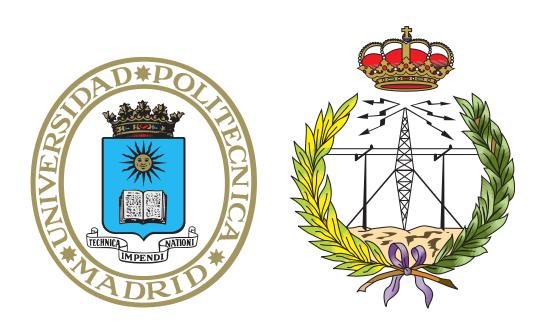
TECHNICAL UNIVERSITY OF MADRID SCHOOL OF TELECOMMUNICATIONS SYSTEMS AND ENGINEERING

Semester Project



Bc. et Bc. Jaroslav Svoboda Jaime Sancho Diallo Elhadj Sadou

Smart Lighting

UBIQUITOUS AND SECURE NETWORKS AND SERVICES

Lecturer: Jesus Rodriguez Molina

Subject coordinator: Jose Fernan Martinez Ortega

Degree programme: Master in Systems and Services Engineering for the Information society

Contents

| 1 | Introduction and Executive Summary | 2 | | | |
|---|---|--------|--|--|--|
| 2 | 2 Scheduling of workload | | | | |
| 3 | Requirement Analysis 3.1 General use | 5 6 | | | |
| 4 | Project Design 4.1 UML Component Diagrams | 9 | | | |
| 5 | Testing | 13 | | | |
| 6 | Conclusion | 14 | | | |

1. Introduction and Executive Summary

2. Scheduling of workload

3. Requirement Analysis

3.1 General use

Figure 3.1: General use

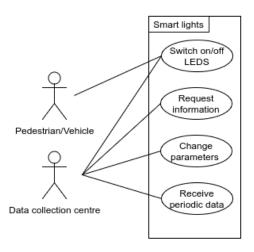


Table 3.1: General use

| Scope and Objectives | The lights on the lampposts switch on/off depending on the information of the movement and light sensor of the motes (if the light is low and the movement sensor detects something, the lights turn on, otherwise they turn off). The data collection centre receives periodically this data and it is capable to change some parameters or request some information of the motes. |
|----------------------|---|
| Actors | Pedestrians/vehiclesData control department of the city |
| Preconditions | The sensor nodes are on A pedestrian/vehicle comes/goes away from the node |
| Post-conditions | The lights will turn on/off All the measured information will be sent to the city data manager |
| Sequence Description | 1. Obtain measurements of movement, temperature, humidity and light. |
| | 2. If there is a change in the movement sensor and the light measured is low, toggle the lights on the node. |
| | 3. Send the status of light to the data control department of the city. |
| Exceptions | The node is not able to communicate with the data control department of the city. |

3.2 Receive Periodic Data

Figure 3.2: Receive Periodic Data

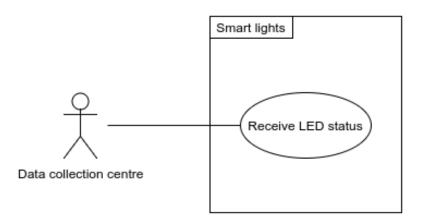


Table 3.2: Receive Periodic Data

| Scope and Objectives | The motes are placed in the lampposts of the street getting information of the movement and the light. This information added to the LED status of the lamppost is sent to the data collection centre periodically. |
|----------------------|---|
| Actors | Data control department of the city |
| Preconditions | The sensor nodes are on The sensor nodes have sufficient memory to store measured values when connection is off |
| Post-conditions | The measured value is stored in memory The measured value is sent to the data control department of the city |
| Sequence Description | Obtain measurement Check the network If network not available store data Send all data to the data control department of the city |
| Exceptions | In case of error, report to the data control department of the city |

3.3 Request data

Figure 3.3: Request Data

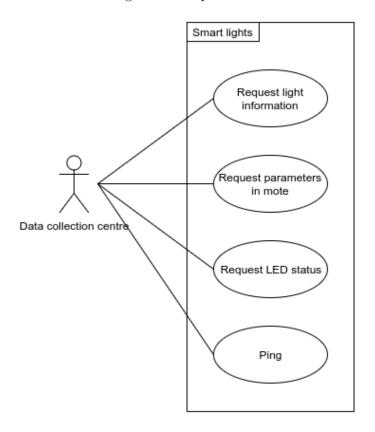


Table 3.3: Request Data

| Scope and Objectives | The data collection centre have the possibility to request the information of the light and LED status of the motes and also the parameters that are using the motes. Furthermore, a ping functionality exists to check if the mote is alive. |
|----------------------|---|
| Actors | Data control department of the city |
| Preconditions | The sensor nodes are on The sensor node is online to get the request |
| Post-conditions | The measured value and statuses are sent to the data control department of the city |
| Sequence Description | Check if node is online If not then repeat x times Request current data from node |
| Exceptions | The sensor node does not respond after x requests |

3.4 Change parameters

Figure 3.4: Change parameters

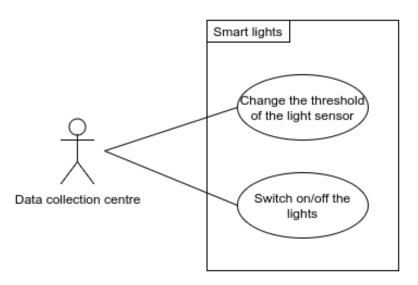
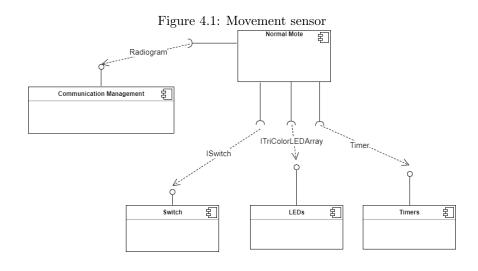


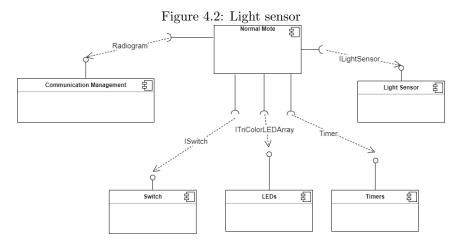
Table 3.4: Change parameters

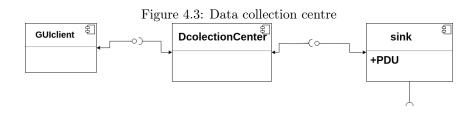
| Scope and Objectives | The data collection center is able to change the threshold of the light sensor and also has the possibility of switch on/off the lights in case of emergency. |
|----------------------|---|
| Actors | Data control department of the city |
| Preconditions | The sensor nodes are on The sensor node is online to get the request Led light is working |
| Post-conditions | The light is turned on/off Parameters are modified The status is sent to the data control department of the city |
| Sequence Description | Check if node is online If not then repeat x times Request switching on/off the light or change of parameters Send ack when light turns on or parameters are changed |
| Exceptions | The sensor node does not respond after x requests The light does not turn on/off |

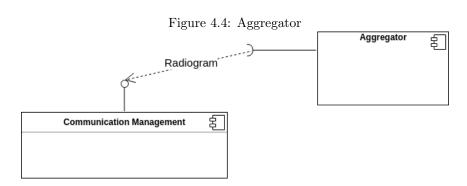
4. Project Design

4.1 UML Component Diagrams









4.2 UML Class Diagrams

Connector + startApp() : protected void + pauseApp() : protected void + destroyApp() : protected void + open(address : String) + packetsAvailable() : boolean + receive(datagram : Radiogram) + send(datagram : Radiogram) NormalSensor NormalSensor

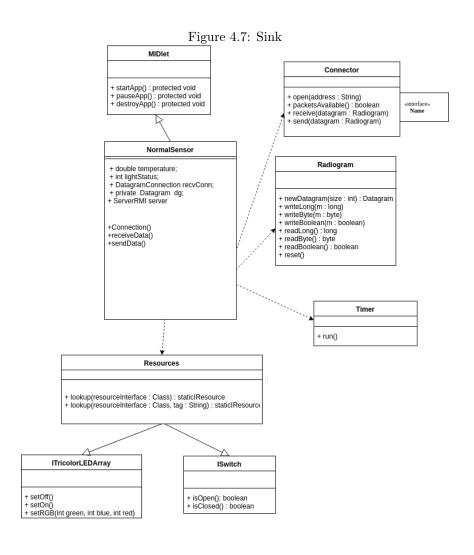
+ sw1 : Iswitch
+ movement : boolean
+ movement eriod : int
+ leds : ITricolorLEDArray
+ LEDStatus : boolean
+ prevStatus : boolean
+ lightSensor : ILightSensor
+ light : boolean
+ lightPeriod : int
+ lightPeriod : int
+ threshold : int
+ connAgg : RadiogramConnection
+ connLight : RadiogramConnection
+ xAgg : Radiogram
+ rAgg : Radiogram
+ rAgg : Radiogram
+ IAgg : Radiogram
+ IAgg : Radiogram
+ IAgg : Radiogram
+ IAgg : Radiogram
+ BROADCAST : String
+ ADDRESSAGGREGATING: String
+ timerLight : Timer
+ timerMovement : Timer Radiogram + newDatagram(size : int) : Datagram + writeLong(m : long) + writeByte(m : byte) + writeBoolean(m : boolean) + readLong() : long + readByte() : byte + readBoolean() : boolean + reset() Timer + run() + cancel() + runTimerMovement(period : int) + stopTimer() + getLight() + LEDManager() + commandsManager() + lookup(resourceInterface : Class) : staticIResource + lookup(resourceInterface : Class, tag : String) : staticIResource ITricolorLEDArray ISwitch + setOff() + setOn() + setRGB(int green, int blue, int red) + isOpen(): boolean + isClosed() : boolean

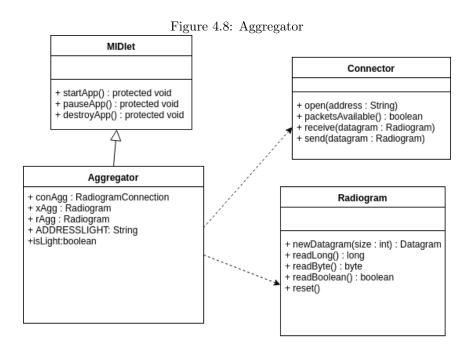
Figure 4.5: Movement sensor

MIDlet Connector + startApp() : protected void + pauseApp() : protected void + destroyApp() : protected void + open(address : String) + packetsAvailable() : boolean + receive(datagram : Radiogram) + send(datagram : Radiogram) LightSensor LightSensor

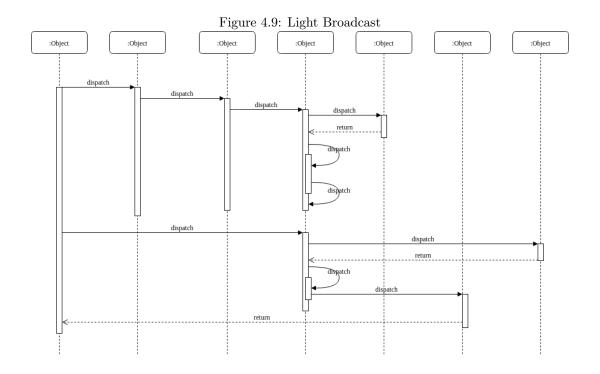
+ sw1 : Iswitch
+ movement : boolean
+ movementPeriod : int
+ leds : ITricolort.EDArray
+ LEDStatus : boolean
+ prevStatus : boolean
+ lightSensor : ILightSensor
+ light : boolean
+ light beolean
+ lightPeriod : int
+ threshold : int
+ conAgg : RadiogramConnection
+ xAgg : Radiogram
+ rAgg : Radiogram
+ ADDRESSAGGREGATING: String
+ timerLight : Timer
+ timerMovement : Timer Radiogram + newDatagram(size : int) : Datagram + writeLong(m : long) + writeByte(m : byte) + writeBoolean(m : boolean) + readLong() : long + readByte() : byte + readBoolean() : boolean + reset() Timer + runTimerMovement(period : int) + stopTimer() + getLight() + LEDManager() + commandsManager() + run() +cancel Resources + lookup(resourceInterface : Class) : staticIResource + lookup(resourceInterface : Class, tag : String) : staticIResource ITricolorLEDArray ISwitch lLight + setOff() + setOn() + setRGB(int green, int blue, int red) + isOpen(): boolean + isClosed() : boolean + getAverageValue() : int

Figure 4.6: Light sensor





4.3 Sequence Diagrams



5. Testing

6. Conclusion