كلية الهندسة

SMART HOME SYSTEM

DECEMBER 28, 2018

AIN SHAMS UNIVERSITY – FACULTY OF ENGINEERING
1 Al-Sarayyat st, Abbassiya, Cairo 11517, Egypt

FACULTY OF ENGINEERING

AIN SHAMS UNIVERSITY

CSE222: Software Engineering (2)

Smart Home System

Submitted By:

Abdelrahman Ibrahim ELGhamry	**Hossam ELDin Khaled Mohmed**	**Abdelrahman Amr Issawi**
Ghamry98@hotmail.com 16P3043@eng.asu.edu.eg	hossampen97@gmail.com 16P3025@eng.asu.edu.eg	aid-issawi@hotmail.com 16P6001@eng.asu.edu.eg

Submitted to:

Prof. Dr. Gamal Abdelshafy

DECEMBER 28, 2018

A REPORT FOR SOFTWARE ENGINEERING COURSE CODDED CSE222 WITH THE REQUIREMENTS OF AIN SHAMS UNIVERSITY

TABLE OF CONTENTS

1.0 BRIEF DESCRIPTION

Smart Homes can be described as introduction of technology within the home environment to provide convenience, comfort, security and energy efficiency to its occupants.

The project presents a low cost and flexible home control and monitoring system using an embedded microcontroller with IP connectivity for accessing and controlling devices and appliances remotely using Smart phone application adding intelligence to home environment can provide increased quality of life.

Home automation not only refers to reduce human efforts but also energy efficiency and time saving. With the introduction of the Internet of Things (IoT), the research and implementation of home automation are getting more popular.

2.0 SYSTEM REQUIREMENTS

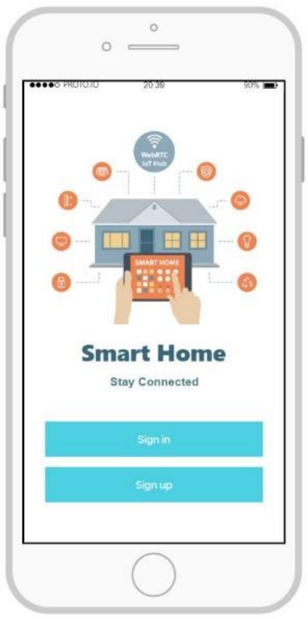
2.1 Functional Requirements

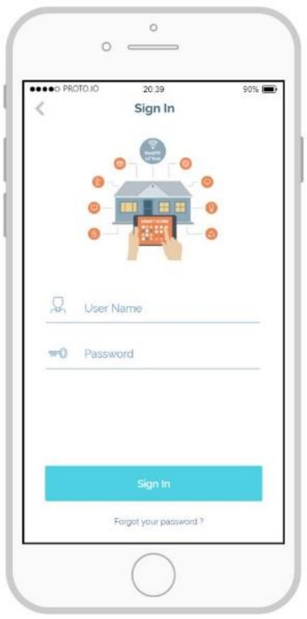
1. The system maintains a room temperature by reading the heat sensor data and adjusting the air-conditioner, the temperature value is set by the user.
2. The system automatically closes the gas lock whenever the gas sensor indicates a gas leak and notify the user.
3. If the system fails to close the gas lock, the system automatically reports a gas leak to New Jersey Natural Gas with the location of the leak and alert the user.
4. The system automatically closes the water valves whenever the water sensor indicates a water leak and notify the user.
5. If the system fails to close the water valve, the system automatically reports the location of the leak to nearest water station and alerts the user.
6. The system handles remote controls from the user through mobile application.
   1. Set the room temperature.
   2. Lock/unlock any door registered on the system.
   3. Switch on and off any light registered on the system.
7. The system offers a secure online connection, so the user can be in control of the system from anywhere in the world.

2.2 Non-Functional Requirements

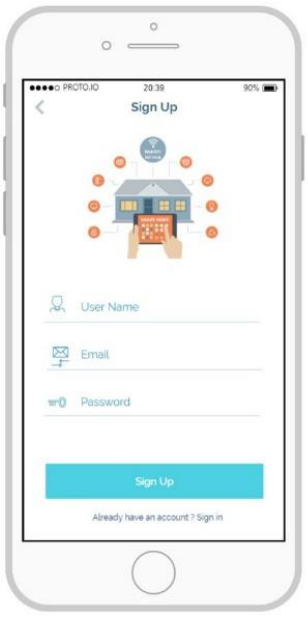
1. Deploy at least two gas sensors to detect gas leaks.
2. System’s response time to handle user controls must not exceed five seconds.
3. Divide the system into components for the end-user to choose the desired functionalities.
4. The budget of this project has a maximum of 100,000$.

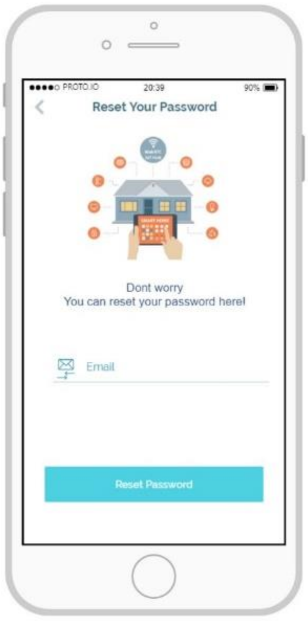
3.0 EVOLUTIONARY PROTOTYPE

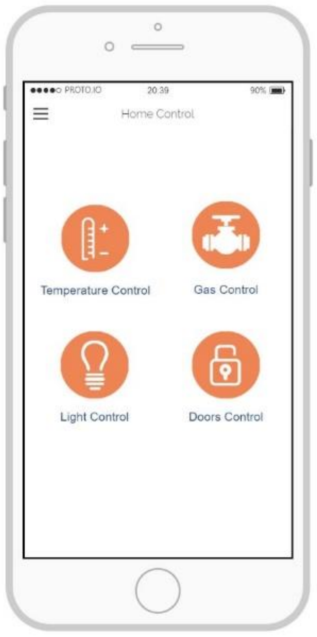




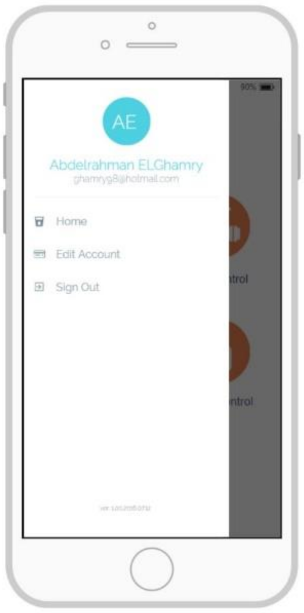
1. Startup screen of the application, where the user can sign in if he got an existing account or chooses to sign up.
2. User uses his login credentials as username and password to access his home control application.

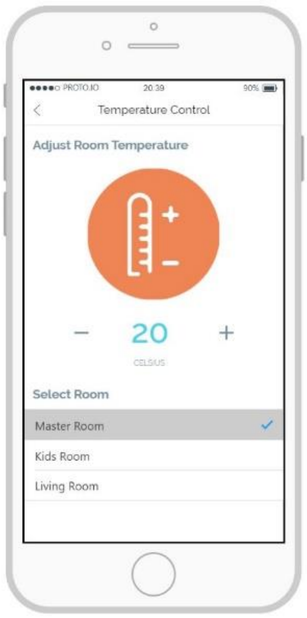




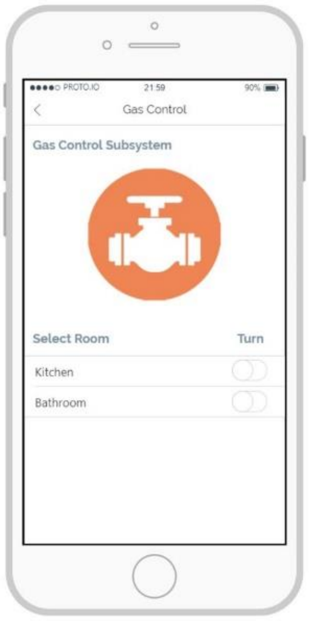


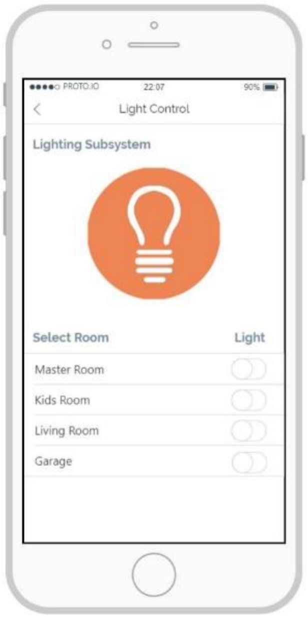
3. The user can make a new account and enjoy using the application.
4. The user can reset his account password if he forgot it.
5. The home page where the user can have full access control on home temperature, gas, light and doors.

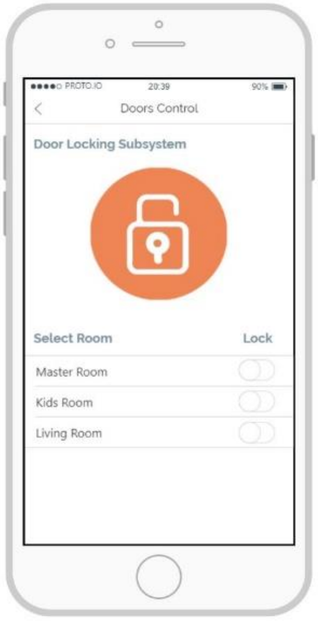




6. The user can access his dashboard and edit his account details or sign out.
7. The user can choose the room and adjust its temperature.



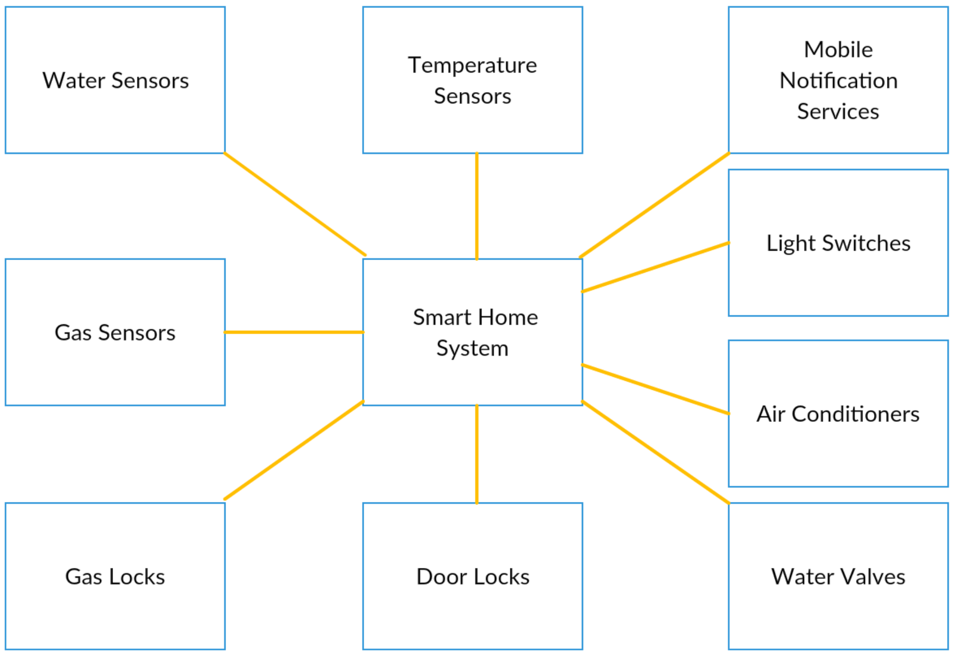




8. The user can check and turn on/off the gas line.
9. The user can check and control the lights all over the home.
10. The user can check and control the doors all over the home.

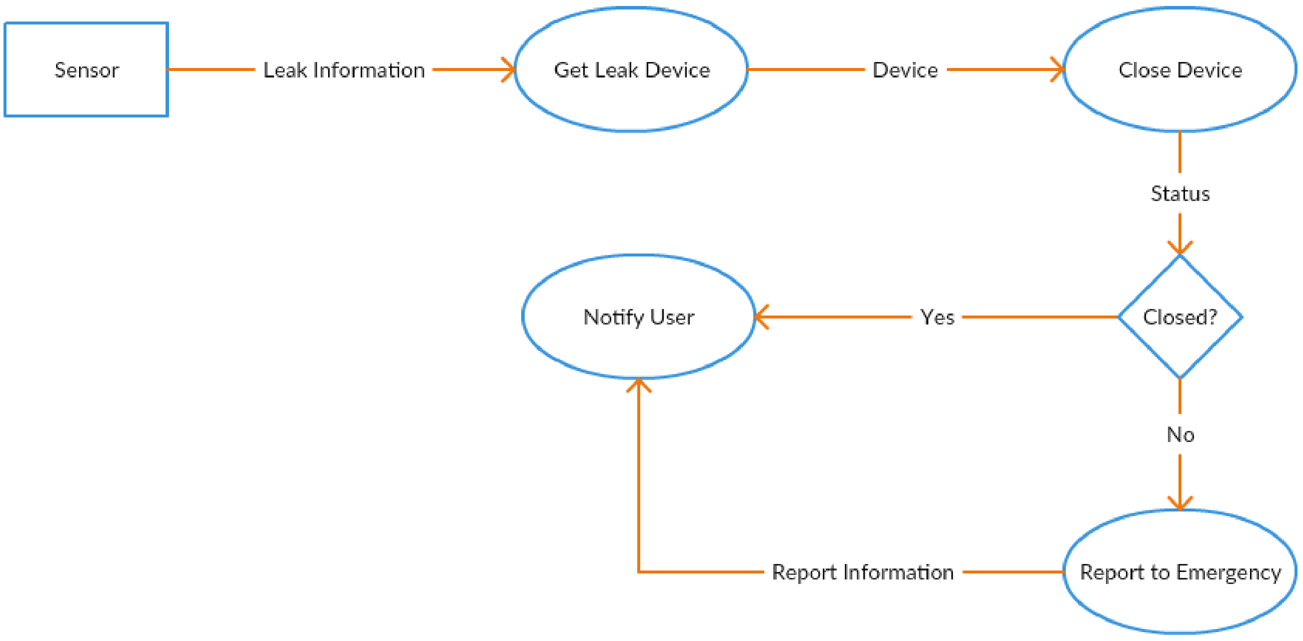
4.0 SYSTEM MODELS

4.1 Context Model

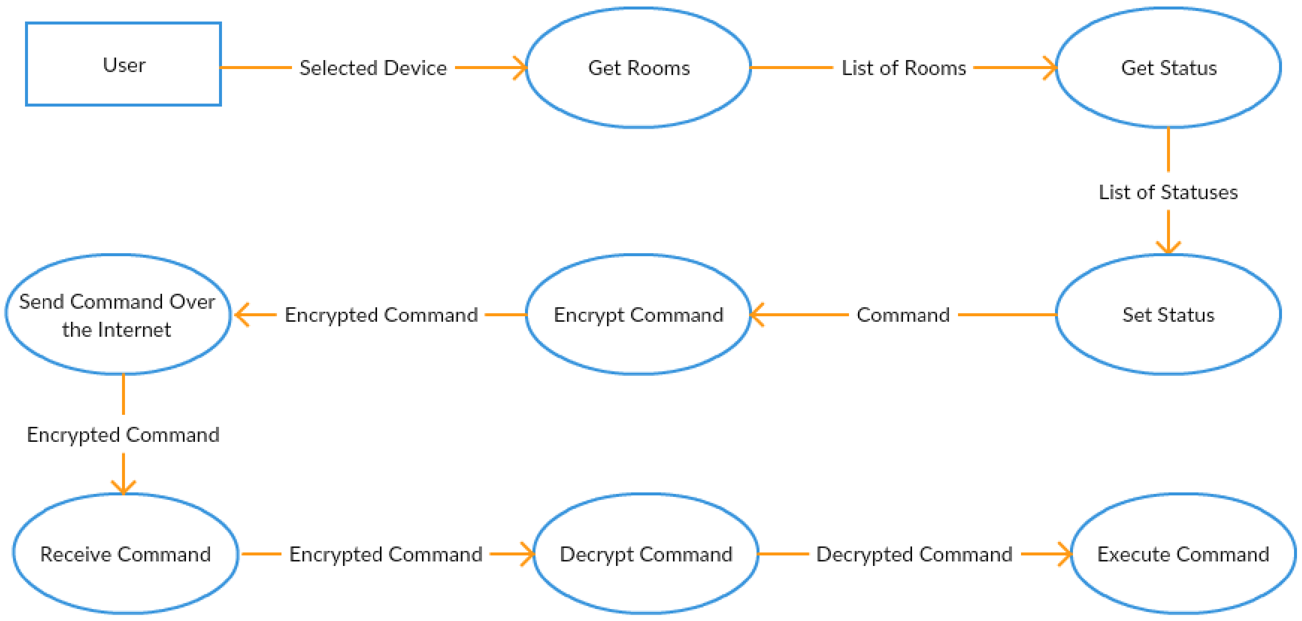


4.2 Process Model

4.2.1 Monitoring Sub-System



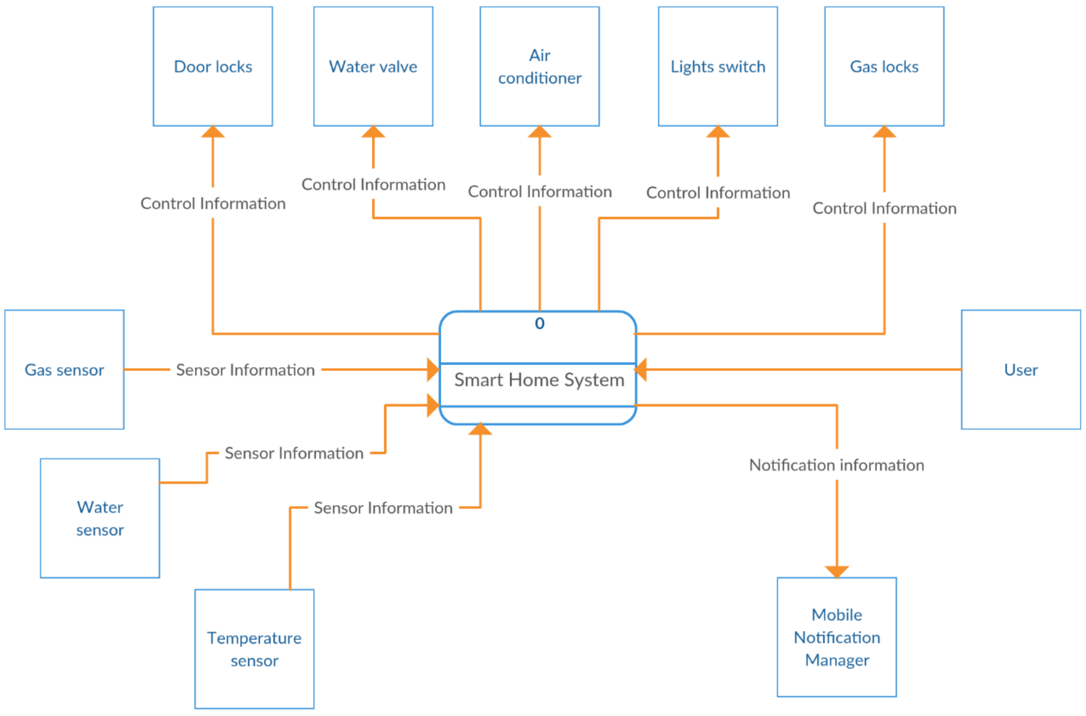
4.2.2 User Command Sub-System



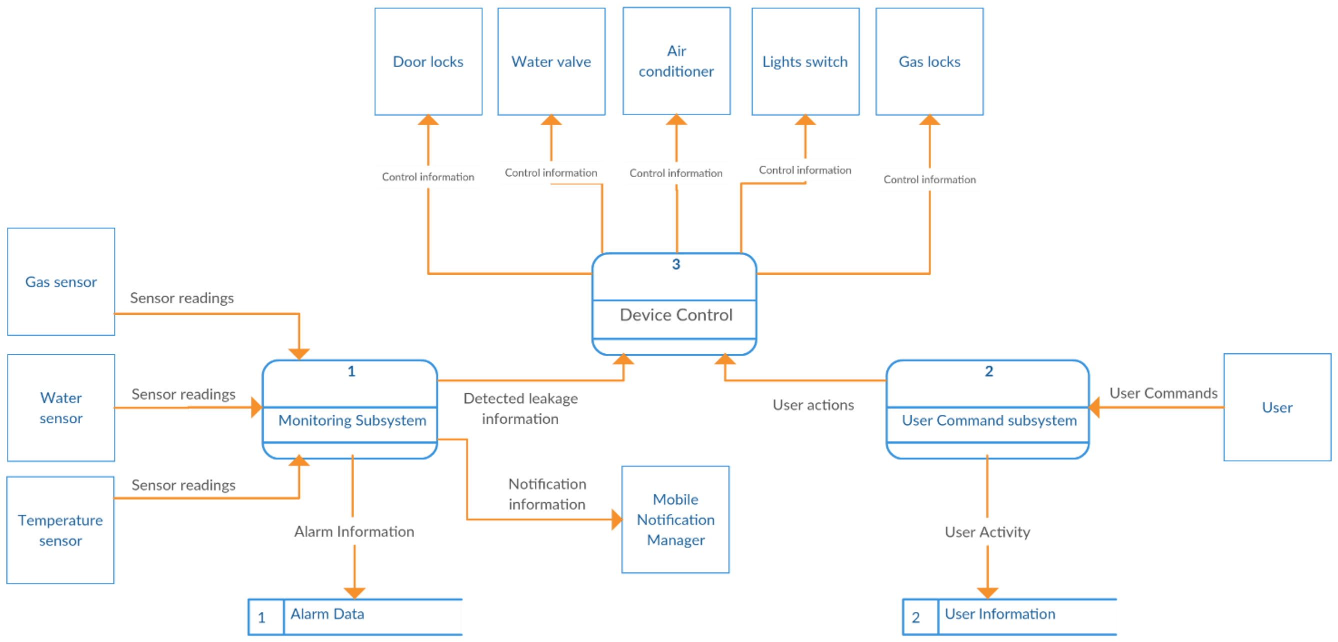
4.3 Behavioral Model

4.3.1 Data Flow Diagram

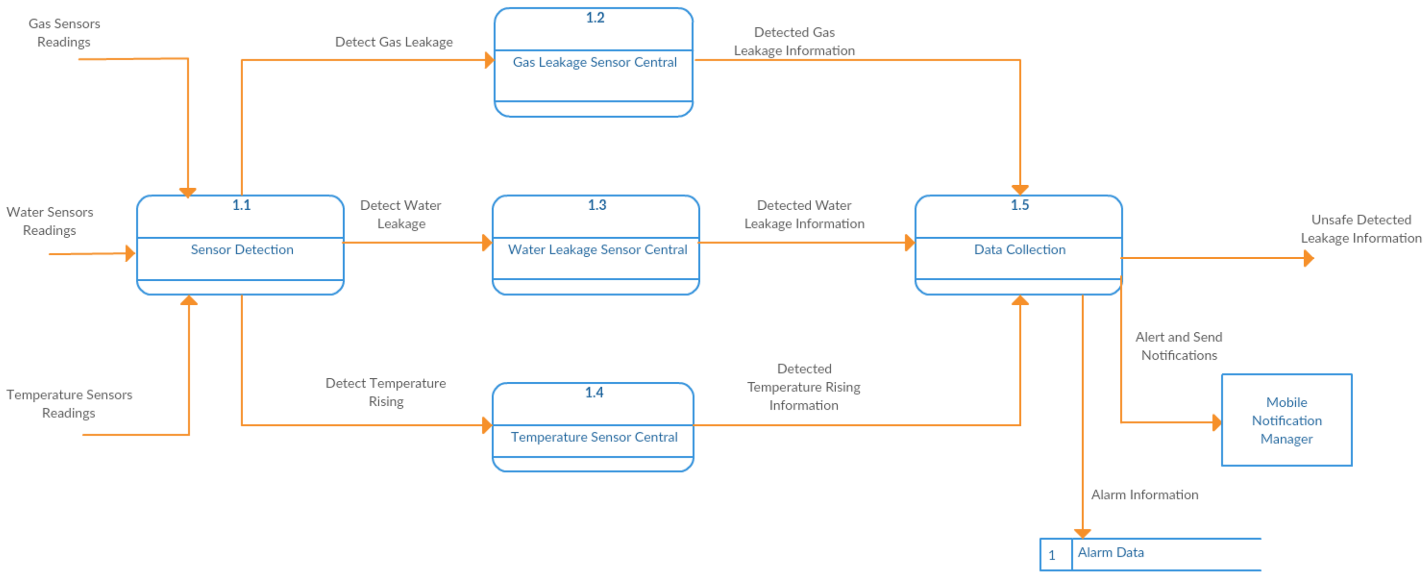
4.3.1.1 Context Diagram



4.3.1.2 Level 0 Diagram

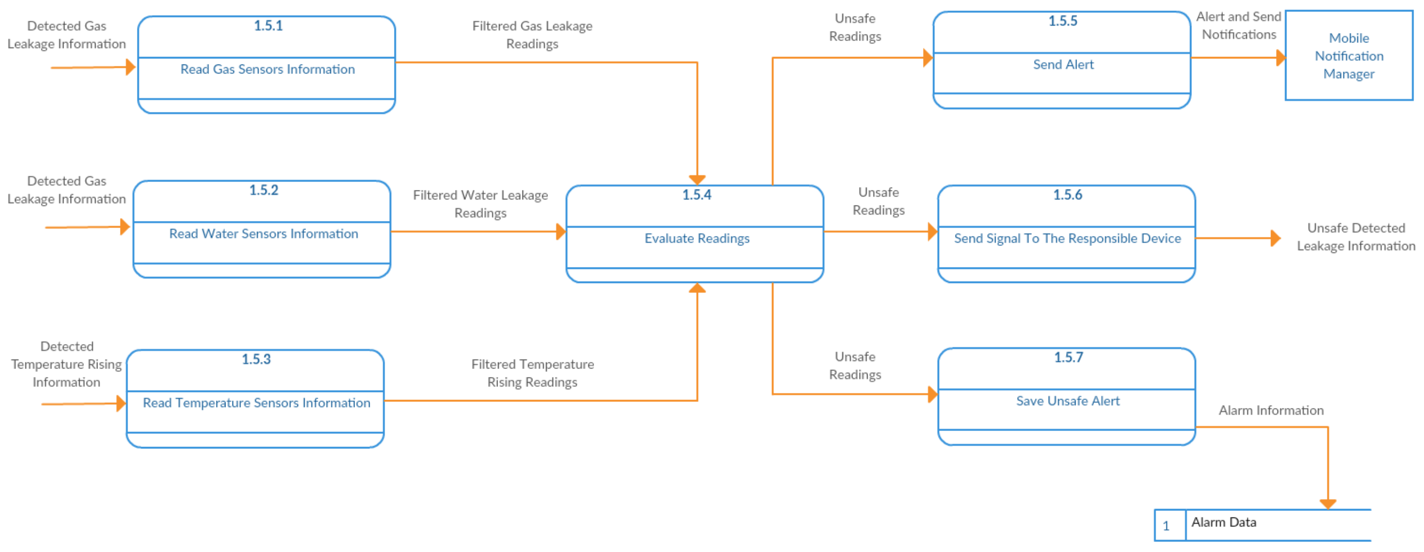


4.3.1.3 Level 1 Diagram

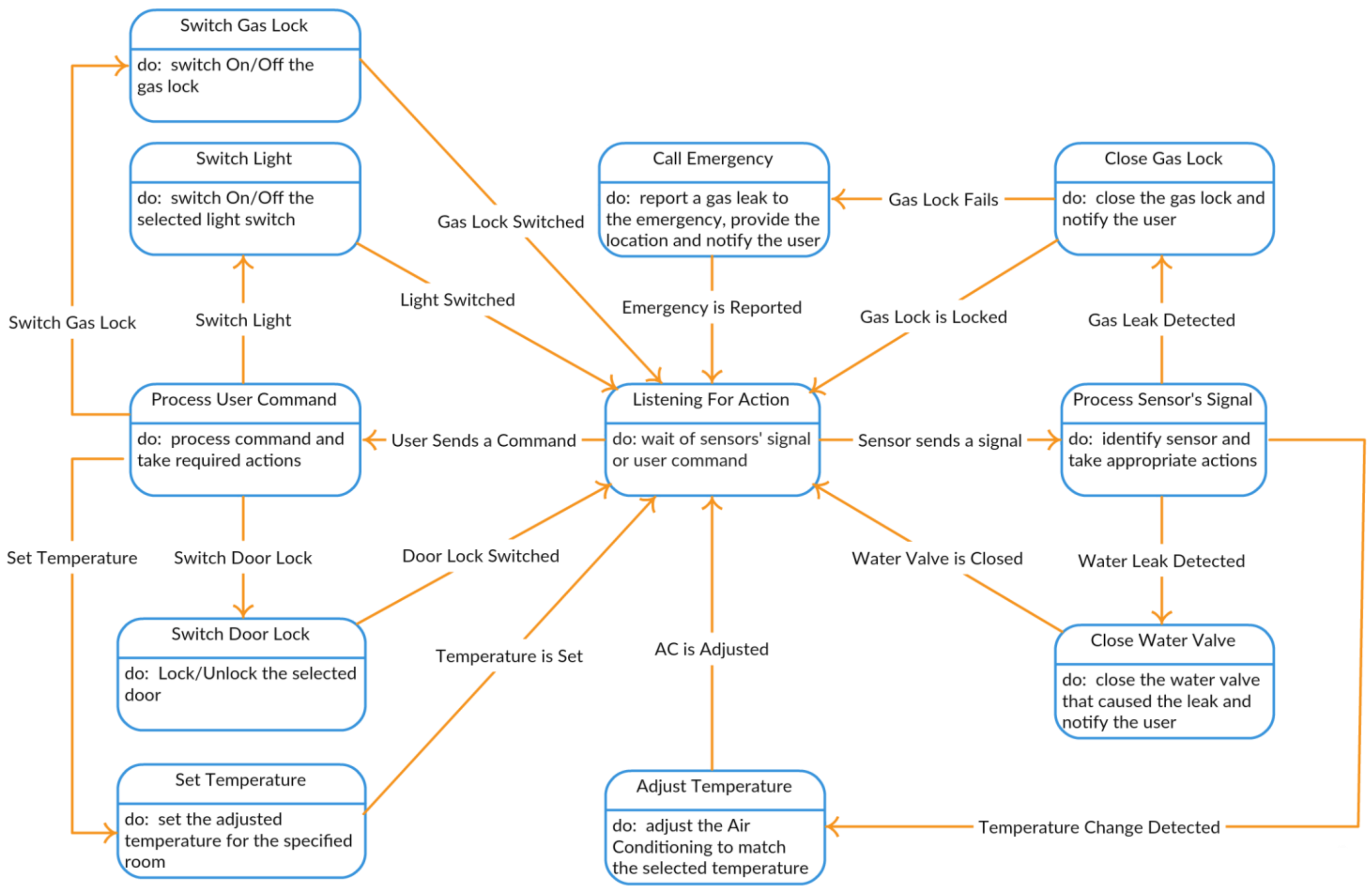


Gas Sensors Readings
Detect Gas Leakage
1.2
Gas Leakage Sensor Central
Detected Gas Leakage Information
Water Sensors Readings
1.1
Sensor Detection
Detect Water Leakage
1.3
Water Leakage Sensor Central
Detected Water Leakage Information
1.5
Data Collection
Unsafe Detected Leakage Information
Temperature Sensors Readings
Detect Temperature Rising
1.4
Temperature Sensor Central
Detected Temperature Rising Information
Alert and Send Notifications
Mobile Notification Manager
Alarm Information
1 Alarm Data

4.3.1.4 Level 2 Diagram


Detected Gas Leakage Information
1.5.1
Read Gas Sensors Information
Filtered Gas Leakage Readings
Detected Gas Leakage Information
1.5.2
Read Water Sensors Information
Filtered Water Leakage Readings
Detected Temperature Rising Information
1.5.3
Read Temperature Sensors Information
Filtered Temperature Rising Readings
1.5.4
Evaluate Readings
Unsafe Readings
1.5.5
Send Alert
Alert and Send Notifications
Mobile Notification Manager
Unsafe Readings
1.5.6
Send Signal To The Responsible Device
Unsafe Detected Leakage Information
Unsafe Readings
1.5.7
Save Unsafe Alert
Alarm Information
1
Alarm Data

4.3.2 State Machine Diagram


Switch Gas Lock
do: switch On/Off the gas lock
Switch Light
do: switch On/Off the selected light switch
Call Emergency
do: report a gas leak to the emergency, provide the location and notify the user
Close Gas Lock
do: close the gas lock and notify the user
Gas Lock Switched
Gas Lock Fails
Light Switched
Emergency is Reported
Gas Lock is Locked
Switch Gas Lock
Switch Light
Gas Leak Detected
Process User Command
do: process command and take required actions
User Sends a Command
Listening For Action
do: wait of sensors' signal or user command
Sensor sends a signal
Process Sensor's Signal
do: identify sensor and take appropriate actions
Set Temperature
Switch Door Lock
Door Lock Switched
Water Valve is Closed
Water Leak Detected
Switch Door Lock
do: Lock/Unlock the selected door
Temperature is Set
AC is Adjusted
Close Water Valve
do: close the water valve that caused the leak and notify the user
Set Temperature
do: set the adjusted temperature for the specified room
Adjust Temperature
do: adjust the Air Conditioning to match the selected temperature
Temperature Change Detected

4.3.2.1 State Description

State	*Description*
Listening for Action	The system is waiting for a sensor to send a signal or a command from the user.
Process Sensor's Signal	The system is processing the sensor's signal, to identify the sensor and the appropriate action.
Close Gas Lock	The system tries to close the gas lock and notifies the user of a gas leak.
Call Emergency	The system reports a gas leak to the emergency, provides the location and notify the user.
Close Water Valve	The system decides which water valve to close, close it and notify the user.
Adjust Temperature	The system controls the air conditioning to match the selected temperature.
Process User Command	The system is processing the command from the user and identifies which actions are required.
Switch Door Lock	The system locks or unlocks a door based on the user command.
Set Temperature	The system sets the desired temperature of a room specified by the user.
Switch Light	The system switches a light on or off based on the user command.
Switch Gas Lock	The system switches on or off the gas lock based on the user command.

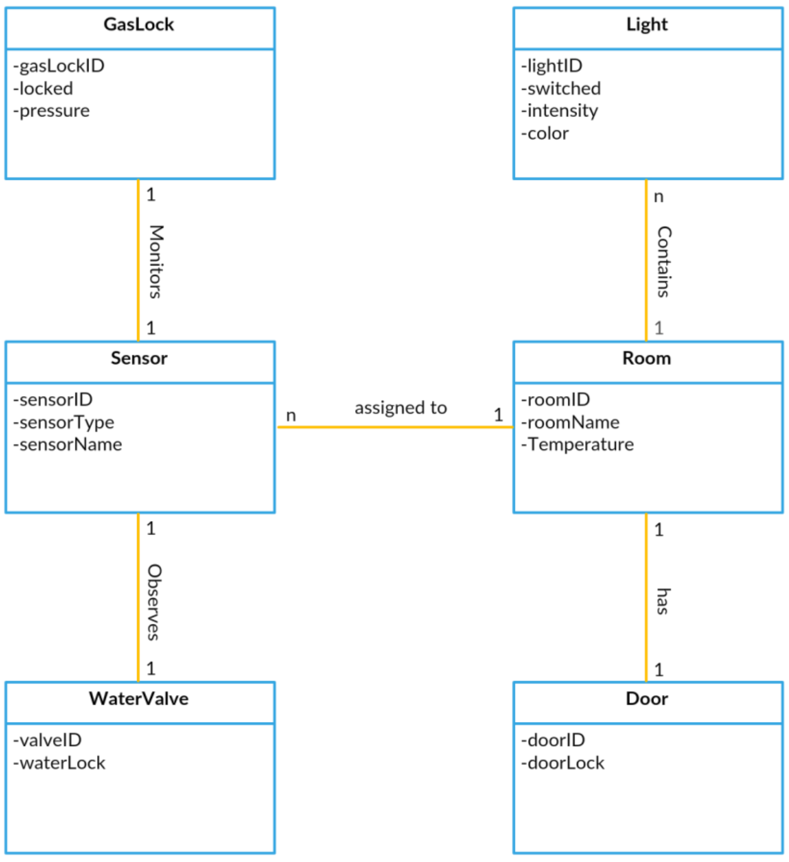
4.3.2.2 Stimuli Description

Stimuli	*Description*
Senor sends a signal	The system detected a signal from a sensor could be the gas or the water or the heat sensor.
Gas Leaked Detected	The system identified a gas leak from the gas sensor signal.
Gas Lock Fails	The system fails in closing the gas lock.
Water Leak Detected	The system identified a water leak from the water sensors array.
Temperature Change Detected	The system identified a temperature change with respect to the user selected temperature.
User sends a Command	The system received a command from the user.
Switch Door Lock	The system processed "Switch Door Lock" command from the user.
Set Temperature	The system processed "Set Temperature" command from the user.

Switch Light	The system processed "Switch Light" command from the user.
Switch Gas Lock	The system processed "Switch Gas Lock" command from the user.
Gas Lock Switched	The system switched the gas locked.
Temperature is set	The system set the selected temperature.
Light Switched	The system switched the selected light switch.
Door Lock Switched	The system switched the lock of the selected door.
Gas Lock is Locked	The system could lock the gas lock successfully after detecting a gas leak.
Water Valve is Closed	The system closed the water valve that caused the water leak.
AC is Adjusted	The system adjusted the air conditioner to match user's selected temperature.
Emergency is Reported	The system reported a gas leak to the emergency.

4.4 Semantic Model

4.4.1 Semantic Diagram



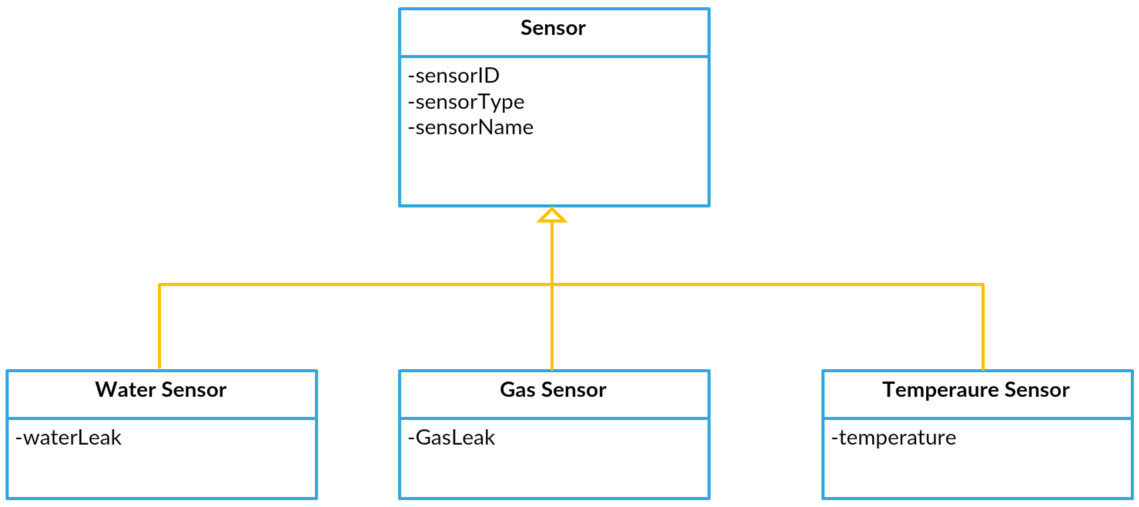
4.4.2 Data Dictionary

Name	*Description*	*Type*	*Date*
Gas Lock	Details about a gas lock which the system can lock/unlock.	Entity	19-12-2018
gasLockID	A unique ID to identify a gas lock so the system can control it.	Attribute	19-12-2018
GasLock	Determines whether the gas lock is locked or not.	Attribute	19-12-2018
Pressure	Holds the gas pressure value.	Attribute	19-12-2018
Monitors	A 1:1 relationship between a gas lock and the sensor that checks for gas leakage.	Relation	19-12-2018
Light	Details about the light bulb that the system can control.	Entity	19-12-2018
LightID	A unique ID to identify a light switch so the system can control it.	Attribute	19-12-2018
Switched	Determines whether the light is turned on or off.	Attribute	19-12-2018
Intensity	Holds the light's intensity value which the system can modify.	Attribute	19-12-2018
Color	Hold the light's color value which the system can modify.	Attribute	19-12-2018
Contains	A 1:n relationship between the room and the light switches in the room.	Relation	19-12-2018
Room	Details about a room that contains some light switches, a door and some sensors.	Entity	19-12-2018
RoomID	A unique ID for the room, so the system can identify.	Attribute	19-12-2018
RoomName	The room's name, modifiable by the user.	Attribute	19-12-2018
Temperature	Holds the room temperature that system maintains.	Attribute	19-12-2018
Has	A 1:1 relationship between a door and a room that has the door	Relation	19-12-2018
Door	Details about a door which the system can lock/unlock.	Entity	19-12-2018
DoorID	A unique ID for the door, so the system can identify.	Attribute	19-12-2018
DoorLock	Determines whether a door is locked or unlocked.	Attribute	19-12-2018

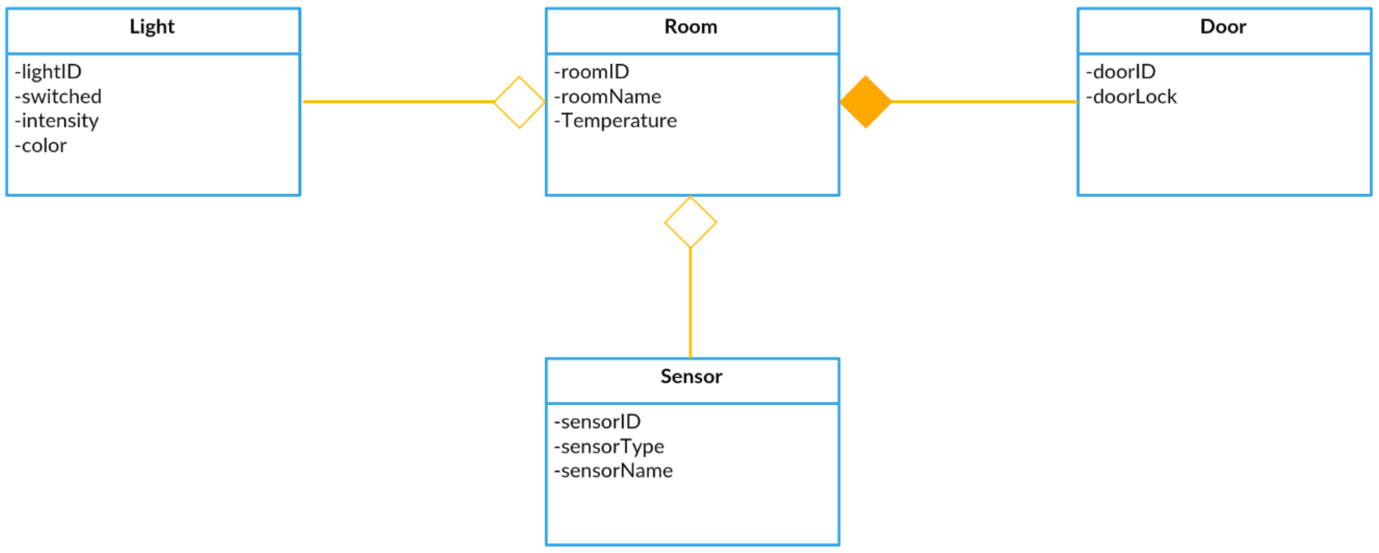
Sensor	Details about a sensor which the system uses to identify leaks.	Entity	19-12-2018
SensorID	A unique ID for the sensor, so the system can identify it.	Attribute	19-12-2018
SensorType	Determines the type of the sensor can be gas, water or temperature.	Attribute	19-12-2018
SensorName	Holds a name for a sensor to simply assign a sensor to a room.	Attribute	19-12-2018
Assigned To	A 1:n relationship between a room and the sensors assigned to the room.	Relation	19-12-2018
Observes	A 1:1 relationship between a water valve and the sensor that checks for water leakage.	Relation	19-12-2018
WaterValve	Details about a water valve which the system can lock or unlock.	Entity	19-12-2018
ValveID	A unique ID for the valve, so the system can identify it.	Attribute	19-12-2018
WaterLock	indicates whether a water valve is locked or unlocked	Attribute	19-12-2018

4.5 Object Model

4.5.1 Inheritance Diagram

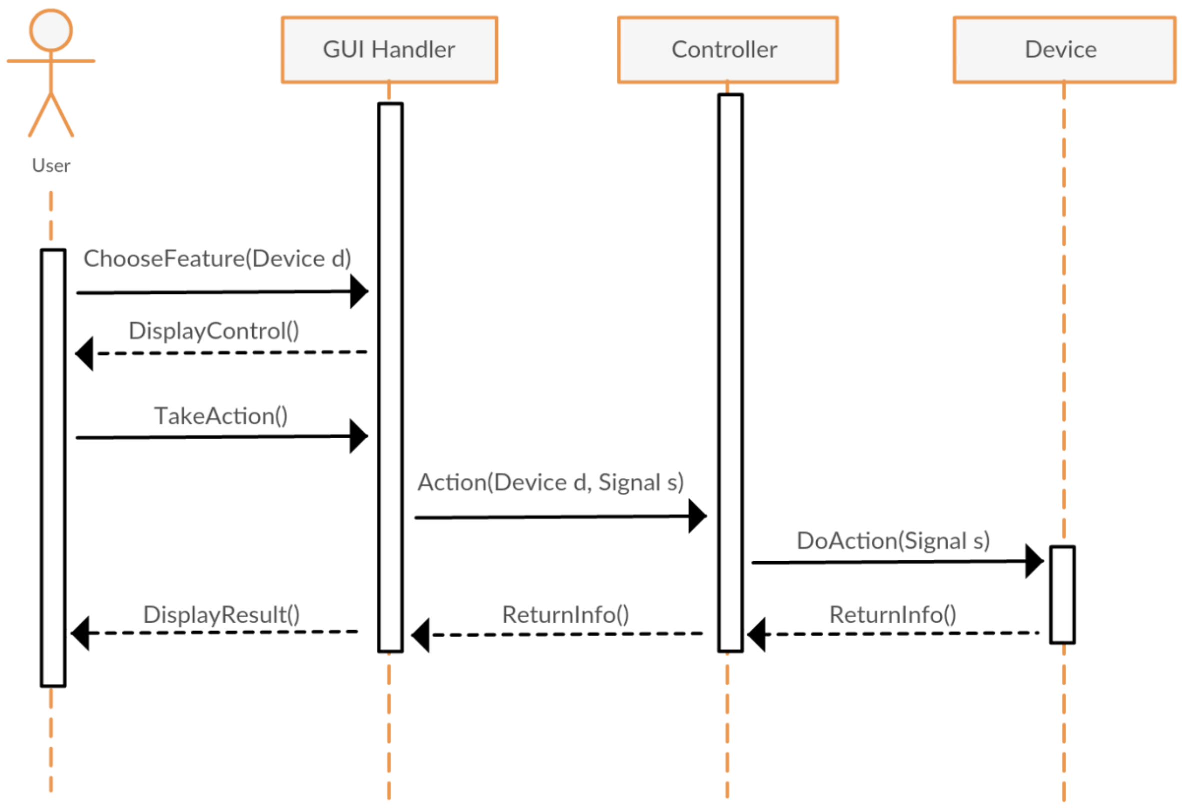


4.5.2 Aggregation Diagram

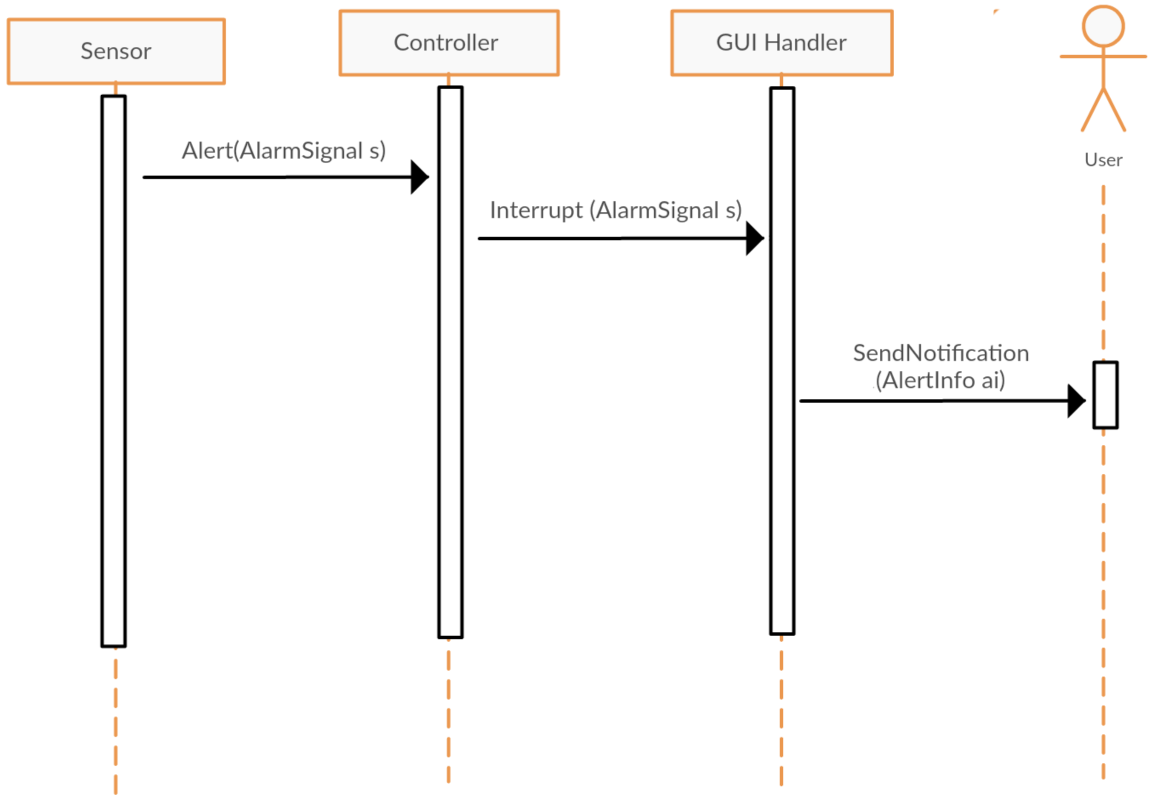


4.6 Sequence Model

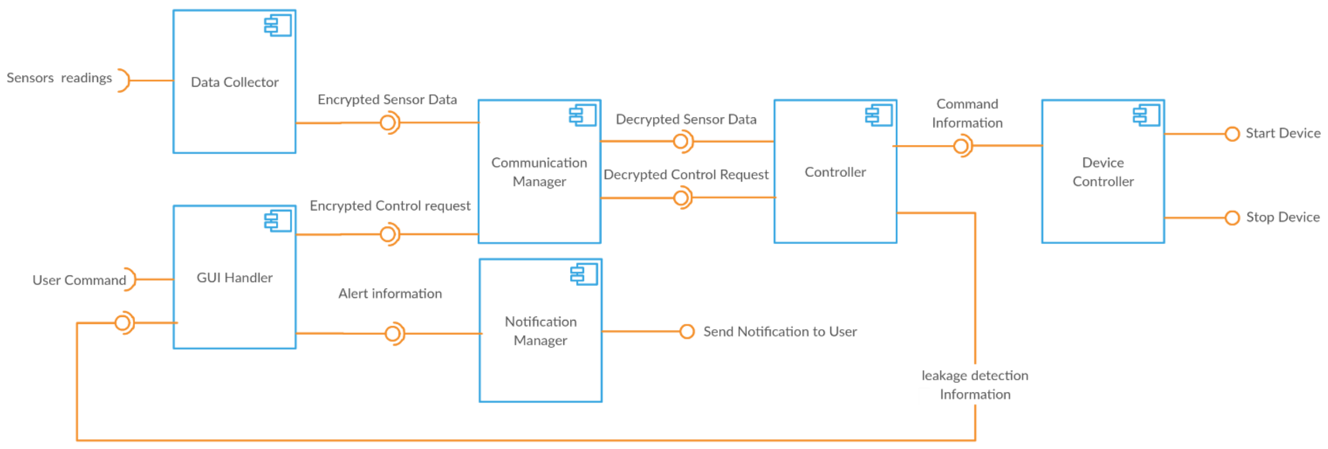
4.6.1 User Commands Sub-System



4.6.2 Monitoring Sub-System

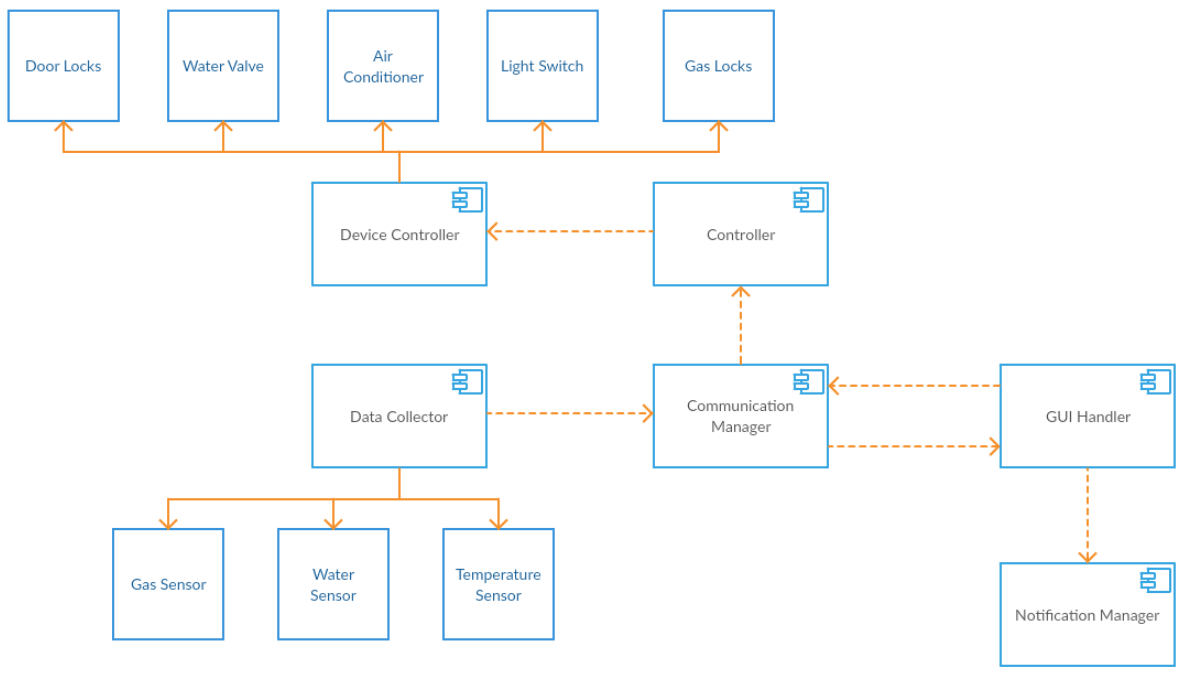


4.7 Component Model



5.0 ARCHITECTURE DESIGN

5.1 Design Model



5.2 Decisions Taken

1. **Issue:** The programing language.
   Alternatives: Java, C++, C# or JavaScript.
   Decision: JavaScript.
   Rationale: In order to use (ReactJS) library which is a cross platform library, contains application generation capabilities for (iOS, Android and Web), Thus, the user can access his home devices using any platform. It will also decrease the project total cost, as we won't need native developers for each platform.
   Status: approved.

2. **Issue:** The sensors, devices and mobile phone connection.
   Alternatives: Wi-Fi or Bluetooth.
   Decision: All the devices and sensors will be connected via Wi-Fi.
   Rationale: We preferred using Wi-Fi over Bluetooth, in order to access your home devices from anywhere outside your home and also to increase the speed of communication between devices and controller, thus increasing system performance.
   Status: approved.

5.3 Stakeholders

1. Architect	2. System Designer	3. Developers
4. Requirements engineer	5. Tester and integrator	6. Maintainer
7. System Analyst	8. Quality assurance people	9. Project Manager

6.0 VIEW MODEL

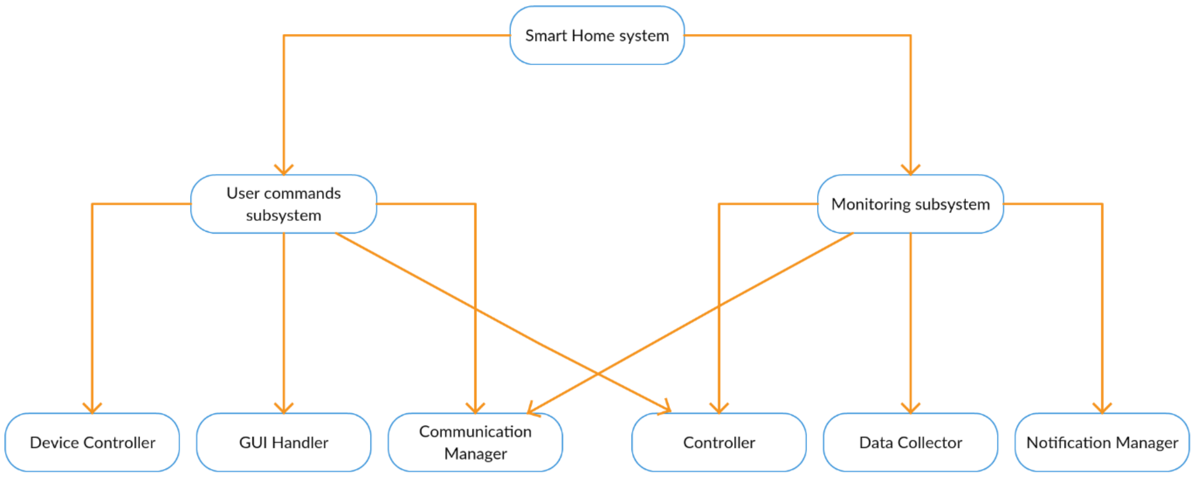
6.1 Module View

1. **Decomposition**

   Smart home system consists mainly of 2 subsystems.

   - User Commands subsystem.
   - monitoring subsystem.

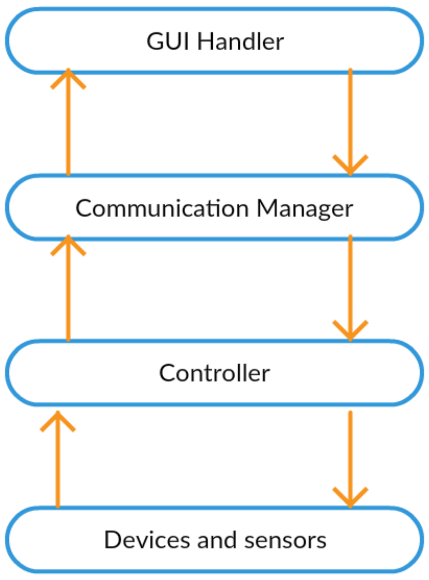
   Device controller and GUI Handler are mainly submodules of user commands subsystem. Data collector and the notification manager are mainly submodules of monitoring subsystem. Controller, and Communication manager are related to both submodules.



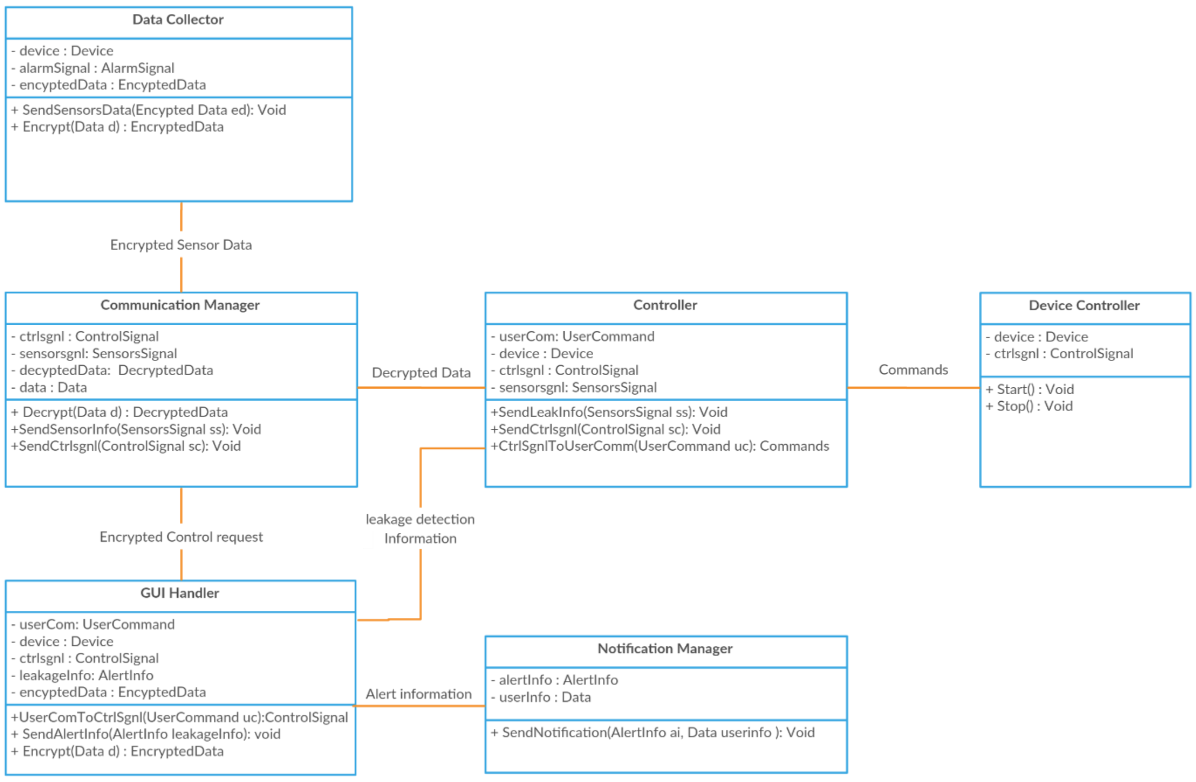
2. **Uses**

   - Data collector uses the communication manager to communicate with the Controller.
   - Controller uses the communication manager to get sensors data from the data collector.
   - Controller uses the GUI handler to send leakage information to the user.
   - GUI handler uses the Notification manager to send the alert information to the user.
   - GUI handler uses the communication manager to communicate with the controller (send the user commands).
   - Device Controller uses the Controller to get commands and set the devices.

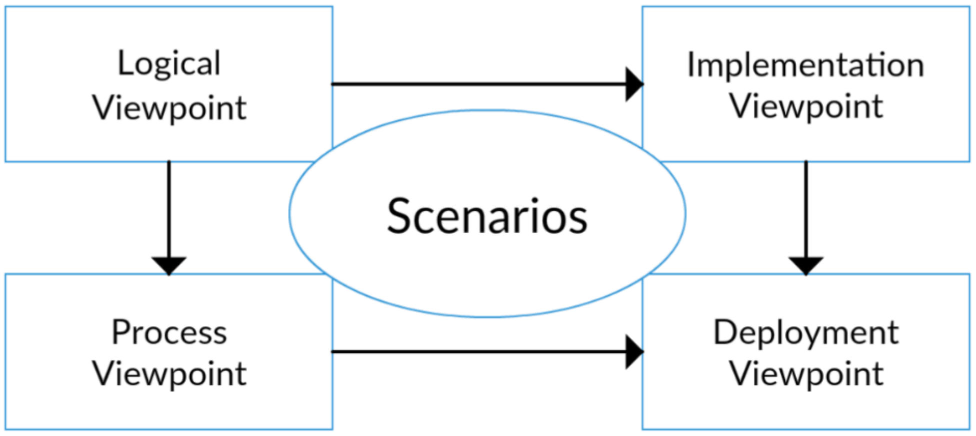
3. Layers



4. Classes


Data Collector
- device : Device
- alarmSignal : AlarmSignal
- encyptedData : EncyptedData
+ SendSensorsData(Encypted Data ed): Void
+ Encrypt(Data d) : EncryptedData
Encrypted Sensor Data
Communication Manager
- ctrlsgnl : ControlSignal
- sensorsgnl: SensorsSignal
- decyptedData: DecryptedData
- data : Data
+ Decrypt(Data d) : DecryptedData
+SendSensorInfo(SensorsSignal ss): Void
+SendCtrlsgnl(ControlSignal sc): Void
Decrypted Data
Controller
- userCom: UserCommand
- device : Device
- ctrlsgnl : ControlSignal
- sensorsgnl: SensorsSignal
+SendLeakInfo(SensorsSignal ss): Void
+SendCtrlsgnl(ControlSignal sc): Void
+CtrlSgnlToUserComm(UserCommand uc): Commands
Commands
Device Controller
- device : Device
- ctrlsgnl : ControlSignal
+ Start() : Void
+ Stop() : Void
Encrypted Control request
leakage detection
Information
GUI Handler
- userCom: UserCommand
- device : Device
- ctrlsgnl : ControlSignal
- leakageInfo: AlertInfo
- encryptedData : EncryptedData
+UserComToCtrlSgnl(UserCommand uc):ControlSignal
+ SendAlertInfo(AlertInfo leakageInfo): void
+ Encrypt(Data d) : EncryptedData
Alert information
Notification Manager
- alertInfo : AlertInfo
- userInfo : Data
+ SendNotification(AlertInfo ai, Data userinfo): Void

6.2 View Model



6.2.1 Logical Viewpoint

The system provides to its end users a home control services, where user can:

- switch on/off the gas locks.
- switch on/off the selected light switch.
- lock/unlock the selected door.
- adjust temperature for the specified room.

The system also provides a monitoring service, where:

- Whenever the gas sensors detect a gas leakage, the system automatically closes the gas lock and send a notification to the user, and if the system fails to close the gas lock, it automatically reports a gas leak to New Jersey Natural Gas with the location of the leak and alert the user.
- Whenever the water sensors detect a water leakage, the system close the water valve that caused the leak and notify the user, and if the system fails to close the water valve, it automatically reports a water leak to the nearest water station with the location of the leak and alert the user.
- if the temperature sensor of any room detected a change in temperature, the system adjusts the air conditioner to match the selected temperature.

6.2.2 Process Viewpoint

Should deploy at least two gas and water sensors to detect gas and water leaks to guarantee that is no fault.

The system should have fast response time to handle user commands, it must not exceed five seconds. each service must run in different thread in order to work concurrently and optimize the whole system performance, controller should run on its own thread in order to ensure the system availability.

The system throughput must be high as the user may do many commands successively, the system as well must save all the users data and actions.

6.2.3 Implementation Viewpoint

- Data Collector
- Communication Manager
- Controller
- Device Controller
- GUI Handler
- Notification Manager

These are the components of the smart home system; the data Collector component works as intermediate between sensors and the controller as it gets the sensors' readings and checks whether the system state is safe or not.

If it's unsafe it sends the collected data to the communication manager in an encrypted form, the communication manager should send it to the controller, also the Communication manager is responsible for decrypting and delivering the user commands to the Controller.

Device Controller component gets the control signal generated through the Controller component and apply it to the specified device with this control signal.

GUI Handler component takes the user commands and send it to the controller through the communication manager. It also displays the alert notifications by receiving this notification data from the Notification Manager in order to deliver it to the user's phone.

6.2.4 Deployment Viewpoint

To start working with the smart home system the user needs a mobile phone with an IOS or Android operating systems and a secure online connection which allows the user to control the system from anywhere over the world.

The user can send a control signal while he is connected to the internet, the signal is transferred to the user’s house, ideally the system should have a fast response time to handle the user command, it must not exceed five seconds.

It’s observed that if the number of devices connected to the network exceeded 25 devices, the response time increases to six seconds.

The system is always ready to get the user commands’ signals unless the phone is disconnected from the internet.

The online connection allows the user to get alert notifications of any leakage whenever the user is connected to the internet and if the notification is not delivered which means the user is not connected to the internet the system re-send the notification data as a SMS to the user.

The system saves the users activities and the system warnings to a disk with a limited space of 100 MB, once its full the system archive the stored information and reset the storage.

6.2.5 Scenario

There are two scenarios in the smart home system:

- Firstly, the User Commands system scenario where the user chooses from the user interface whether to switch on/off the gas locks, switch on/off the selected light switch, lock/unlock the selected door, or set the adjusted temperature for the specified room. The controller controls the selected device and display the results of the modification to the user throw the user interface.

- Secondly, the monitoring system scenario where the sensors send a signal to the Data collector with its readings and the Data collector checks whether it’s safe or not, if it’s not safe it sends the sensors data encrypted to the controller to take actions towards the unsafe device and sends a notification to the user with the alert information.

7.0 ARCHITECTURE STYLE

IMPLICIT INVOCATION:

Problem:

1. Need to handle sensors events.
    - Gas sensor signals for a gas leak; the system attempts to close the gas lock, if the system fails, an emergency is reported, and the user is notified.
    - Array of water sensors signal for water leak at a specific water pipe; the system closes the corresponding water valve.
    - Temperature sensor in each room signals for a temperature change with respect to the user selected temperature, the system adjusts the corresponding air condition to restore the temperature.

2. Need to handle user events.
    - The user can request to lock or unlock a specified door, the system locks or unlocks the specified door.
    - The user can request to lock or unlock the gas lock, the system locks or unlocks the gas lock.
    - The user can request to switch on or off a specified light bulb, the system switches on or off the specified light bulb.
    - The user can request to set the temperature of a specified room, the system sets the temperature of the specified room.

Context:

1. Requires an embedded system with real-time computing constraints and event handling compatibility.
2. Requires gas sensors, water sensors and temperature sensors.
3. Requires electrically controllable door locks, water valves, gas locks.
4. Requires special air conditioners that supports system's controls.
5. Requires mobile operating system that offers online notification and request handlers.

Solution:

- **System model:** independent, reactive processes, invoked when an event is raised.
- **Components:** processes that signal events and react to events.
- **Connectors:** automatic invocation.
- **control structure:** decentralized control. Components do not know who is going to react.

Variants:

- Door locks, water valves, gas locks, air conditioners interfaces.
- Language with special features (Java Script); as it contains React framework which is capable of generating cross platform application.

8.0 COMPONENTS

8.1 Components Usability and Reusability

In this system, we have 6 components:

1. **Data Collector Component.**
   - **Reusable:** The component is generalized as it has general method names, it can deal with different sensors and it can handle exceptions consistently through component interface. It's also associated with stable domain abstraction, as it will always be responsible for collecting sensors' data and checking whether it's in safe range or not.

2. **Communication Manager Component.**
   - **Reusable:** The component is generalized as it has general method names, it can deal with different encrypted data and it can handle exceptions consistently through component interface. It's also associated with stable domain abstraction, as it will always be responsible for decrypting sensors' data and decrypting user commands, thus, responsible for security level of traffic.

3. **Controller Component.**
   - **Usable:** The component is not reusable or generalized as it's directed towards specific control functionalities, with application-specific method names and attributes, so, it can't be used outside this application.

4. **Device Controller Component.**
   - **Reusable:** The component is generalized as it has general method names, it can deal with different devices and it can handle exceptions consistently through component interface. It's also associated with stable domain abstraction, as it will always be responsible for executing given commands on the specified devices.

5. **GUI Handler Component.**
   - **Usable:** The component is not reusable or generalized as it's directed towards specific functionalities, with application-specific method names and attributes, as it's designed for handling the GUI of this application only, so it can only deal with "Controller Component" specified in this application.

6. **Notification Manager Component.**
    - **Reusable:** The component is generalized as it has general method names and attributes, it can manage any application notifications and it can handle exceptions consistently through component interface. It's also associated with stable domain abstraction, as it will always be responsible for notifying the user of any alert.

8.2 Components Characteristics

All the 6 components are having the following characteristics:

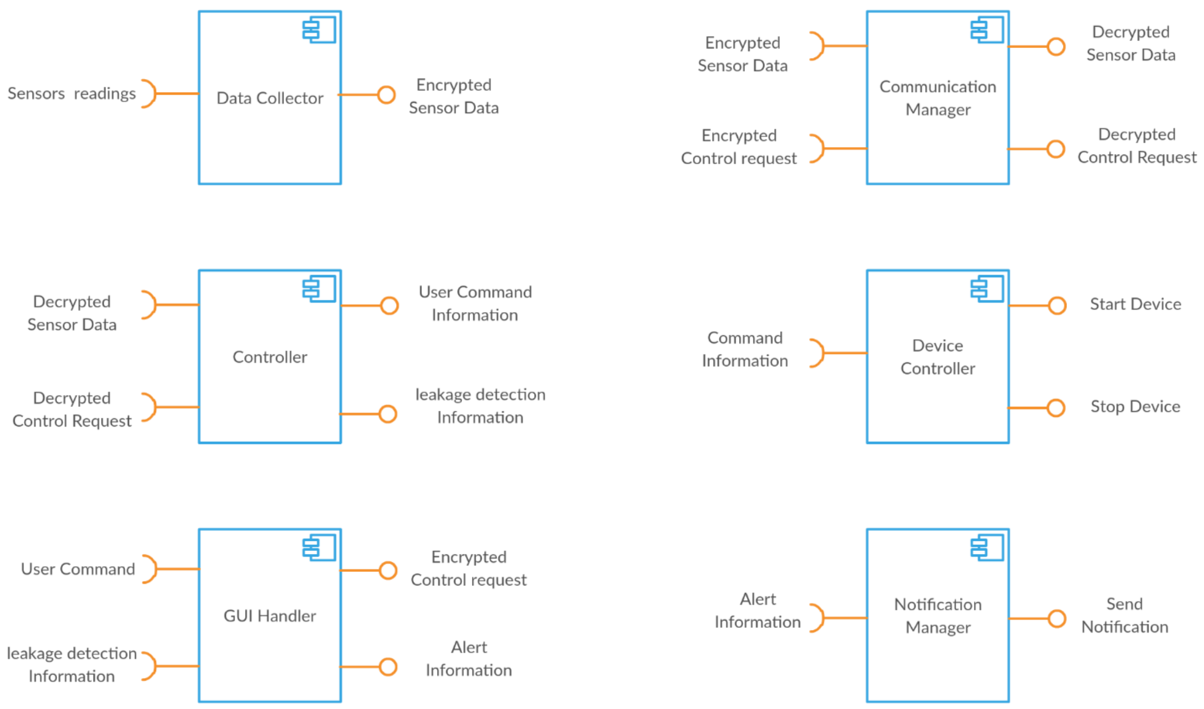
- **Standardized:** The component has been conformed to some standardized component model. The component has definite interfaces, documentation, composition and deployment.

- **Independent:** This component is composable and deployable without having to use other specific components. All the needed externally provided services are explicitly set out in a 'requires' interface specification.

- **Composable:** All external interactions take place through publicly defined interfaces. In addition, it provides external access to information about itself such as its methods and attributes.

- **Deployable:** The component is a self-contained component and its able to operate as a stand-alone entity on some component platform that implements the component model. It doesn't need to be compiled before it is deployed.

8.3 Components Interfaces



1. **Data Collector Component.**
   - **Requires:** Sensors readings.
   - **Provides:** Encrypted Sensors Data.
   - **Process:** Collects the sensors readings from all monitoring sensors, checks whether it’s in the safe range or not, if it’s unsafe the data is sent encrypted to the successive component.

2. **Communication Manager Component.**
   - **Requires:** Encrypted Sensors data, Encrypted Control request.
   - **Provides:** Decrypted Sensors data, Decrypted Control request.
   - **Process:** Decrypts the system’s encrypted inputs (provided sensors data and user commands) and send the data to process it to the successive component.

3. **Controller Component.**
   - **Requires:** Decrypted Sensors data, Decrypted Control request.
   - **Provides:** Command information, Leakage detection information.
   - **Process:** Sends commands to the devices whether came from the user or from an unsafe state, also it sends the leakage information which led to the unsafe state to the successive component.

4. **Device Controller Component.**
    - **Requires:** Command information.
    - **Provides:** Start device, stop device.
    - **Process:** Start/Stop the device which is specified by the controller component.

5. **GUI Handler Component.**
    - **Requires:** User command, Leakage detection information.
    - **Provides:** Encrypted Control request, Alert information.
    - **Process:** Sends the user commands encrypted to be done, also sends the alert information which came from the controller to the successive component.

6. **Notification Manager Component.**
    - **Requires:** Alert information.
    - **Provides:** Send notification.
    - **Process:** Sends the alert information as a notification to the specified user.

8.4 Components Internals Hiding

The Controller, Device Controller, GUI Handler, Notification Manager, Communication manager are considered as Black box components, the specification only is known at implementation, the input and output forms only are known. Data Collector component is considered as grey box component, as the unsafe range is modified.

8.5 Process of Component Development

Based on the functional requirements it's observed that the system needs a component that collects the data from the sensors and checks its range of safety, a component that decrypts and transfer signals, a component that uses a given signal to do a specific action, a component that deliver the system information to the user in a notification form, a user interface component which sends a command signals and alert information, finally a controller component to send the commands signals and alert information.

- ➢ In order to reduce the time taken to build the project, we decided to buy some reusable components.

1. Data Collector Component.

- **Find**
  A component which gets sensors data and checks whether the readings are in the safe range specified or not, if the readings are un safe it provides this information to another component.
- **Select**
  The selected component is the component with the fastest response time when unsafe state occurs.
- **Verify**
  The selected component is tested individually by entering random numbers and record its actions when a number out of the safe range is entered, also calculate the response time.
- **Integration**
  The component is integrated with other components. Unsafe state was injected to the system the component sends the readings to the successive components without a need to an adaptor successfully in the expected response time.
- **Testing**
  The component is assembled with the other selected components, the system worked as expected in a fast response time.
  Its observed that if two unsafe readings are recorded successively within time 5 seconds the component will discard the second one.
- **Store**
  The selected Data collector component is added in a component repository ready for being used. All it's tests results and problems had also been saved.

2. **Communication Manager Component.**
   - **Find**
     Find a component which gets an encrypted signal, decrypt it and deliver it to another component.
   - **Select**
     The selected component is the component which can decrypt the existed encryption algorithm with the fastest response time.
   - **Verify**
     The selected component is tested individually by entering a known data encrypted to it and record the results whether it's right or not, the response time is calculated.
   - **Integration**
     The component is integrated with other components. An encrypted signal with a specific user command or sensor data is given without the need of using an adaptor and the output is the expected decrypted output in the expected response time.
   - **Testing**
     The component is assembled with the other selected components, the system worked as expected in a fast response time.
     Its observed that if two commands or sensors readings are entered successively within time 5 seconds the component will discard the second one.
   - **Store**
     The selected Communication manager component is added in a component repository ready for being used.
     All it's tests results and problems had also been saved.

3. **Notification Manager Component.**
   - **Find**
     Find a component which gets the alert info from the system and sends it as a notification to the user.
   - **Select**
     The selected component is the component with the fastest response time.
   - **Verify**
     This component is tested individually by entering alert information and checks whether it's sent to the user or not, the response time is calculated.
   - **Integration**
     The component is integrated with other components. Numbers are entered to the data collector component which leads to an unsafe, the component sent a notification to the specified user successfully in the expected response time.

- **Testing**
  The component is assembled with the other selected components, the system worked as expected in a fast response time.
- **Store**
  The selected Notification Manager Component is added in a component repository ready for being used.
  All it's tests results and problems had also been saved.

4. **Device Controller Component.**
   - **Find**
     Find a component which deliver the commands to the specified device.
   - **Select**
     The selected component is the component which delivers the command to the specified device in a fast response time.
   - **Verify**
     The selected component is tested individually by injecting a command and see whether it's done or not, the response time is calculated.
   - **Integration**
     The component is integrated with other components. Actual command is given without the need of using an adaptor, and the command happened successfully in the response time range.
   - **Testing**
     The component is assembled with the other selected components, the system worked as expected in a fast response time.
     Its observed that if two commands are entered successively within time 5 seconds the component will discard the second one.
   - **Store**
     The selected Device Controller Component is added in a component repository ready for being used.
     All it's tests results and problems had also been saved.

- In order to increase the efficiency of the project, we decided to implement some usable components.

5. **Controller Component.**
   - **Find**
     Find a component which processes the decrypted user commands or sensors data to control a specific device and send information.
   - **Design**
     The component is designed with considering the standards, it's functional requirements, and nonfunctional requirements of the system.
   - **Implementation**
     The design is implemented using JavaScript to ensure usability, and also to ensure achieving the functional requirements.
   - **Verify**
     The selected component is tested individually by entering a decrypted command or data and checks whether it's done/delivered or not, the response time is calculated.
   - **Integration**
     The component is integrated with other components. Actual user command is given and an unsafe sensor reading as well without the need of using an adaptor, the requirements are achieved successfully in the response time range.
   - **Testing**
     The component is assembled with the other selected components, the system worked as expected in a fast response time.
     Its observed that if two commands or 2 sensors readings are entered successively within time 5 seconds the component will discard the second one. But it's fine to deal with a user commands and sensor data in the same time.
   - **Store**
     The selected Device Controller Component is added in a component repository ready for being used.
     All it's tests results and problems had also been saved.

6. **GUI Handler Component.**

- **Find**
  Find a component which takes the user command and send it encrypted and also take any system info and send this information to another component.
- **Design**
  The component is designed with considering the standards, it's functional requirements, and nonfunctional requirements of the system.
- **Implementation**
  The design is implemented using JavaScript to ensure usability, and also to ensure achieving the functional requirements.
- **Verify**
  The selected component is tested individually by entering a user command or data and checks whether user command is decrypted and sent or not, also checks whether the information is delivered or not, the response time is calculated.
- **Integration**
  The component is integrated with other components. Actual user command is given and an unsafe sensor reading as well without the need of using an adaptor, the command is done, and the information is delivered, then requirements are achieved successfully in the response time range.
- **Testing**
  The component is assembled with the other selected components, the system worked as expected in a fast response time.
- **Store**
  The selected Device Controller Component is added in a component repository ready for being used. All it's tests results and problems had also been saved.

9.0 TYPES OF COMPOSITION

Sequential composition where the composed components are executed in sequence. In the smart home system there is 2 sequences.

1. Firstly, when user command is given to the GUI handler component, GUI handler provides the encrypted control request to the Communication manager component where decryption occurs in it and the data is provided to the Controller component. finally, the controller provides the user command information to the device controller component to control the specified device.

2. Secondly, when sensors provide unsafe readings to the Data collector component, it provides the encrypted sensors data to the Communication manager component where decryption occurs in it and the data is provided to the Controller component. Controller provides the detected leakage information to the GUI handler component, GUI send the Alert information to the Notification component to be sent after that to the user.

10.0 FUNCTIONAL POINTS

Complexity Factor: Fi	**0**	**1**	**2**	**3**	**4**	**5**	**Fi**
Backup and recovery	0	0	1	0	0	0	**2**
Data communication	0	0	1	0	0	0	**2**
Distributed processing function	0	0	0	0	0	1	**5**
Is performance critical?	0	0	0	0	1	0	**4**
Existing operating environment	0	0	0	0	0	1	**5**
On-line data entry	0	0	1	0	0	0	**2**
Input transaction built over multiple screens	0	1	0	0	0	0	**1**
Master files updated on-line	0	0	0	0	0	1	**5**
Complexity of inputs, outputs, inquiries, files	0	0	0	1	0	0	**3**
Complexity of processing	0	0	1	0	0	0	**2**
Code design for reuse	0	0	0	0	1	0	**4**
Are conversion/installing included in the design	0	0	0	0	0	1	**5**
Multiple installation	0	0	0	0	1	0	**4**
Application designed to facilitate change by the user	1	0	0	0	0	0	**0**
						$\sum F$	**44**
Complexity Adjustment Factor:	**0.65 + 0.01 * $\sum F$**						**1.09**

Parameter	**Count**	**Simple**	**Average**	**Complex**	**Total**
Number of user inputs	4	6	7	8	28
Number of user outputs	7	3	5	6	35
Number of inquiries	7	2	5	8	35
Number of files	2	5	6	7	12
Number of external interfaces	8	7	9	11	72
Unadjusted Function Count					182
		Complexity Adjustment Factor			1.09
				FP	**198**

11.0 COST ESTIMATION

There are many ways and methods to estimate the cost of the project, and surly all these methods should return approximately near result.

Software costs estimation is difficult, as what appears to be a simple problem on the surface is much harder or technically challenging to implement in reality.

The following 4 methods will work on estimating the smart home system cost.

11.1 Functional Points Method

Assuming that:

1. Estimated FP = 198
2. Organization average productivity = 6 FP/pm
3. Burdened labour rate = 2500$

Then:

1. Estimated effort = FP/Organizational Productivity = 198/6 = 33 pm
2. Cost per FP = Labour/Organizational Productivity = 2500/6 = 416.67 $/FP
3. Project Cost = Labour * Effort = 2500 * 33 = 82,500$

11.2 LOC Method

Functions	Estimated LOC	LOC/pm	$/LOC	Cost	Effort
Leak detection	600	300	13	7800	2
Alert	150	110	15	2250	2
Report Emergency	100	70	20	2000	2
Execute command	300	200	11	3300	2
Mobile GUI	8000	550	10	80000	15
Total	9150			95350	23

Assuming that:

1. Estimated LOC = 9150
2. Organization average productivity = 246 LOC/pm
3. Burdened labour rate = 2500$

Then:

1. Estimated effort = LOC/Organizational Productivity = 9150/246 = 38 pm
2. Cost per FP = Labour/Organizational Productivity = 2500/246 = 10.2 $/LOC
3. Project Cost = Labour * Effort = 2500 * 38 = 95,000$

11.3 Expert Judgement

One of the hardest things to do in software development is to determine how long and how much it will take to deliver a new software product. So, the company decided to cooperate with an expert to estimate the application cost. And based on his previous experience he estimated that the application would cost 90,000$.

11.4 Estimation by Analogy

The cost of the project is computed by comparing the project to a similar project in the same application domain. And based on this analogy, the estimated application cost would be 85,000$.