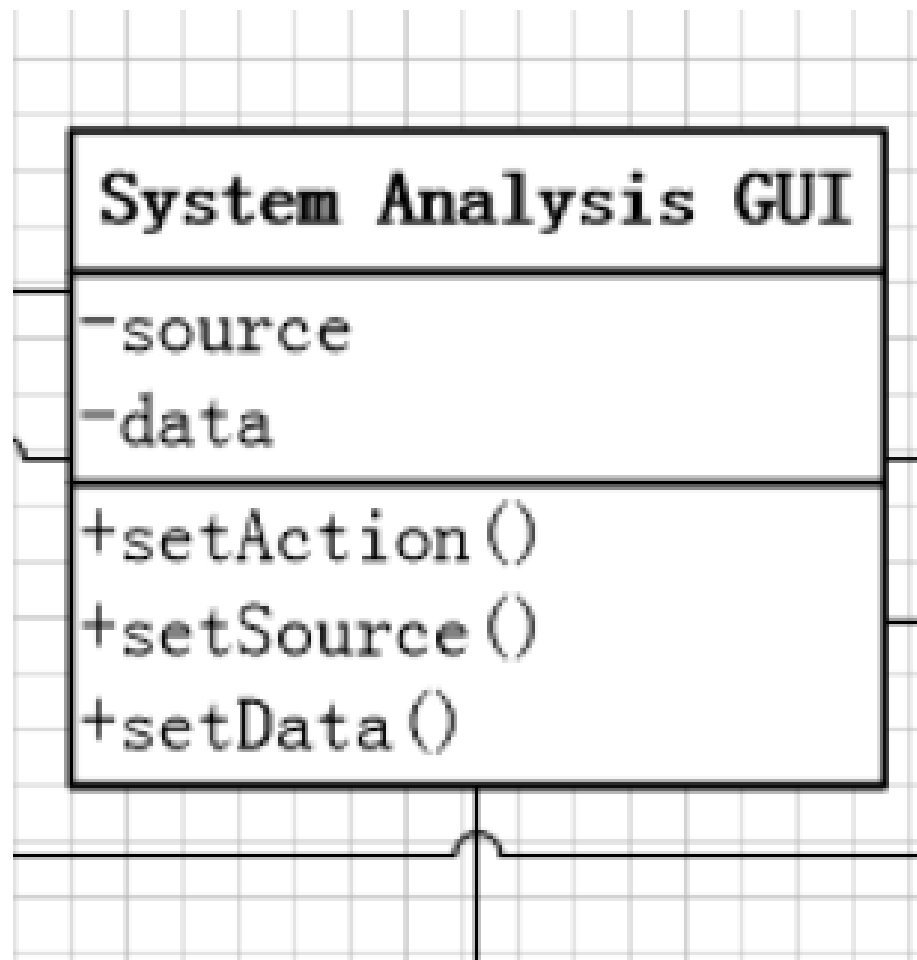


Figure 18: System Analysis GUI

## 8 .Communication Network GUI

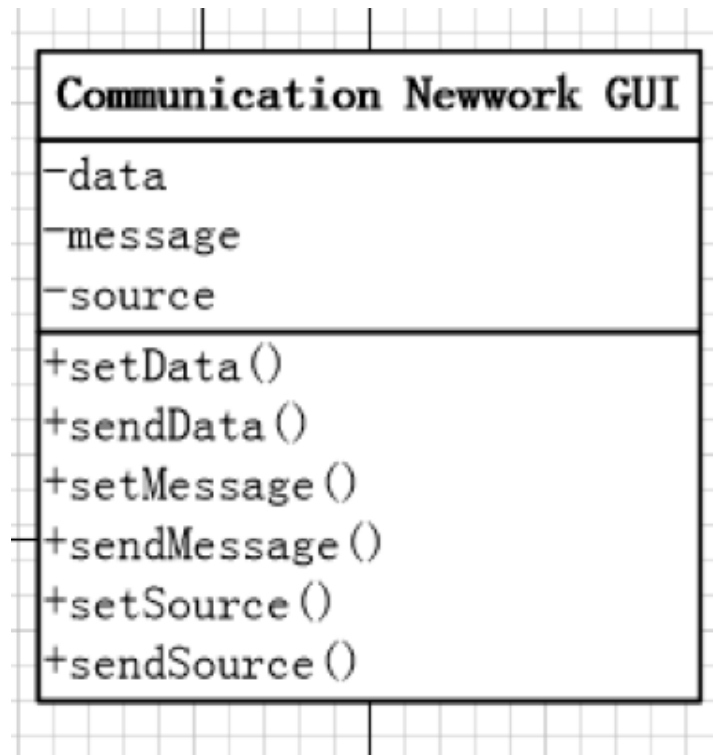


Figure 19: Communication Network GUI

The data types and operation signatures:

source: java:string

data: java:float

message:java:string

setMessage():void

setSource():void

setData():void

sendMessage():void

sendSource():void

sendData():void

Communication network are connected with almost all classes in this system.

The function of this part is very important, data transmission and conversion cannot realize without the communication network, it get message, data, and source from the other parts and then send them into the corresponding parts

## 9 .Server GUI

The data types and operation signatures:

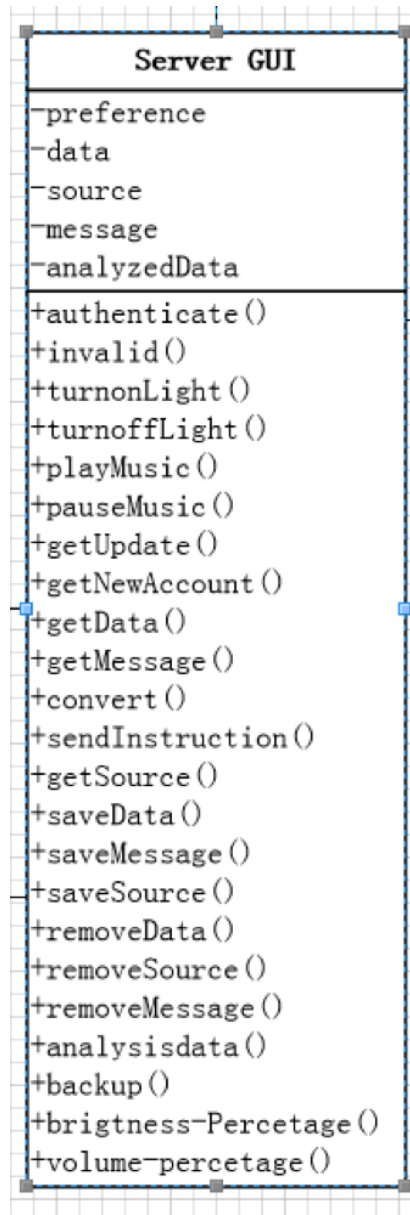


Figure 20: Server GUI

Preference:java:string

Username: java:string  
Password: java:string  
email: java:string  
source: java:string  
data: java:float  
message:java:string  
analyzedData: java:float  
authenticate():bool  
invalid():bool  
turnonLight():void  
turnoffLight():void  
playMusic():void  
pauseMusic():void  
getUpdata():void  
getNewAccount:void  
getData():float  
getMessage():string  
convert():void  
sendInstruction():void  
getSource():string  
saveData():float  
saveMessage():string  
saveSource():string  
removeData():void  
removeSource():void  
removeMessage():void  
analysisdata():void  
backup():void  
brightness-percentage():void  
volume-percentage():void

The server is the bridge between the hardware and the software, the user only use cellphone application and the information generated by the app will send into the server to store these data or message. After analyzing, server will send the analyzed data and message to the Arduino, and at the same time server need to give feedback to the software and hardware.

## 10. Arduino System GUI

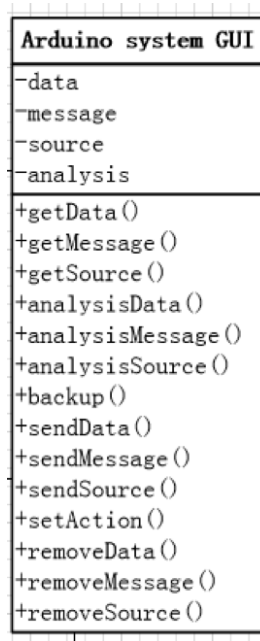


Figure 21: Arduino GUI

```
source: java:string
data: java:float
message:java:string
getData():float
getMessage():string
getSource():string
analyzedMessage: java:string
analyzedSource: java:string
analyzedData: java:float
backup():void
sendData():void
sendMessage():void
sendInsourse():void
removeData():void
removeSource():void
removeMessage():void
```

The Arduino system will get information from server and then it will analyze these information and convert them to the corresponding instruction for the module.



## 11. Device Interface Module

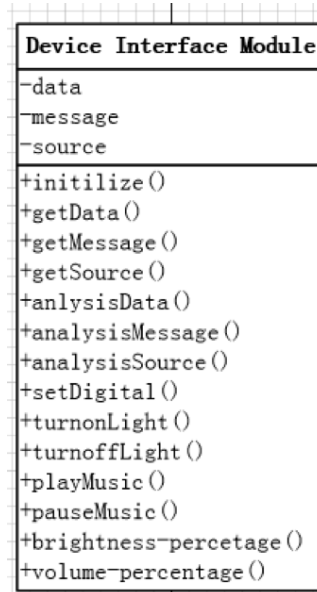


Figure 22: Device Interface Module GUI

```

source: java:string
data: java:float
message:java:string
initialize():void
getData():float
getMessage():string
getSource():string
analyzedMessage: java:string
analyzedSource: java:string
analyzedData: java:float
setDigital():void
turnonLight():void
turnoffLight():void
playMusic():void
pauseMusic():void
brightness-percentage():void
volume-percentage():void
  
```

The module enable receive the instruction from the Arduino system and execute the corresponding execution.

### 3.3 Traceability Matrix

From the figure, we know that the different GUIs will be involved in different interfaces, for example login GUI will be connected with the login interface, new account interface, main interface and server, and it will send or get data or message from these parts. We have already provide the involve information about these part in last section, hence, we will do not explain the relationship of every GUI and interfaces, however, some important information we should learn from the matrix. 1. User account GUI is the key among the UI part, it connect the login, create new account, and main interface. 2. System analysis GUI play an important role in the control part, whether the server can get the right information or not depends on whether the system analysis GUI work properly. 3. Server is the bridge between the user interface and Arduino. Server will get and send data or message from almost all parts and at the same time it should give some feedback to some specific parts. Besides that it also analyzes the data and message and convert them to make them match with the corresponding parts. 4. Communication network is the basic request for the whole project. Data transmission and message convert cannot work properly without the communication network.

login interface	new account interface	light control interface	music control interface	main interface	communication network
X	X			X	
X	X				
X	X			X	X
X		X	X	X	X
		X		X	X
			X	X	X
			X		X
		X	X		X
X	X	X	X		X
X	X	X	X		X
					X
					X

Table 1: traceability matrix

## 4 System Architecture and System Design

### 4.1 Architectural Styles

Our project is too large to be completed by a single person in a short period. To make the development manageable, the best way is to divide it into components that can be developed (and later maintained) separately. Each component will be a work assignment for a team or individual. It is often thought that this decomposition is a management decision, determined primarily by the talent available. The decomposition is a critical design decision to be made on the basis of simple technical criteria. The result is a very unconventional, but easily maintained, design. And through discussion we decided to divide out project into 3 parts.

1. client and server architecture;
2. database architecture;
3. event-driven architecture;

In terms of that clients can make communication with their own house through user interface, we can simply treat our system as client and server architecture. When consider when clients register and they can set up their private account and preference, those information would be stored in database, for next time they log in, the system would call the certain data to control home intelligence which in our project is light and music, thus, we can see our system as data architecture. In our interface especially main interface, clients can see a list of the state of all of their home devices, when user wants to change their states, they can send their command by voice or clicking of the button to control the hardware, it makes our system fall under the event-driven model.

Now, I would introduce those three subsystem way.

**Client and server architecture:** Client/server architecture is an underlying framework that consists of many PCs and workstations as well as smaller number of mainframe machines, connected via local area networks and other type of computer networks. In our project, the destination is to establish the communication between clients and server so that clients can achieve their goal. And this architecture also allowed different clients to control the hardware through authorizing and indentify them.

**Database architecture:** Database architecture is a collection of concepts that can be used to describe the structure of a data, and provide the necessary means to achieve this abstraction. It is composed of models, policies, rules or standards that govern which data is collected, and how it is sorted, arranged, integrated and put to use in data systems and in organizations. Data is usually one of several architecture domains that form the pillars of an enterprise architecture or solution architecture. In our project, database architecture can be divided into lower subsystem - data access subsystem and data store subsystem. Data access get the data from data store and communicate with intelligence through

server. Data store can hold the private information and preference of all the clients.

**Event-driven architecture:** is a pattern promoting the production, detection, consumption of, and reaction to events. From a formal perspective, what is produced, published, propagated, detected or consumed is a message called event notification, and not the event itself, which is the state change that triggered the message emission. For example, in our projects, clients can send their command or click on the button when they want to change the state of the light and music, their voice or the action just like a trigger to control the hardware.

## 4.2 Package Diagram

When modeling a large scale system, we are working with a high volume of model elements. We describe a model from different views and different phases, hence are in different types. UML package helps to organise and arrange model elements and diagrams into logical groups, through which we can manage a chunk of project data together. We can also use packages to present different views of the system's architecture. In addition, we can use package to model the physical package or namespace structure of the application to build. With this, we follow the convention of naming package like using lower case for Java package.

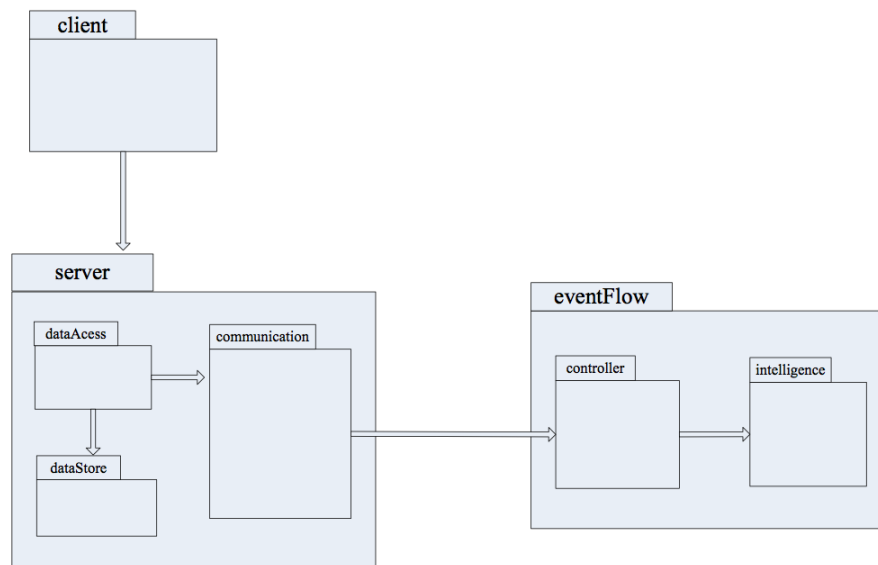


Figure 23: package diagram

### 4.3 Database

**A very simple database** In the user case we mentioned about the login and log out method. Generally, for the security purpose we put the user's account and passwords in the server. When the user request to login or log out, the device will send a message to the server and the server cross compares the username and passwords between the input and the data in the database. So far a very simple database is needed in our system, to keep the usernames and the related passwords for users to login and log out. This database is nothing but one table. We design it as follows:

#ID	Real Name	Account	Password	Address	Mark
1	Li Xinyu	Arthur	123456	812 Benner	0
2	Guo Jiaqi	Guo	123456	Busch	0
3	Nayyar	Vidur	123456	Dogglass	0

Table 2: A simple example of the basic table in the database

**Other requirements for the database** As we all know the Apps for the cellphones or tablets should be as light as possible so that it will keeps the system running smoothly and clean. However the home automation system needs to keep action records and the status of Smart elements. Thus we decide to put the action records into servers.

Also, we mentioned the status of the Smart elements at home should be tracked; we also need to keep records of the elements' status in the server.

Based on the requirements above we design some more tables in the database. They are showing in the table 2 and table 3.

#ID	Account	IP	Time	Element ID	Action
1	Arthur	128.98.12.128	12:40AM Oct.29 2013	L12	Turn on the light
2	Guo	128.98.12.128	12:40AM Oct 29 2013	L13	Turn off the music
3	Vidur	128.98.12.128	12:40AM Oct29 2013	M11	Turn up the light

Table 3: Table of action records

#ID	Account	Element ID	Time	Status	Mark
1	Arthur	L12	12:40AM Oct.29 2013	On 100	0
2	Arthur	L13	12:40AM Oct 29 2013	On 50	0
3	Arthur	M11	12:40AM Oct29 2013	Off	0

Table 4: Table of element status

**Relational Database** As we can see the three databases are related with each other, based on that we can use relational database to organize all the

tables in the database. The logic relationship between different table in the database is shown as follows:

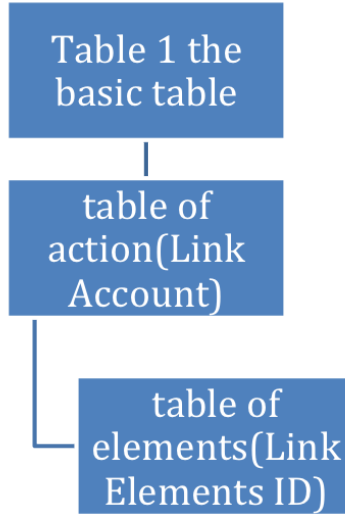


Figure 24: the logic of relational database

#### 4.4 The thread algorithm of the Server

The functional requirements of the server are to process the signals from clients and send commands to the Smart Elements. We have already discussed about those algorithm in previous chapters. In this section we are going to talk about the thread algorithm of the server. The algorithm serves the non-functional requirement that the server should be able to communicate with multiple clients at the same time.

In general, If the server have only one thread and once the server is in communication with a client, the thread is taken. No more clients can build the communication with the server because and the entire client is in a queue. When the communication of the previous client is finished, the thread will be freed and the server can build the communication with another client. Usually the communication is fast and the user can hardly feel the waiting time in the queue. However is the communication is blocked no clients in the queue can build the communication with the server until the current client is removed.

To avoid such problem we use multi-thread method to build the server. The server keeps search for the connection requirement. If there is a requirement, the server will start a new thread to communicate with the client while the main thread is still searching for the requirement.



The logic diagram of the algorithm is as follows:

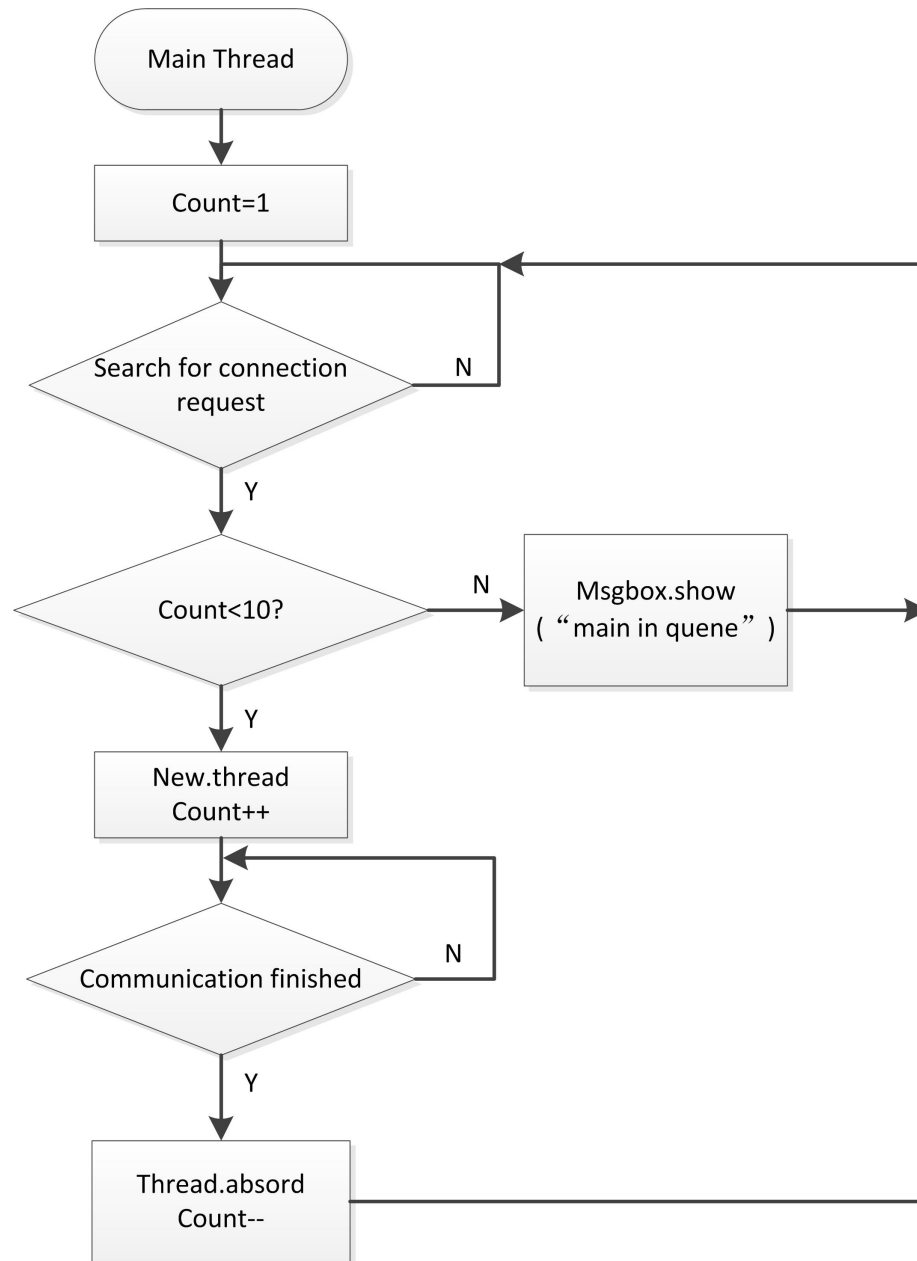


Figure 25: logic diagram of algorithm

## 5 Hardware Development

Hardware and controllers are an essential part of the projects that are expected to bring about some electrical or mechanical changes in its environment. To bring about any change in the physical world, the computer that runs a program can only send out signals in the form of electrical pulses. These pulses are of no use unless they are properly coupled to a well-developed hardware system that can perform the physical changes to realize what the software intended to do.

In the Voice Controlled Home Automation System, users requirements of turning on/off a light or changing the volume of a music player cannot be satisfied until the server is coupled with the hardware to perform the actual actions. For this purpose we interface a number of micro-controllers, relays, sensor, networking techniques and communication equipment with the server.

The hardware implementation is not just about connecting the wires and the components together; it forms the basis of many user case and system requirements. It requires proper and precision designing that is done while keeping in mind not just the users requirements but also the aspects of the system performance ranging from the systems power-performance ratio, cost of implementation, system usability, system reliability and system flexibility. There is a lot of programming involved in developing the hardware to run the controllers efficiently and to maintain the connectivity between it and all the mechanical devices it is in contact with.

- The two main components which form the backbone of the hardware development are the two micro-controller-units (MCU) used:
  1. ARDUINO Mega-series
  2. PIC 18F4520 Micro-controller
  3. ATtiny-85 (Due to their prominent role in this section, it would be worthwhile to read some technical specifications of them.)
- The other main concepts that govern the smooth working of the hardware system and hence the system as a whole are:
  1. Communication mode and equipment
  2. Networking mode

Lets start with discussing the various parts involved.

## 5.1 ARDUINO

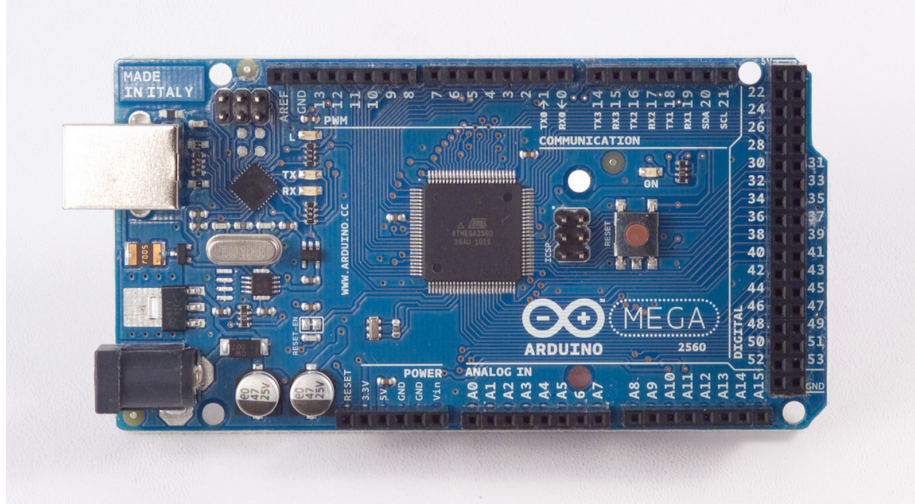


Figure 26: Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, MaxMSP).

**Arduino Mega 2560** The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller. The important feature that is being utilized in this project is that it has four Serial RX and TX Ports. Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage(recommended)	7-12V
Input Voltage(limits)	6-20V
Digital I/o Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8KB
EEPROM	4KB
Clock Speed	16 MHz

### Arduino Mega 2560's Specifications

**Communication** The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega16U2 on the board channels one of these over USB and provides a virtual com port to software on the computer. The Arduino software includes a serial monitor that allows simple textual data to be sent to and from the board.

**Programming** The Arduino Mega can be programmed with the Arduino software. The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming).

## 5.2 PIC Microcontroller

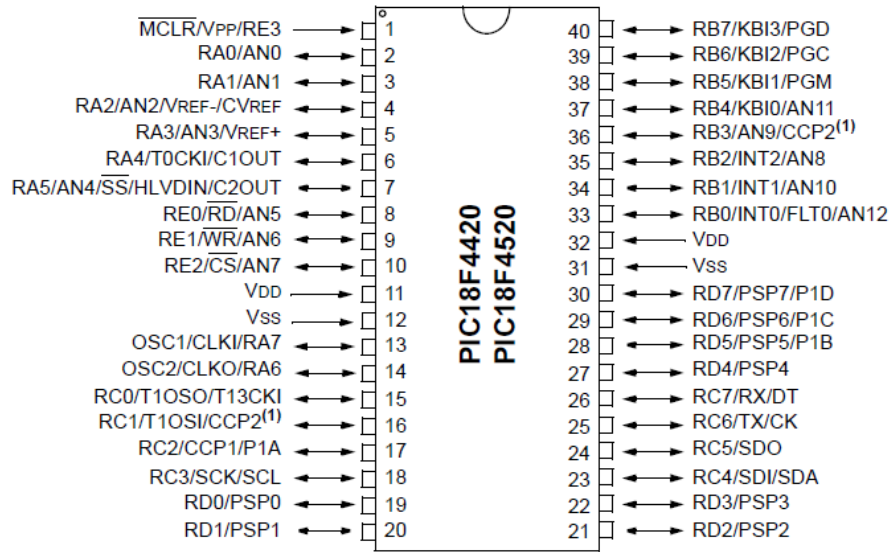


Figure 27: PIC 18F4520

**PIC** is a family of modified Harvard microcontrollers made by Microchip Technology. The name PIC initially referred to as "Peripheral Interface Controller" was changed to "Programmable Intelligent Computer".

PICs are popular with both industrial developers and hobbyists alike due to their low cost, wide availability, large user base, extensive collection of application notes, availability of low cost or free development tools, and serial programming (and re-programming with flash memory) capability.

**PIC 18F Microcontroller Series** The PIC 18F microcontrollers have replaced the PIC16-series microcontrollers that have been around for many years. Microchip Inc. has developed the PIC18 series of microcontrollers for use in high pin count, high density, and complex applications. The PIC 18F microcontrollers offer cost efficient solutions for general-purpose applications written in C that use a real-time operating system (RTOS) and require a complex communication protocol stack such as TCP/IP, CAN, USB, or ZigBee. PIC18F devices provide flash program memory in sizes from 8 to 128 Kbytes and data memory from 256 to 4Kbytes, operating at a range of 2.0 to 5.0 volts, at speeds from DC to 40MHz.

**Performance** The architectural decisions are directed at the maximization of speed-to-cost ratio. The PIC architecture was among the first scalar CPU designs and is still among the simplest and cheapest. The Harvard architecture in which instructions and data come from separate sources simplifies timing and microcircuit design greatly, and this benefits clock speed, price, and power consumption.

#### **Advantages of using PIC 18F microcontroller**

- Small instruction set to learn
- RISC architecture
- Flexibility to program in the widely used C language
- Built in oscillator with selectable speeds
- 5-V operating voltage
- Low power consumption
- Compatibility with arduino
- High reliability
- Inexpensive microcontrollers
- Wide range of interfaces including I2C, SPI, USB, USART, A/D, programmable comparators, PWM, LIN, CAN, PSP, and Ethernet.

**Compiler Development** While several commercial compilers are available, in 2008, Microchip released their own C compilers, C18 and C30, for the line of 18F 24F and 30/33F processors.

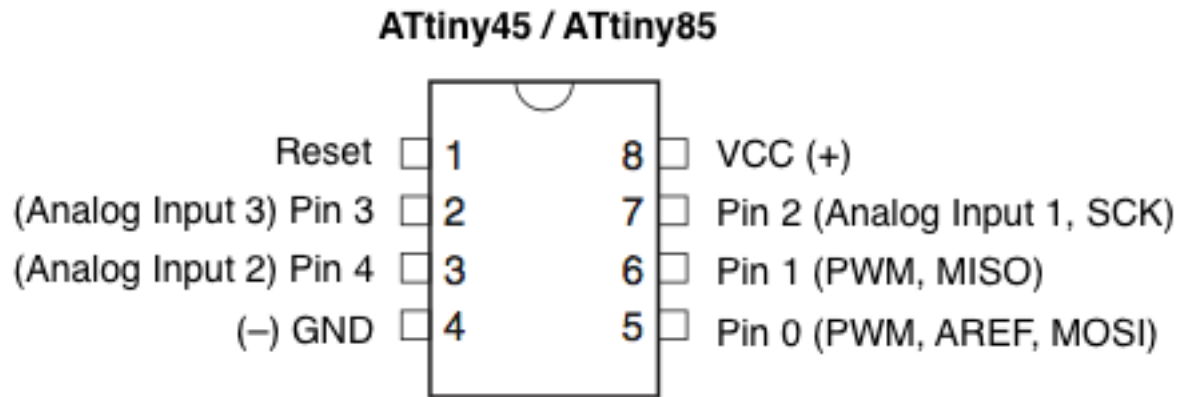
### 5.3 ATtiny-85

The high-performance, low-power Atmel 8-bit AVR RISC-based microcontroller combines 8KB ISP flash memory, 512B EEPROM, 512-Byte SRAM, 6 general purpose I/O lines, 32 general purpose working registers, one 8-bit timer/counter with compare modes, one 8-bit high speed timer/counter, USI, internal and external Interrupts, 4-channel 10-bit A/D converter, programmable watchdog timer with internal oscillator, three software selectable power saving modes, and debugWIRE for on-chip debugging. The device achieves a throughput of 20 MIPS at 20 MHz and operates between 2.7-5.5 volts.

By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

ATTiny-85 is the best choice for this project because of its compatibility with arduino. This chip is used as a replacement for PIC controller while using IR communication.

Parameter	Value
Flash (Kbytes)	8 Kbytes
Pin Count	8
Max. Operating Frequency	20 MHz
CPU	8-bit AVR
# of Touch Channels	3
max I/O Pins	6
Ext Interrupts	6



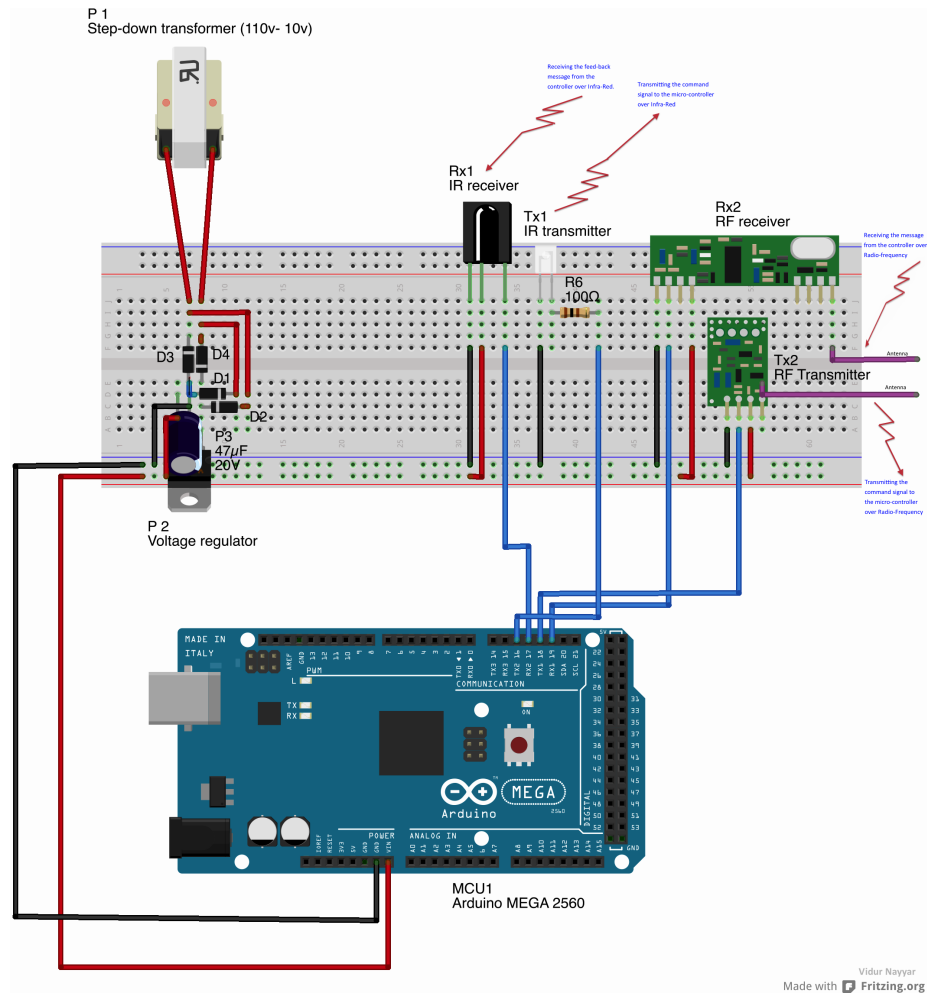


Figure 28: Board Diagram of Arduino as a Tranceiver HUB



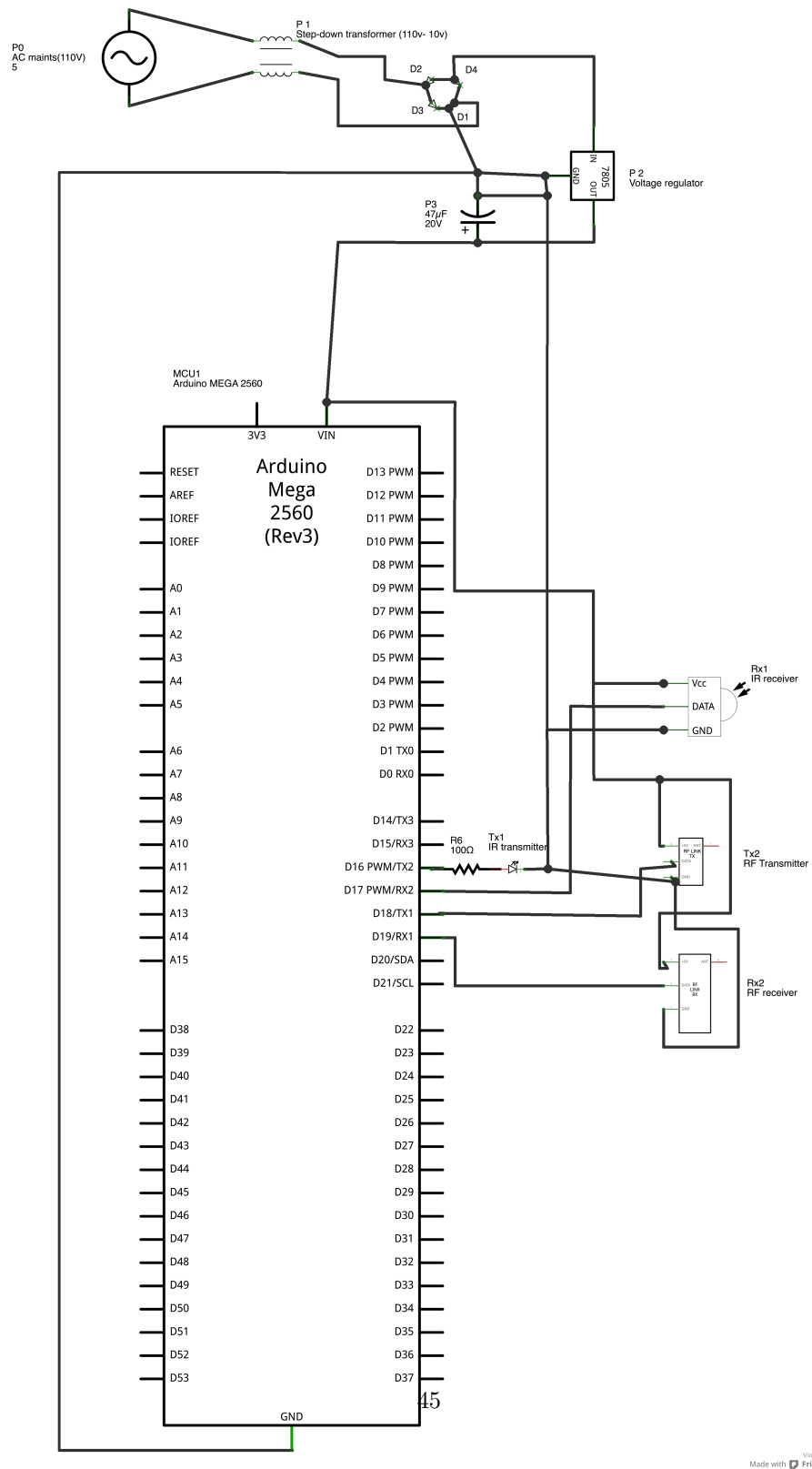


Figure 29: Schematic Diagram of Arduino as a Tranceiver HUB

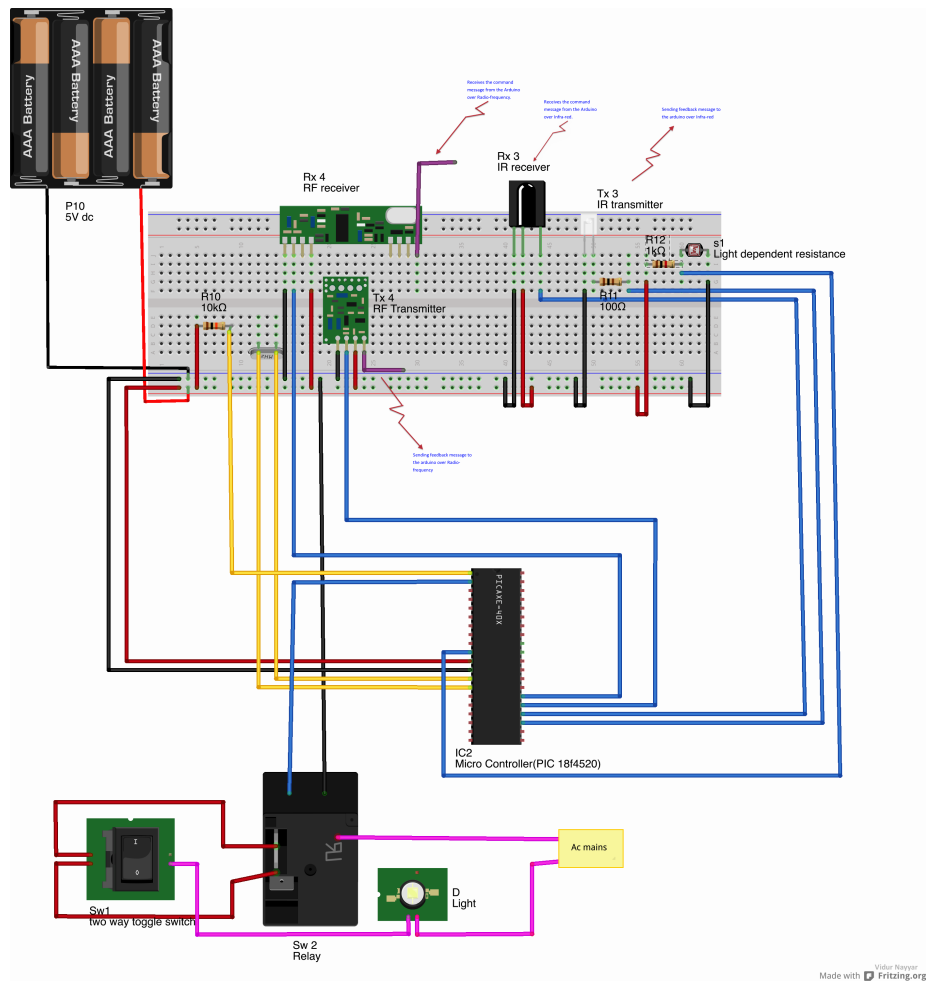


Figure 30: Bread-Board Diagram of PIC18F4520 as a Smart Element

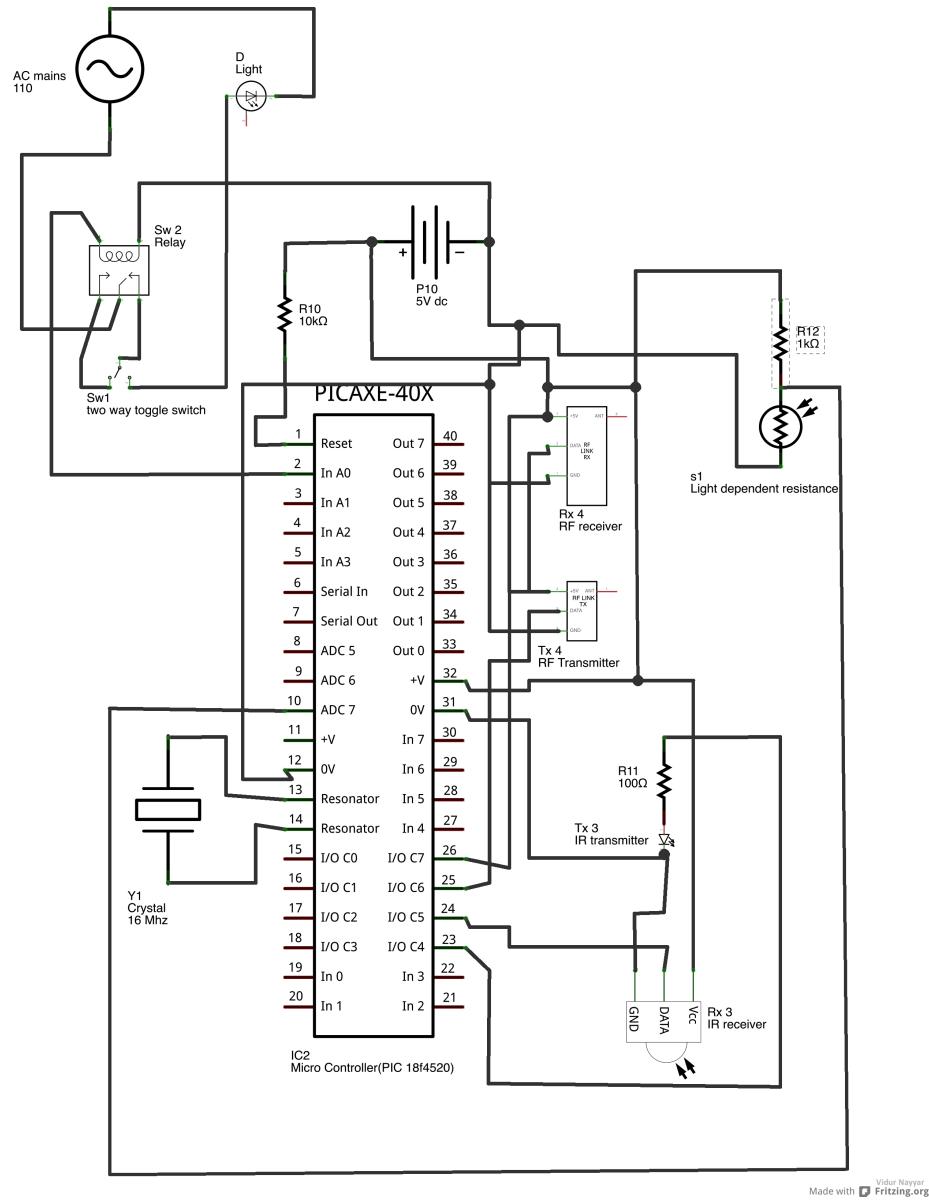


Figure 31: Bread-Board Diagram of PIC18F4520 as a Smart Element



48



## 5.4 Part List detail and their role

**Infrared Transmitter:** This component is a LED (Light Emitting Diode), which emits light in the Infrared domain at 700 nanometers (nm) to 1millimeter(mm), which is below the visible spectrum. This comprises of one of the components that is responsible for sending communication signals between the two controllers (i.e. Arduino and PIC 18F4520 Microcontroller). It comprises of two pins, one is connected to the ground and the other to the output pin of the controller through a resistor. It emits when the output pin of the controller gives out a high pulse. We modulate the signal sent over this transmitter to omit the problem of interference and to utilize the maximum performance of the LED by passing maximum power through it for a small amount of time to avoid it from burning.

In this project an Infrared Emitters High Speed Emitter that operates at 5V, and requires 160mW of power to emit an Infrared signal at 940nm wavelength at 22Degree is used. Emitters emitting at 940nm are preferred because the receiver is most sensitive at this wavelength.

Infrared transmitter is used to make system as cost effective as possible, and because of the low cost of Infrared components, we can hook them to many devices to automate those devices. This system has been designed in a way to revolutionize home automation by providing maximum usability at a low cost. Another benefit of using Infrared communication is that the power consumption is minimum that is an important concern in case of the microcontroller attached to the devices (like door lock) that are powered by batteries.

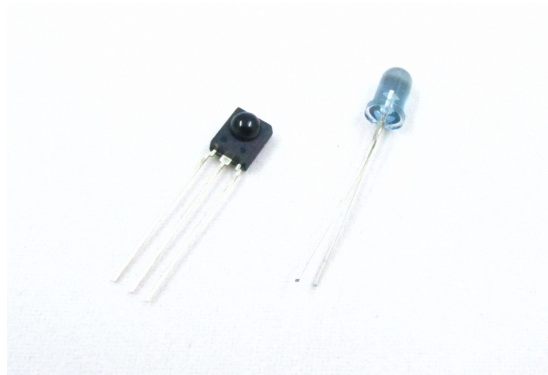


Figure 34: IR receiver and IR transmitter

**IR-Phototransistor:** This device is used as an infrared receiver to capture the transmitted signal, demodulate it, and passes it to the microcontroller. It acts like a normal transistor switching on and off depending upon the intensity of IR light being shined at it. As seen in figure X, The received IR signal is picked up by the IR detection diode on the left side of the diagram. This signal is amplified and limited by the first 2 stages. The limiter acts as an AGC circuit to get a constant pulse level, regardless of the distance to the handset. As you can see only the AC signal is sent to the Band Pass Filter. The Band Pass Filter is tuned to the modulation frequency of the handset unit. Common frequencies range from 30kHz to 60kHz in consumer electronics. The next stages are a detector, integrator and comparator. The purpose of these three blocks is to detect the presence of the modulation frequency. If this modulation frequency is present the output of the comparator will be pulled low, which can be detected by the controller to which it is attached.

For this application, we use an IR receiver that works for carrier frequency of 38KHz and can detect from a distance of 30 meters at a 45 degree view angle. Its operating voltage is 2.7 -5.5 volts and output current is 5 mA which can be easily coupled to the input pins of the controllers.

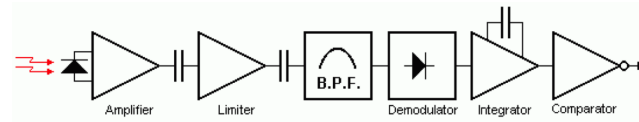
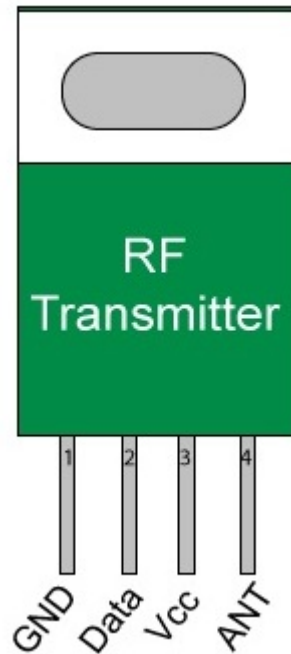


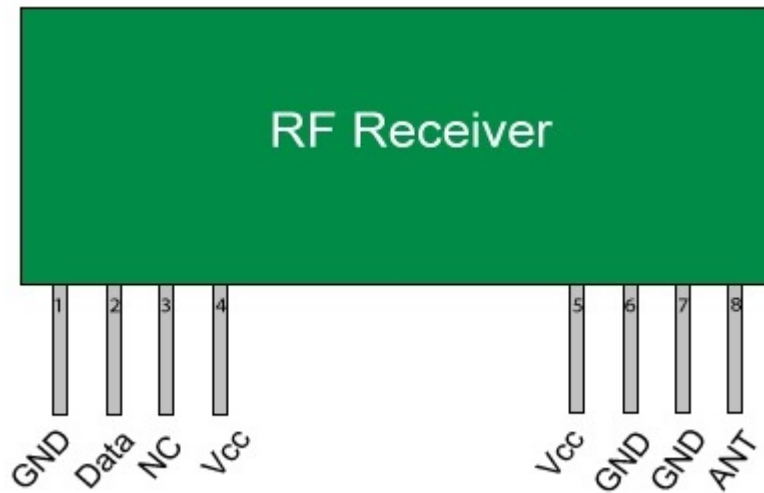
Figure 35: IR receiver(internal view)

**RF 434MHz transmitter Module:** The RF module, as the name suggests, operates at Radio Frequency. The corresponding frequency range varies between 30 kHz & 300 GHz. In this RF system, the digital data is represented as variations in the amplitude of carrier wave. This kind of modulation is known as Amplitude Shift Keying (ASK). It is a small 4 pin module that requires only power, ground and data. It is small enough to fit into any project and add wireless communication. These modules have up to 500 ft range in open space. The transmitter operates from 2-12V. Higher the Voltage, greater is the range of the transmitter. This is an upgrade from IR communication, as we dont need line of sight communication in this and the range is greater. An RF transmitter receives serial data and transmits it wirelessly through RF through its antenna connected at pin4. The transmission occurs at the rate of 1Kbps - 10Kbps.



**RF 434MHz receiver Module:** The rf interface receiver module is larger for demodulation and amplification circuitry. It also has a simple array of power, ground and data pins to connect up with the PIC's USART. The transmitted data is received by an RF receiver operating at the same frequency as that of the transmitter. These wireless receivers work with our 434MHz transmitters. They can easily fit into a breadboard and work well with microcontrollers to create a very simple wireless data link. Since these are only receivers, they will only work communicating data one-way, we would need two pairs (of different frequencies) to act as a transmitter/receiver pair. These modules are indiscriminate and will receive a fair amount of noise. Both the transmitter and receiver work at common frequencies and don't have IDs. Therefore, a method of filtering this noise and pairing transmitter and receiver will be necessary.





**Relay and Manual-switch in a two-way switch configuration:** Relay is an electro-magnetic device that acts like a switch. The microcontroller can switch On/Off any device by controlling the relay. A two-way toggle switch is used as a manual switch. In short, relay is the switch whose state can be changed by the controller (by the voice command of the user) and the manual switch is the switch whose state can be changed physically by the user. The systems requirement was to have a system in which the user had the option to choose between switching On/Off the light either manually or by using the voice command. To realize this requirement, the hardware was designed in such a parallel and self-adapting way that will allow the use of a device manually and by using voice control at the same time without inducing any problem in the accessibility. The device is made to be self-adaptive by making the controller sense the users manual input and thereby alter its switching method as per the system requirement (ST-16).

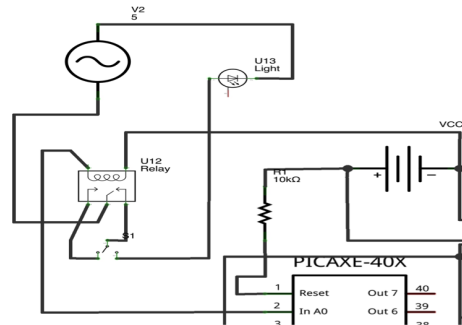


Figure 36: Smart switch

**LDR (Light Dependent Resistanc):** This device is used as a sensor. The special feature of this sensor is that the resistance of this sensor is inversely proportional to the light falling on it. We will use this device to get a real-time feedback of the luminance of the light or to determine if the light is on/off. If the light is ON, the LDR will offer minimum resistance and the voltage drop across the pin of the microcontroller would be maximum. In the case when the light is OFF, the voltage drop across the pin of the microcontroller would be minimum, this way the microcontroller can sense if the light is ON/Off. This not only gives the user a real-time status of the element (light) but also informs the user in case the light fails to respond in the desired manner. It realizes the following system requirements.(ST1,ST6,ST-10 ST-17)

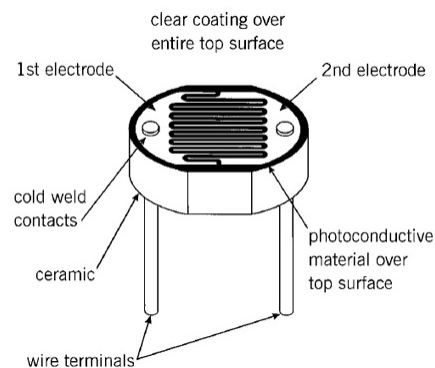


Figure 37: Light Dependent Resistor

**Electric Microphone:** is an acoustic-to-electric transducers or sensors that converts sound into an electrical signal. Higher the loudness of the sound, greater is the electric signal produced by the microphone. This property of the

microphone is used in the project to get a real-time feedback of the music player. By using the mic, the controller will be updated in real-time, with the volume of the music player. This information can be used to send as an feedback to the user.

The microphone is attached to the analog pin of the ATtiny-85 and gives a varied analog input corresponding to the ambient sounds. It realizes the following system requirements. (ST1,ST6,ST-10 ST-17)

**Push Button:** The push button is used as a mechanical sensor. This famous sensor is a button that sends a signal when pressed and does not send when not pressed. This device can be used to get a real-time feedback if the door is locked or not. This feature was not present in the original system requirements, but adding more us abilities to it show that the system is flexible and accepts future expansions and alterations in the system.

**RF434MHz transmitter Module:** A small 4 pin module that requires only power, ground and data. Device that we will use in the Radio frequency transmitter to send the transmitted signal passed to it by the microcontroller. It is small enough to fit into any project and add wireless communication.

**Arduino Wi-Fi Sheild:** The Arduino Wi-Fi Shield connects the Arduino to the Internet using 802.11 wireless specification (Wi-Fi). With this the arduino can be given commands over the Internet and the developer can have an access to it remotely.

Using this device helps us achieve the System requirement (18, 19) Supportibility is realized as in case of an issue, the technician can remotely solve the problem and keep a log of it in his database. This can also be helpful in increasing the security of the system, in case a thief or malicious user interferes with the system, the system can inform the authorities in charge.

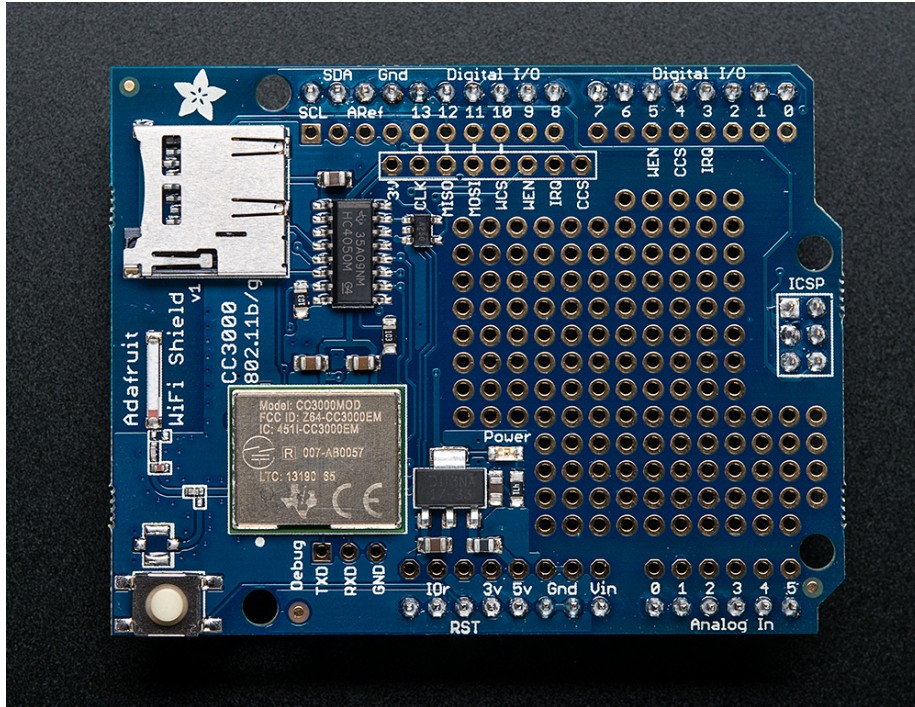


Figure 38: Arduino WIFI shield

**16 MHz Crystal:** This is the external clock of the PIC micro-controller. The controller can transmit data over this frequency. The reason for choosing a 16 MHz crystal oscillator is that arduino operates at 16MHz, for compatibility purpose we needed a microcontroller to operate at the same frequency.

## 5.5 Communication between the hardware

The communication between electronic devices is all about pulses of energy. Serial Communication involves changing the voltage of an electrical connection between the sender and receiver at a specific rate. Each interval of time represents one bit of information. The sender changes the voltage to send a value of 0 or 1 for the bit in question, and the receiver reads whether the voltage is high or low. There are two methods that the sender and receiver can use to agree on the rate at which bits are sent. In asynchronous serial communication, the rate is agreed upon mutually and clocked independently by sender and receiver. In synchronous serial communication, the sender pulses a separate connection high and low at a steady rate, controls it.

There are two common types of wireless communication in most peoples lives: infrared light communication and radio communication. The main difference

between them, from a user or developers position, is their directionality. There are three types of devices common to both IR and RF systems: transmitters, which send a signal but cant receive one; receivers, which receive a signal but cant send one; and transceivers, which can do both. In this project we use a transmitter and receiver pair than a transceiver to save ourselves from the complexity of interference in transceivers. In the Voice based home automation system the transmitter keeps on sending the message till the time its receiver receives a feedback that the message has been sent. So we can say that the controller intelligently switches between the transmitter and receiver.

## 5.6 Working of Infrared Communication

IR communication works by pulsing an IR LED at a set data rate, and receiving the pulses using an IR photodiode. Its simply serial communication transmitted using infrared light. Since there are many everyday sources of IR light (the sun, incandescent light bulbs, any heat source), its necessary to differentiate the IR data signal from other IR energy. To do this, the serial output is sent to an oscillator before its sent to the output LED. The wave created by the oscillator, called a carrier wave, is a regular pulse thats modulated by the pulses of the data signal.

The receiver picks up all IR light but filters out anything thats not vibrating at the carrier frequency. Then it filters out the carrier frequency, so all thats left is the data signal. This method transmits data using infrared light without getting interference from other IR light sources.

The directional nature of infrared makes it more limited, but we use it in our project because it is cheaper than radio, and requires less power. As radios get cheaper, more power-efficient, and more robust, its less common to see an IR port on a computer. However, its still both cost-effective and power-efficient for line-of- sight remote control applications. In Home automation, power consumption parameter is important because the smart elements that are placed remotely operate on batteries.

## 5.7 Working of Radio-Frequency Communication

Radio relies on the electrical property called induction. Any time you vary the electrical current in a wire, you generate a corresponding magnetic field that emanates from the wire. This changing magnetic field induces an electrical current in any other wires in the field. The frequency of the magnetic field is the same as the frequency of the current in the original wire. This means that if you want to send a signal without a wire, you can generate a changing current in one wire at a given frequency, and attach a circuit to the second wire to detect current changes at that frequency. Thats how radio works.

The distance that you can transmit a radio signal depends on the signal strength, the sensitivity of the receiver, the nature of the antennas, and any obstacles that block the signal. The stronger the original current and the more sensitive the receiver, the farther apart the sender and receiver can be. The two wires act

as antennas. Any conductor can be an antenna. The length and shape of the antenna and the frequency of the signal all affect transmission. for a straight wire antenna is as follows: Antenna length = 5,616 in. / frequency in MHz = 14,266.06 cm. / frequency in MHz.

In Voice controlled home automation we use digital RF COMMUNICATION. Digital radios superimpose digital signals on the carrier wave, so there must be a digital device on either end to encode or decode those signals. In other words, digital radios are basically modems, converting digital data to radio signals, and radio signals back into digital data. Though the power consumption in RF communication is more than in IR communication, but the increased range and omnidirectional property of Radio waves makes it more desirable.

## 5.8 Star Network Topology:

The Star Network Topology is used to connect the server to each of the smart elements controlled by the micro-controllers. The arduino is used as a central HUB that acts as a conduit to transmit messages. This consists of a central node, to which all other nodes (smart elements) are connected; this central node provides a common connection point for all nodes through a hub (arduino). In star topology, every node (Smart element) is connected to a central node called a hub (arduino). The switch is the server and the peripherals are the clients.

The only alteration made to the star topology is that arduino is an active hub and directs the feedback from each node (smart element) to the server instead of broadcasting it. The arduino only broadcasts the message from the server and directs the message towards all the nodes.

The star network topology is adopted in Voice control home automation for its advantages which are:

1. Better performance: star topology prevents the passing of data packets through an excessive number of nodes. At most, 3 devices and 2 links are involved in any communication between any two devices. Although this topology places a huge overhead on the central hub, with adequate capacity, the hub can handle very high utilization by one device without affecting others.
2. Isolation of devices: Each device is inherently isolated by the link that connects it to the hub. This makes the isolation of individual devices straightforward and amounts to disconnecting each device from the others. This isolation also prevents any non-centralized failure from affecting the network.
3. Benefits from centralization: As the central hub is the bottleneck, increasing its capacity, or connecting additional devices to it, increases the size of the network very easily. Centralization also allows the inspection of traffic through the network. This facilitates analysis of the traffic and detection of suspicious behavior.
4. Easy to detect faults and to remove parts.
5. No disruptions to the network when connecting or removing devices.
6. Installation and configuration is easy since every one device only requires a link and one input/output port to connect it to any other device(s).

## 5.9 Multiplexing, Modulation, Algorithms and Protocols

While transmitting via Radio Frequency or Infrared, anyone with a compatible receiver can receive your signal (in line of sight for IR communication). There's no wire to contain the signal, so if two transmitters are sending at the same time, they will interfere with each other. This is the biggest weakness of radio frequency and Infrared communication: a given receiver has no way to know who sent the signal it's receiving.

The only way to avoid such interference is to (modulate the pulse in case of infrared communication) multiplex the signal in case of Radio frequency communication. Sharing in radio communication is called multiplexing, and this form of sharing is called time division multiplexing. Each transmitter gets a given time slot in which to transmit. Multiplexing helps transmission by arranging for transmitters to take turns and to distinguish them based on frequency, but it doesn't concern itself with the content of what's being said. To make sure the message is clear, it's common to use a data protocol on top of using multiplexing. A number of protocols have been used to make the home automation system secure and error free.

**Handshake protocol:** A protocol similar to the handshaking protocol is adopted for the communication between arduino and the other controllers attached to make the elements of the house, smart elements. As per the protocol that we have incorporated, when the user sends a command message to the arduino. The arduino relays the message to the controller either via RF or IR mode of communication, and keeps on sending the message till the controller replies with a feedback message. Once the feedback is received by the arduino, it needs to send an acknowledge message to the controller. The controller keeps on sending the feedback message till the time the arduino replies the feedback with an acknowledge message. This way the communication is reliable and in case of an error, the problem can be isolated.

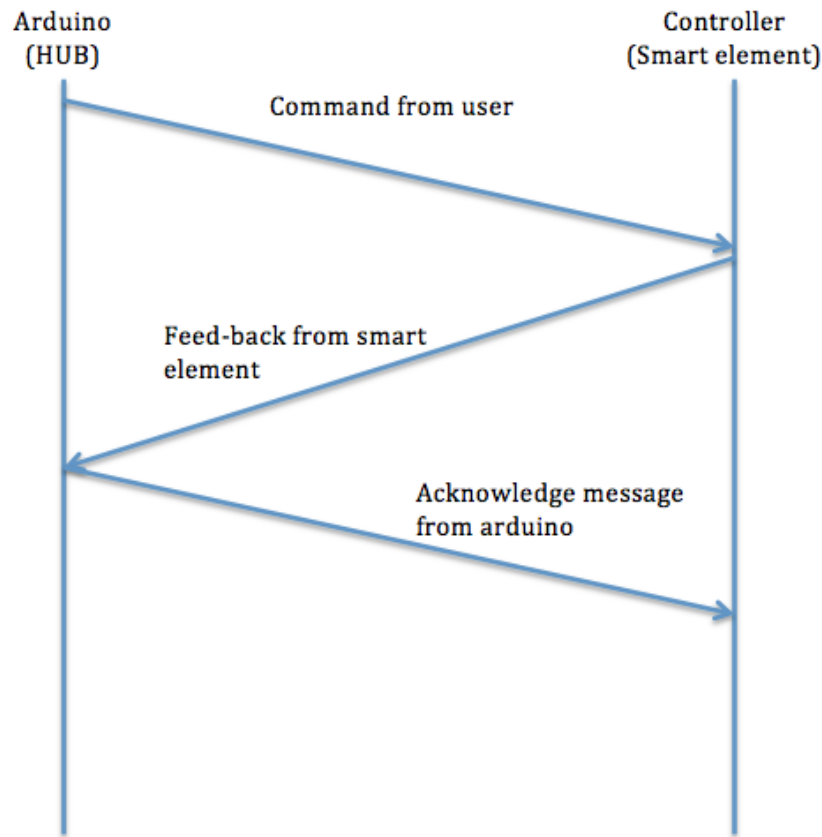


Figure 39: Hand-shake protocol

**Checksum algorithm:** The message bits communicated over Infrared and Radio frequency have to go through interference and noise, which may mix some noise or error bits in the original message. To detect such error bits and reject the message with such errors, we incorporate an algorithm as a part of our protocol. According to this protocol, every time a message is received by the receiving circuit of the controller. The controller must check the message using the checksum algorithm and only proceed towards decoding the received message if the checksum algorithm is satisfied.

According to this algorithm, the message bits are divided into 3 parts, header, data and tail. The header contains the address information of the receptor, the data contains the information of the action that needs to be performed and the tail contains some bits used to check if the message is correct or if it has some errors in it. What the controller does to implement this algorithm is that it computes the sum of the timing of all the high bits in the message and compares



t to the sum of timings of all the high bits in the tail. For the message to be considered error free and decoded, the sum of timing of high bits in the tail and the sum of timing of high bits in the tail should be equal. If they are not equal, it means it has error and the message is discarded and receptor waits for the next message.

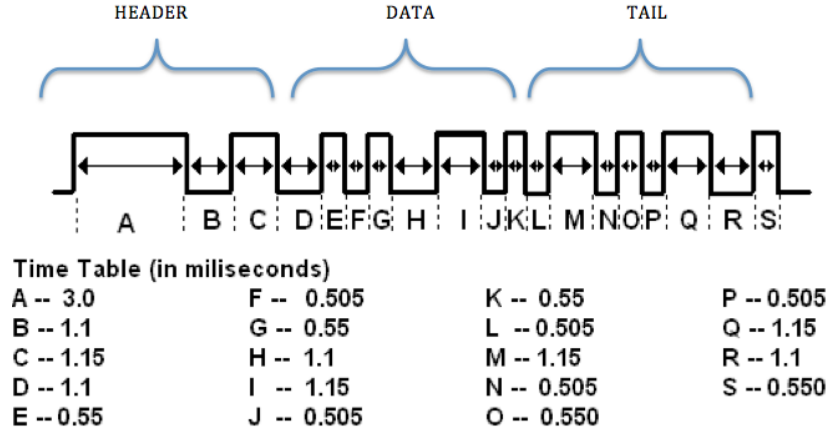


Figure 40: checksum algorithm

**Pulse-window algorithm:** During the communication test, it was seen that the noise signal was either too long a pulse or too short. So we defined another algorithm to be a part of the communication protocol. According to the Pulse-window algorithm, a higher limit and a lower limit of the accepted pulse is predefined. This way the receiver does not waste time in decoding or even checking the message for error in case it is not in the predefined range. This increases the efficiency of the receiver and saves it from overburden and errors. A high-low pair in the message is defined as a pulse. In our project we accept the message with pulse length from 4 to 12.

**Fuzziness algorithm:** In Infrared and Radio frequency communication there are always some signal errors and timing errors due to interference and timing latency of the controllers. To overcome this situation, and to make our system a little tolerant to errors, we introduce the Fuzziness algorithm. According to this algorithm if the received message has some difference in the pulse high time or pulse low time of any pulse in comparison to the saved pulse timings, it is acceptable till the time it is within the range of a predefined percentage of fuzziness.

Pulse is accepted if the Absolute value of (pulse time - saved pulse time) is less than (fuzziness/100) x pulse time

## 6 Algorithm

In this section we are going to talk about an algorithm we have used and some perspective algorithms.

### 6.1 The keywords extraction algorithm

As we know a command sentence usually consists of different kind of words. Some of the words in a command are important to the server while other words are not necessary. We use a word filter to extract some of the keywords from the command.

After analyzing the command sentences, we find that usually typical command contains three kinds of keywords, target, action and degree. Usually we use nouns to represent the target, we use verbs for action and we use adjectives or percentages for degrees.

Based on above we have three word pools, for target, action and degrees. Once a new word is input, the server segments every word and compare those words with words in the pool. If the words are same or share the similar meaning the server is able to extract the keywords.

The diagram is as follows:



Figure 41: The algorithm of keywords extraction

### 6.2 The smart control algorithm

Sometimes understanding the commands needs not only the keywords, the priori knowledge is equally important. If the command is simply "turn on the light" which light should be turned on by the system? We know if it is dinnertime the light in the dinner room should be turned on, if it is bedtime the bedroom's light should be turned on.

Based on the huge number of records of user's command history the system is able to conclude the user's habits and based on the user's habits the system can do the right thing based on some very simple command sentences.

As it is mentioned before, the Server is linked to an Access database, which keeps all the action records. Based on the machine learning method we can try to build the algorithm based on user's habit.

We use three most important factors as the key feature in the algorithm: time,

place and date. The main purpose of the algorithm is to sort the user's command and based on the command records to conclude some patterns. For example, based on the user's command records of in the living room we get the following table.

Among all 300 cases		
Actions	Time	Sequence
Turn on the light	6:00am-6:00pm	0
Turn on the light	6:00am-8:00pm	270
... ..		

Figure 42: Table 1 the statistic result of user's habit based on place

From the table above we can see that there is a high possibility that the user will turn on the light in the living room between 6:00pm to 9:00 pm. So if the user's command is turn on the light and the time is between 6:00pm to 9:00pm the system will turn on the light in the living room based on the records.

### 6.3 The reminder algorithm

Based on the user's habit, we are not only able to make the system understand the simple commands in different time, we can even make the server able to predict user's command.

There are some fixed schedule in our daily life like we turn on the light in our bedroom at the bedtime and we turn off the music when we leave home. We are able to calculate the percentage of some specific event. If the percentage of a certain event in a certain time is more than 90% we call it the high possibility event. The system will do these things automatically everyday. Also the list of high possibility will be updated every day to make sure there will be less statistics mistakes.

The classification of the event is in the following chat:

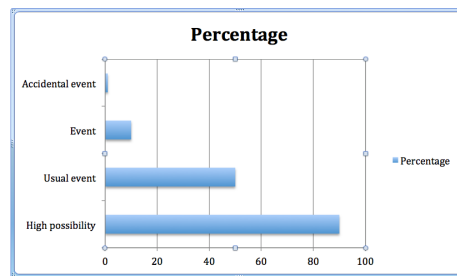


Figure 3 the classification of events

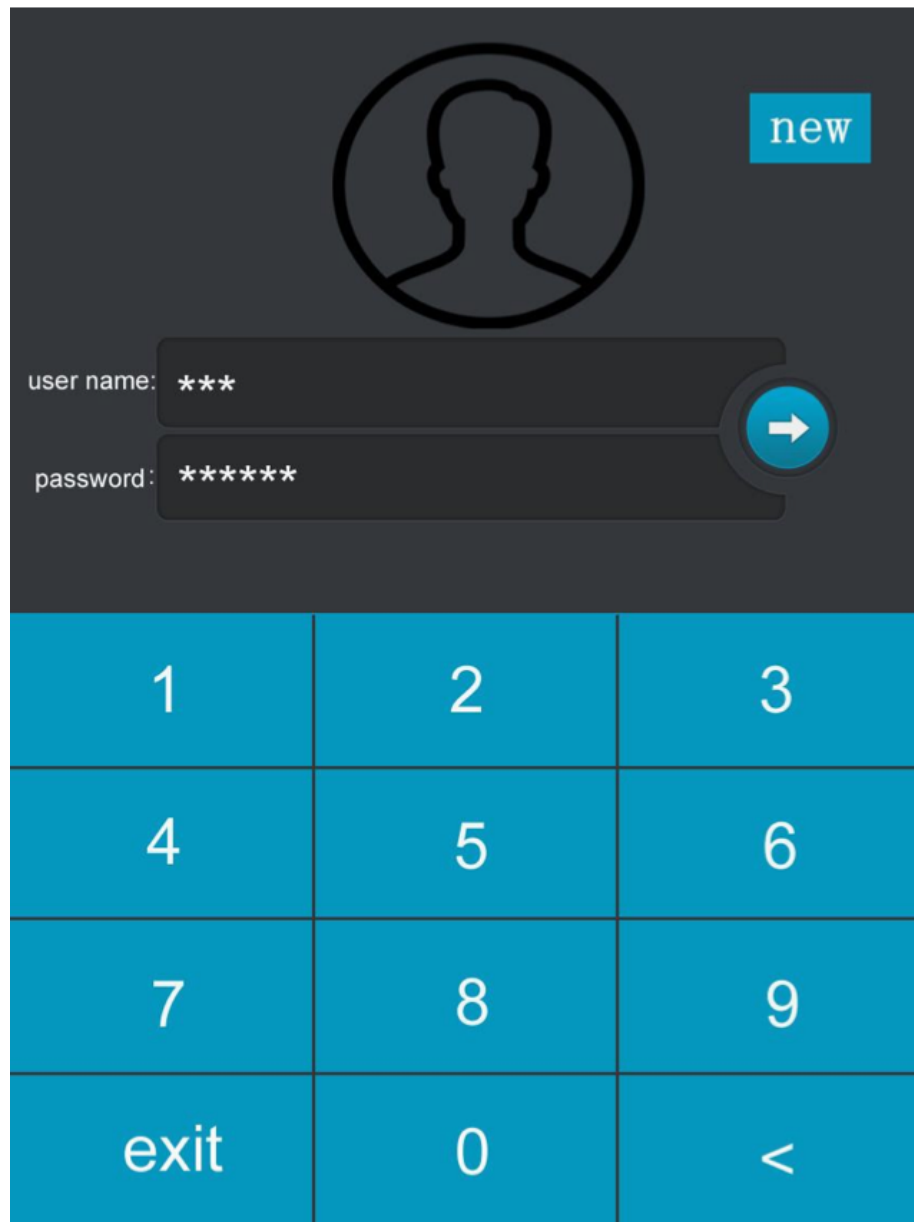
Figure 43: Table 1 the statistic result of user's habit based on place

## 7 User interface design and implementation

In report 1, we have made a lot of effort on our interface design, it can generally cooperate with server to realise the function of our product. However, we still found some problems existing in previous interface while preparing the demo1. Thus, with the help of suggestions of others, we try to modify it and to do some slight changes to make our interface more friendly to user. The following is the brief description and images of the primary changes to our interface (What I do not mention here still remained the same)

### a. Add register interface

In our previous design, we have owned login interface through which users can be endowed with authority. When consider that our project allows different users to work on it, it is necessary to create additional screen to satisfy this function. So we establish a new register interface. here comes our arrangement, when people using our home automation, the first interface is still the login page, in this page, we make a slight change that add a new button on it. If the user has not registered yet, they can click on the "create new user" button and come to the register interface. It contains a couple of text box, when users come to this interface, they can click the empty text box to input their basic information as they are required. The basic information includes name, age, password and their email. After enter the correct information, they can click on the "create" button to get authority. At the same time, the system would bring them back to the login interface.



The image shows a user registration interface. At the top, there is a dark grey header area. On the left, there is a circular icon of a person's head and shoulders. To the right of this icon is a blue button with the word "new" in white. Below the header, there are two input fields. The first is labeled "user name:" and contains three asterisks "\*\*\*". The second is labeled "password:" and contains seven asterisks "\*\*\*\*\*". To the right of these fields is a blue circular button with a white right-pointing arrow. Below the input fields is a 4x3 grid of blue buttons. The buttons are labeled as follows: Row 1: 1, 2, 3; Row 2: 4, 5, 6; Row 3: 7, 8, 9; Row 4: exit, 0, <.

1	2	3
4	5	6
7	8	9
exit	0	<


Figure 44: Add register interface

b. Improve brightness adjusting button

Consider that adjust brightness with "+" button and "-" buttons is inconvenience to users, we decide to exchange those two buttons into a drag strip which

can dramatically decrease user effort. For example, if the user wants to adjust the lightness fro 10% to 90%, they have to do 8 clicks with old interface, with new interface, the user just needs to drag it at anywhere they want instead of clicking a button too many times. What's more, the drag sttrip can provide users with good visual presentation about the brightness. We believe those changes can improve te operability and avaiability of our interface without increasing its complexity. The current interface would make our product seems more reliable and with higher quality.

Figure 45: Improve brightness adjusting button



name:

email:

age:

password:

Figure 46: New user signup

## 8 Design of tests

### a. Casual Test Cases

**Use Case 1:** Light Controlling

User turn on or off the lights by sending the voice message to the server from the Android based cellphone.

**Use Case 2:** Modify the lightness

User control the music player by their voice. And the server sends the control instruction to the MCU after processing the voice message.

**Use Case 3:** Light Controlling

The users can modify the lightness of the light by their voice command.

**Use Case 4:** Modify the volume of music

The user can turn up/down the music by sending the related voice message to the server from the Android based cellphone.

**Use Case 6:** User log in

A user logs in to his/her account in order to configure the alarm system. The tests confirm that the correct passwords and usernames allow the logins to the correct account.

**Use Case 8:** Remove user

User can remove the deserted user and their information.

**Use Case 12:** Control information feedback

The users can check the feedback of voice message in order to check whether the voice have been identified by the server. If the message identification failed, the feedback information can lead the user to send again or cancel the operation.

**Use Case 13:** Network error notification

The users can get the notification of network error from the user interface of Android cellphone if the signal of network is weak. The signal the users get is consist of 2 parts: Between cellphone and server; between server and MCU.

### b. Descriptive Test Cases

**Use Case 1:** Light Controlling

Success

- The light turns on when user press the light controlling button.



- The light turns on when user send a voice message containing the key words of turn on & light.

#### Failure

- User press the light controlling button, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User press the light controlling button, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.

#### **Use Case 2:** Music Controlling

##### Success

- The Music turns on when user press the light controlling button.
- The Music turns on when user send a voice message containing the key words of turn on & music.

#### Failure

- User press the music controlling button, but the signal of turn on the music does not go through to the server, and the music does not turn on.
- User press the music controlling button, the signal of turn on the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn on.
- User send a voice mail which contains the key words of turn on & music, but the signal of turn on the music does not go through to the server, and the music does not turn on.
- User send a voice mail which contains the key words of turn on & music, the signal of turn on the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn on.

### **Use Case 3:** Modify the lightness

#### Success

- The lightness grow/fade when user press the lightness controlling button.
- The lightness grow/fade when user send a voice message containing the key words of turn up/down & light.

#### Failure

- User press the light controlling button, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User press the light controlling button, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, but the signal of turn on the button does not go through to the server, and the light does not turn on.
- User send a voice mail which contains the key words of turn on & light, the signal of turn on the button go through the server, but the signal which the server gives out does not go through to the hardware, and the light does not turn on.

### **Use Case 4:** Modify the volume of the music

#### Success

- The Music turns up/down when user press the volume controlling button.
- The Music turns up/down when user send a voice message containing the key words of turn up/down & music.

#### Failure

- User press the music volume button, but the signal of turn up/down the music does not go through to the server, and the music does not turn up/down.
- User press the music volume button, the signal of turn up/down the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn up/down.
- User send a voice mail which contains the key words of turn up/down & music, but the signal of turn up/down the music does not go through to the server, and the music does not turn up/down.

- User send a voice mail which contains the key words of turn up/down & music, the signal of turn up/down the music go through the server, but the signal which the server gives out does not go through to the hardware, and the music does not turn up/down.

**Use Case 6:** User Log In

Success

- User inputs username and password successfully. The system has to verify if the username and password exists and matches in the system. The match is made and the user is logged into his/her account.

Failure

- User inputs username and password, but when the system goes to verify, the username is incorrect and it doesnt match any username in the database and it displays a wrong username/ password error message.
- User inputs username and password, but when the system goes to verify, the password does not match with the one saved in the database and it displays a wrong username/password error message.
- User inputs username and password, but when the system goes to verify, neither the username nor the password match with the one saved in the database and it displays a wrong username/ password error message.

**Use Case 8:** Remove user

Success

- The user names and data is successfully removed and the deserted user-names cannot be used again.

Failure

- The user names and data is not removed and the deserted usernames can be used again.
- The user names and data is partially removed, the deserted usernames cannot be used again, however the data left takes up spaces in the server.

**Use Case 12:** Control information feedback

Success

- The user interface get feedback to check whether the voice have been identified by the server.

Failure

- The user interface get no feedback to check whether the voice have been identified by the server.
- The user interface get feedback to check whether the voice have been identified by the server, however the feedback is wrong.

**Use Case 13:** Network error notification

Success

- The user interface of Android cellphone display a text of connection notifications of both the signal between cellphone and server and the signal between server and MCU, which is right.

Failure

- The user interface of Android cellphone display no notification of the network status.
- The user interface of Android cellphone display a text of connection notifications of both the signal between cellphone and server and the signal between server and MCU, which is wrong.
- The user interface of Android cellphone display a text of connection notifications one of the signal between cellphone and server and the signal between server and MCU, which is wrong.

## 9 Project Management

In this section we are going to talk about our tasks and plans. We have 10 kinds of jobs in general, the system design related jobs and paper works. We treat each week as a building circle so that our plan circle is a week. Table 2 shows the works and the related due time. Blue part means the tasks has already been finished; Green part represents the jobs we are working on; the yellow parts mean the tasks will be done in the future.

Task	SEP21	OCT4	OCT11	OCT18	OCT25	NOV1	NOV8	NOV15	NOV22	NOV29	DEC6	DEC13
Website Maintenance												
App UI Design												
App Frame Design												
Server Frame Design												
Smart Element Design												
Communication Design												
System De-Bugging I												
Problem Fix												
System De-Bugging II												
Problem Fix												
Report 1												
Report 2												
Report 3												
Demo 1												
Demo 2												
Archive												
Documentation												

Table 2 Plans of Works

Figure 47: project management

## 10 Reference

- [1]**ANDROID (OS)**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/ANDROID\\_\(OPERATING\\_SYSTEM\)](http://en.wikipedia.org/wiki/Android_(operating_system))
- [2]**FUNCTIONAL REQUIREMENT**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/FUNCTIONAL\\_REQUIREMENT](http://en.wikipedia.org/wiki/Functional_requirement)
- [3]**USER STORIES**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [4]**ISO**. [HTTPS://DEVELOPER.APPLE.COM](https://developer.apple.com)
- [5]**ANDROID APPLICATION PLATFORM**. [WWW.MANTRAIS.COM/MOBILE/ANDROID-APPLICATION-DEVELOPMENT/](http://www.mantrais.com/mobile/android-application-development/)
- [6]**VOICE COMMAND DEVICE**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/VOICE\\_COMMAND\\_DEVICE](http://en.wikipedia.org/wiki/Voice_command_device)
- [7]**NON-FUNCTIONAL REQUIREMENT**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/NON-FUNCTIONAL\\_REQUIREMENT](http://en.wikipedia.org/wiki/Non-functional_requirement)
- [8]**FURPS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/FURPS](http://en.wikipedia.org/wiki/Furps)
- [9]**NON-FUNCTIONAL REQUIREMENT**. [HTTP://WWW.MOUNTANGOATSOFTWARE.COM/BLOG/NON-FUNCTIONAL-REQUIREMENTS-AS-USER-STORIES](http://www.mountangoatssoftware.com/blog/non-functional-requirements-as-user-stories)
- [10]**HOUSING**. [HTTP://MONEYNING.COM/HOUSING/THE-FIVE-YEAR-RULE-FOR-BUYING-A-HOUSE/](http://moneyning.com/housing/the-five-year-rule-for-buying-a-house/)
- [11]**UI-OVERVIEW**. [HTTP://DEVELOPER.ANDROID.COM/DESIGN/GET-STARTED/UI-OVERVIEW.HTML](http://developer.android.com/design/get-started/ui-overview.html)
- [12]**ANDROID**. [HTTP://WWW.ANDROID.COM/ABOUT/](http://www.android.com/about/)
- [13]**MOBILEPHONE UI DESIGN**. [HTTPS://WWW.GOOGLE.COM.HK/SEARCH?Q=MOBILE+UI+DESIGN&SAFE=STRICT&SA=X&TBM=ISCH&TBO=U&SOURCE=UNIV&EI=30LMUOSCLIHQ8QTBNOG4AG&VED=OCduQSAQ&BIW=1280&BIH=650&DPR=1](https://www.google.com.hk/search?q=mobile+ui+design&safe=strict&sa=x&tbm=isch&tbo=u&source=univ&ei=30LMUOSCLIHQ8QTBNOG4AG&ved=OCduQSAQ&biw=1280&bih=650&dpr=1)
- [14]**MOBILE-PATTERNS**. [HTTP://WWW.MOBILE-PATTERNS.COM/](http://www.mobile-patterns.com/)
- [15]**MOBILE APP DESIGN**. [HTTP://WWW.1STWEBDESIGNER.COM/DESIGN/MOBILE-APPS-DESIGNS/](http://www.1stwebdesigner.com/design/mobile-apps-designs/)
- [16]**GLADE**. [HTTPS://GLADE.GNOME.ORG/](https://glade.gnome.org/)
- [17]**HOME AUTOMATION AND CONTROL**. [HTTP://WWW.SAVANTSYSTEMS.COM/HOME\\_AUTOMATION\\_AND\\_CONTROL.ASPX](http://www.savantsystems.com/home_automation_and_control.aspx)
- [18]**SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [19]**STAKEHOLDERS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/STAKEHOLDERS](http://en.wikipedia.org/wiki/Stakeholders)
- [20]**USE CASE**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/USE\\_CASE#ACTORS](http://en.wikipedia.org/wiki/Use_case#actors)
- [21]**AUTOHOME**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF](http://eceweb1.rutgers.edu/~marsic/books/se/projects/autohome/2012-g1-report3.pdf)
- [22]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF](http://eceweb1.rutgers.edu/~marsic/books/se/projects/other/2012-g4-report3.pdf)
- [23]**SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [24]**SYSTEM SEQUENCE DIAGRAM**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/SYSTEM\\_SEQUENCE\\_DIAGRAM](http://en.wikipedia.org/wiki/System_sequence_diagram)
- [25]**SOFTWARE ENGINEERING**. [HTTP://WWW.ECE.RUTGERS.EDU/~MARSIC/BOOKS/SE/](http://www.ece.rutgers.edu/~marsic/books/se/)
- [26]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/AUTOHOME/2012-G1-REPORT3.PDF](http://eceweb1.rutgers.edu/~marsic/books/se/projects/autohome/2012-g1-report3.pdf)

- [27]**SE PROJECT REPORT3**. [HTTP://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF](http://ECEWEB1.RUTGERS.EDU/~MARSIC/BOOKS/SE/PROJECTS/OTHER/2012-G4-REPORT3.PDF)
- [28]**EFFORT ESTIMATION**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/SOFTWARE\\_DEVELOPMENT\\_EFFORT\\_ESTIMATION](http://EN.WIKIPEDIA.ORG/WIKI/SOFTWARE_DEVELOPMENT_EFFORT_ESTIMATION)
- [29]**HARDWARE DOCUMENTATION**. [HTTP://WWW.FRITZING.ORG](http://WWW.FRITZING.ORG)
- [30]**INFRARED COMMUNICATION**. [HTTP://WWW.SBPROJECTS.COM/KNOWLEDGE/IR/SIRC.PHP](http://WWW.SBPROJECTS.COM/KNOWLEDGE/IR/SIRC.PHP)
- [31]**INFRARED COMMUNICATION PROTOCOL**. [HTTP://LEARN.ADAFRUIT.COM/TRINKET-GEMMA-IR-REMOTE-CONTROL/USING-IR-CODES-1](http://LEARN.ADAFRUIT.COM/TRINKET-GEMMA-IR-REMOTE-CONTROL/USING-IR-CODES-1)
- [32]**COMMUNICATION PROTOCOL**. [HTTP://WWW.PIC-TRONICS.COM/IR-TRANSMITTER-AND-RECEIVER.PHP](http://WWW.PIC-TRONICS.COM/IR-TRANSMITTER-AND-RECEIVER.PHP)
- [33]**SERIALPORT COMMUNICATION**. [HTTP://PLAYGROUND.ARDUINO.CC//CSHARP/SERIALCOMMSCSHARP](http://PLAYGROUND.ARDUINO.CC//CSHARP/SERIALCOMMSCSHARP)
- [34]**INFRARED COMMUNICATION COMMUNICATION**. [HTTP://WWW.PYROELECTRO.COM/TUTORIALS/INFRARED\\_IR\\_RECEIVER/IR\\_THEORY\\_1.HTML](http://WWW.PYROELECTRO.COM/TUTORIALS/INFRARED_IR_RECEIVER/IR_THEORY_1.HTML)
- [35]**WIRELESS COMMUNICATION**. [HTTP://WW1.MICROCHIP.COM/DOWNLOADS/EN/DEVICEDOC/ADN006.PDF](http://WW1.MICROCHIP.COM/DOWNLOADS/EN/DEVICEDOC/ADN006.PDF)
- [36]**PIC MICROCONTROLLER**. [HTTP://WWW.MICROCHIP.COM/WWWPRODUCTS/DEVICES.ASPX?DDOCNAME=EN010297](http://WWW.MICROCHIP.COM/WWWPRODUCTS/DEVICES.ASPX?DDOCNAME=EN010297)
- [37]**ARDUINO BASICS**. [HTTP://HTTP://WWW.NFIAUTOMATION.ORG/NFI\\_PROJECTS.HTML](http://HTTP://WWW.NFIAUTOMATION.ORG/NFI_PROJECTS.HTML)
- [38]**COMMUNICATION PROTOCOL**. [HTTP://RATUL-ELECTRONICS.BLOGSPOT.COM/2012/03/RF-MODULE-TRANSMITTER-RECEIVER.HTML](http://RATUL-ELECTRONICS.BLOGSPOT.COM/2012/03/RF-MODULE-TRANSMITTER-RECEIVER.HTML)
- [39]**ARDUINO BASICS**. [HTTP://HTTP://WWW.NFIAUTOMATION.ORG/NFI\\_PROJECTS.HTML](http://HTTP://WWW.NFIAUTOMATION.ORG/NFI_PROJECTS.HTML)
- [40]**MICROPHONE BASICS**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/MICROPHONE](http://EN.WIKIPEDIA.ORG/WIKI/MICROPHONE)
- [41]**IR PROTOCOLS**. [HTTP://WWW.EPANORAMA.NET/LINKS/IRREMOTE.HTML](http://WWW.EPANORAMA.NET/LINKS/IRREMOTE.HTML)
- [42]**STAR NETWORK**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/STAR\\_NETWORK](http://EN.WIKIPEDIA.ORG/WIKI/STAR_NETWORK)
- [43]**HANDSHAKING PROTOCOL**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/HANDSHAKING](http://EN.WIKIPEDIA.ORG/WIKI/HANDSHAKING)
- [44]**CHECKSUM**. [HTTP://EN.WIKIPEDIA.ORG/WIKI/CHECKSUM](http://EN.WIKIPEDIA.ORG/WIKI/CHECKSUM)
- [45]**MAKING THINGS TALK BY TOM IGOE**