**SE-2 Course Project (PhasE 2 COVER sHEET)**

**Discussions Scheduled for Week 12** *(Specific dates TBA by the TAs).*

* Print 1 copy of this cover sheet and attach it to a printed copy of the documentation *(SRS, … etc.)*. You must submit softcopies of all your documents *(as PDFs)*; details will be announced later.
* Please write all your names in Arabic.
* Please make sure that your students’ IDs are correct.
* Handwritten Signatures for the attendance of all team members should be filled in before the discussion.
* Please attend the discussion on time *(announced separately)*, late teams will lose 3 grades.

**Project Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Team Information *(typed not handwritten, except for the attendance signature)*:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **ID**  **[Ordered by ID]** | **Full Name**  **[In Arabic]** | **Attendance**  **[Handwritten Signature]** | **Final Grade** |
| **1** |  |  |  |  |
| **2** |  |  |  |  |
| **3** |  |  |  |  |
| **4** |  |  |  |  |
| **5** |  |  |  |  |
| **6** |  |  |  |  |

**Grading Criteria:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Items** | | **Grade** | **Notes** |
| **Functional Requirements & Non-Functional Requirements –** *including any updates, and all timing constraints.* | **1** |  |  |
| **Bonus: System Architecture –** *including any applied Architectural Pattern(s).* | **1** |  |  |
| **Use-Case Diagram(s)****–** *including all use-cases for the system, and the detailed use-cases description, and any alternative scenarios.* | **1** |  |  |
| **Sequence Diagram(s) –** *including varying fragments, interaction references/gates, different types of messages & constraints, .. etc.* | **2** |  |  |
| **Collaboration/Communication Diagram(s) –** *including different types of messages, and the objects must have stereotypes indicating their categories based on the given class/object structuring criteria.* | **1** |  |  |
| **State-Machine Diagrams –** *for all state dependent objects, and for the entire system too, including “when necessary” Events/Actions, Guards, Entry and Exit events/actions, Composite and Orthogonal states, Submachines, History States.. etc.* | **2** |  |  |
| **Bonus: Object Diagrams –** *including object diagrams that illustrate the preconditions and the post-conditions of selected functions.* | **1** |  |  |
| **Bonus: 2 Design Patterns Applied –** *Including a typed description of the pattern and how is it applied.* | **1** |  |  |
| **Detailed Class diagram –** *including “when necessary” Classes, Attributes & Methods, Interfaces & Abstract Classes, Associations / Aggregations / Generalizations / Association Classes / Qualified Associations, Constraints - including also the categories of the classes based on the given class/object structuring criteria, and stereotypes indicating the type/category of each class. All necessary types/categories should be modelled.* | **2** |  |  |
| **Stimuli/Response Identification (State Transition Table)** | **1** |  |  |
| **Implementation & discussion. Marking the code will be based on the following criteria:**  **1) Requirements are fulfilled.**  **2) Correctly mapping design models into executable code.**  **3) Running correctly.**  **4) Detailed Testing.**  **5) Correct multithreading implementation and synchronization.** | **5** |  |  |

**N.B. I** .. **You must update and resubmit the initial part of the documentation submitted in phase 1** (including the Functional / Non-Functional requirements, Use-case Diagrams & Descriptions, Activity Diagrams, Interaction Diagrams, Object Diagrams, .. etc.).

**15**

**Teaching-Assistant’s Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Table of contents :

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 general description for the project…………………………………………………

1.2 stack holders ……………………………………………………………………………………………………………..

2. functional reqirements 2

2.1 User functional reqirements 2

2.2 System functional reqirements 2

3. Non functional Requirements 3

3.1 look and feel 3

3.2 usability&humantiy 3

3.3 performance 3

3.4 maintainbility 3

3.5 cultural……………………………………………………………………………………..3

3.6 legal ……………………………………………………………………………………3 3.7 security…………………………………………………………………………………………………….3

4. Diagrams 4

4.1 usecase diagram 4

4.2 usecase description 4

4.1 class diagram 4

4.2 activity diagram 4

Sequence diagram …………………………………………………………………..

collaboration diagram……………………………………………………………………………….state machine------------------------------------------------------------------------

design pattern

**A Burglar Alarm System**

**A Burglar alarm system is a system designed to detect intrusion – unauthorized entry – into a building or area. Security**

**alarms are used in residential, commercial, industrial, and military properties for protection against burglary (theft) or**

**property damage, as well as personal protection against intruders. The Burglar alarm system uses several different types**

**of sensors. These include movement detectors in individual rooms, door sensors that detect corridor doors opening, and**

**window sensors on ground floor windows that can detect when a window has been opened.**

**• When a sensor detects the presence of an intruder, the system automatically calls the local police and, using a voice**

**synthesizer reportsthe location of the alarm. Itswitches on lights in the rooms around the active sensor andsets off**

**an audible alarm. Each movement detector should be polled twice per second.**

**• The sensor is normally powered by mains power but is equipped with a battery backup. Power loss is detected**

**using a separate power circuit monitor that monitors the main voltage. If a voltage drop is detected, the system**

**assumes that intruders have interrupted the power supply, so an alarm is raised. The alarm system includes a**

**display, which reports these anomalies upon occurrence.**

**• When single or more sensors are positive, which means that an intruder is detected, the system initiates alarm**

**within half a second of an alarm being raised by a sensor and turns on lights around the site of the positive sensor**

**within half a second of an alarm being raised by a sensor. In addition, when two or more sensors are positive, the**

**system calls the police with a location suspected break-in within 2 seconds of an alarm being raised by a sensor. If**

**alarmsare on, each door and window alarm should be polled twice per second.**

**• If the voltage of the alarm system drops between 10% and 20%, a battery backup must be switched on within a**

**deadline of 50ms, and the system runs a power supply test. If the voltage drops more than 20%, in addition to the**

**previous actions, the system must initiate an alarm and call the police. In case of a power supply failure or sensor**

**failure, the system calls a service technician. Wheneverthe console panic button is positive, the system must initiate**

**an alarm, turn on lights around the console and call the police.**

**• If a clear-alarms command is activated, all active alarms must be switched off,**

**and all lightsthat have been switched**

**on should be switched off.**

Functional Requirements:

Firstly: User requirements

1. Alarm system help the users to keep their places safe and handled by sensors and computer

2- alarm system is a system designed to detect intrusion – unauthorized entry – into a building or area.

System Requirements:

* 1. System must Call police if any abnormal patten take place
  2. System must Call police if power supply has problem
  3. System must technician if power supply has problem
  4. If a clear-alarms command is activated, all active alarms must be switched off
  5. If the voltage of the alarm system drops between 10% and 20%, a battery backup must be switched
  6. Sensors should detect any motion in the area
  7. Sensors should detect any try of enter the build
  8. Sensors should detect any change in status of doors and windows of area

Non Functional requirements :

(1Look-and-Feel Requirements :

-1The system shall use a large range of Alarm sounds

2-The system should use a lot of animation.

3- The system shall use a lot of Bright colors.

(2Usability & Humanity requirements :

1- The system shall be easy to use by member and should be organized in such away users error should be minimized .

2-the The system shall be easy to use by member and easy to learn it

3-Easy to managing whole the alarm alarm system it should take 30 minute after 2 hour training .

4-Users must change the initially assigned login password immediately after the first successful login. Moreover, the initial should never be reused.

3Performance requirements :

1- the system initiates alarm within half a second of an alarm being raised by a sensor.

2- turns on lights around the site of the positive sensor within half a second of an alarm being raised by a sensor.

3-- the system calls police within half a second of an alarm being raised by a sensor

4- the system call technician within half a second of a battery are being raised by a detector

5-the system turn off the alarm by the user within 20 seconed

6- Each movement detector should be polled twice per second.

7-Time to restart after failure for system should be 30 second.

8- The system should keep every id and password safe and hidden .

4 ( Maintainability:

* The system shall be able to handle more than 50 users at the same time within the first year after its publishing.
* The system should be able to support another language with its mother language within one year of publishing .
* The system may be require to be temporarily shut down and new modules as (ex: php classes ) to be added to the system or it may require a minor update in its graphical design (ex: html – css modules) .

5) Cultural requirements:

* The language that used in the interfaces of the system shall be polite and formal.
* The system shall not use any terms that may harm anyone on the planet.
* The system shall use English language in its interfaces.
* System shall not display any violence symbols or words in its interfaces.

6) Legal Requirements:

* The system shall follow restrictions and constraints of police and the safety in the country
* The system shall obey to the privacy policy and terms for all users.

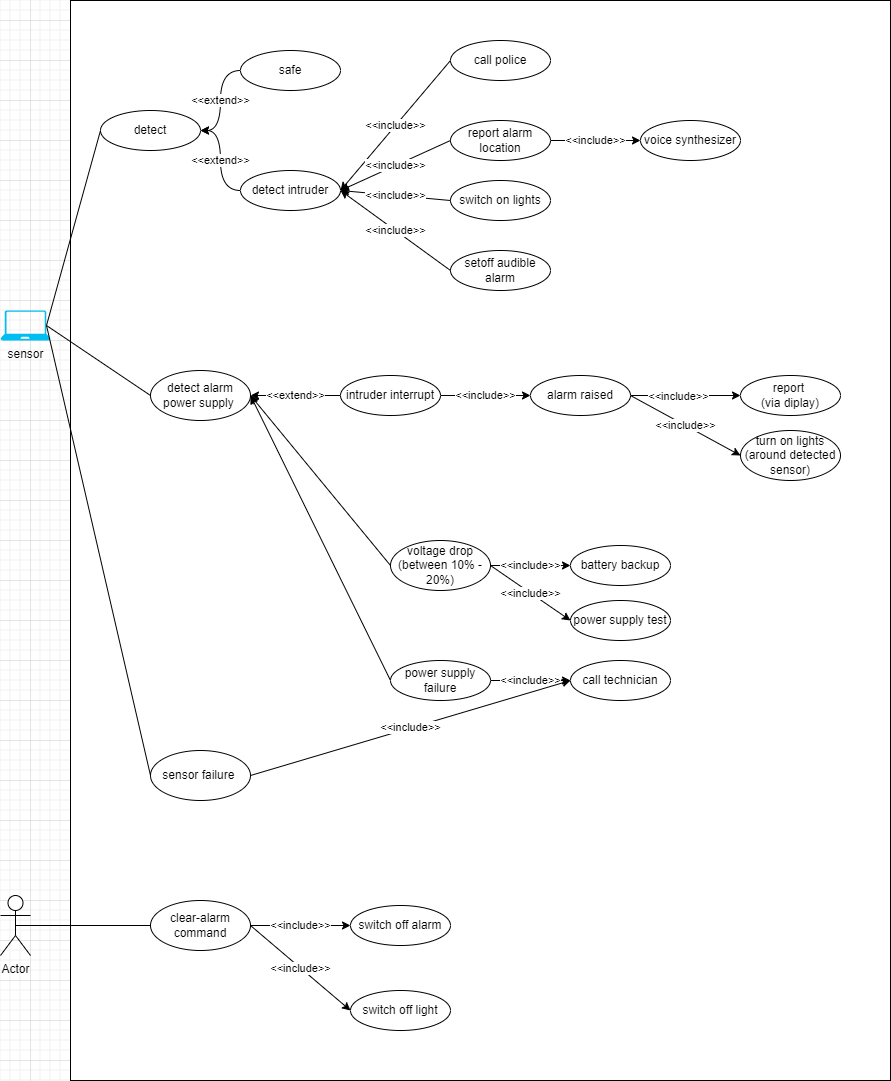
7) Security Requirements:

* The system should distinguish between authorized and non-authorized users.

Ex: user only have access to enter the system

* The system shall maintain information securely and ensure that it is only accessed by authorized users.
* The system shall provide a facility for users to request personal information and request changes to that information as change password.

**UML Diagrams**



**Burgular system:**

**Safe state:**

## sensors description

* + - Burglar system Use different type of sensor:
      * movement detection
      * door sensor
      * window sensor on ground floor

## sensor power supply:

* + - powered by main power
    - equipped with battery backup
    - power loss is detected:
      * using a separate power circuit monitor that monitors the main voltage. If a voltage drop is detected,

## Alarm state

1. **when sensor detect intruder the system:**
   * calls local police
   * report the location of alarm (using voice synthesizer)
   * switch on lights in the room around active sensor
   * setoff audible alarm

## sensor power supply:

* + if intruders have interrupted the power supply:
    - an alarm is raised
    - The alarm system includes a display, which reports these anomalies upon occurrence.
  + If the voltage of the alarm system drops between 10% and 20%,
    - a battery backup must be switched on within a deadline of 50ms,
    - and the system runs a power supply test.
* If the voltage drops more than 20%, in addition to the previous actions,

- the system must initiate an alarm and call the police.

* In case of a power supply failure or sensor failure, the system calls a service technician.

## Sensor:

* + Each movement detector should be polled twice per second.
  + if intruders have interrupted the power supply (which means that an intruder is detected):
    - the system initiates alarm within half a second of an alarm being raised by a sensor
    - and turns on lights around the site of the positive sensor within half a second of an alarm being raised by a sensor.
  + when two or more sensors are positive, (which means that an intruder is detected,):
    - the system calls the police with a location suspected break- in within 2 seconds of an alarm being raised by a sensor.
  + If alarms are on:
    - each door and window alarm should be polled twice per second.
  + Whenever the console panic button is positive
    - the system must initiate an alarm
    - turn on lights around the console
    - call the police.
  + If a clear-alarms command is activated
    - activated, all active alarms must be switched off,
    - and all lightsthat have been switched on should be switched off.

***Use case description***

|  |  |
| --- | --- |
| Use case name | Call police |
| Actors | Police, Alarm |
| Use case description | When the sensor detects ant abnormal thing it arise the alarm which call the police |
| precondition | The alarm is switched off and the police is passive |
| postcondition | Alarm system is raised and police is called |
| Normal flow | Sensor sends to alarm to wake it up then the alarm is call the police |
| Alternative flow | If the alarm failed to call police, it try to connect with the user |
| Nonfunctional | The alarm must report the police within 5 second |
| Extend points | Send the voice recorded |

|  |  |
| --- | --- |
| Use case name | Report alarm locatin |
| Actors | Location, police |
| Use case description | The alarm is sending the location to police |
| precondition | System saves his location to use in any time |
| postcondition | System send location to police |
| Normal flow | When the alarm is a rise the alarm system calls police then send location to |
| Alternative flow | The system fail to send location it try to connect to user |
| Extend points | Voice synthesis |

|  |  |
| --- | --- |
| Use case name | Switch off lights |
| Actors | Alarm |
| Use case description | When alarm system is a raised the light should switch on |
| precondition | Light should be turned on |
| postcondition | Light should be turned off |
| Normal flow | The alarm should turn off the lights |
| Alternative flow | System never can turn off the lights |
| Extend points |  |

|  |  |
| --- | --- |
| Use case name | Set off audible alarm |
| Actors | Alarm, sound system |
| Use case description | The Alarm turn on the voice to make the niobous detect that there is thief |
| precondition | Sound is stored to be ready for ant event |
| postcondition | Alarm turns the sound on |
| Normal flow | If alarm system is a raised the alarm should turn on the sound |
| Alternative flow | If alarm system can’t a raised the alarm should  Connect to the |
| Extend points |  |

|  |  |
| --- | --- |
| Use case name | Detect |
| Actors | sensor |
| Use case description | The sensor detects any abnormal |
| precondition | Sensor is running and reading the reads |
| postcondition | Sensor raises the alarm system |
| Normal flow | Sensor detects an interrupt then it arise the alarm system |
| Alternative flow | Sensor breaks down and can’t detect any abnormalities |
| Extend points | Safe, detects intruder |

|  |  |
| --- | --- |
| Use case name | Detect intruder |
| Actors | Sensors, intruders |
| Use case description | The sensor detects the intruders then a raise the system |
| precondition | The sensor is ready to detect any abnormal |
| postcondition | Sensor send status to alarm system to arise it |
| Normal flow | The sensor detects the intruders then a raise the system |
| Alternative flow | Sensor is broken down and can’t detect any intruders |
| Extend points | Call police, turn on lights, switch in the audible |

|  |  |
| --- | --- |
| Use case name | Detect alarm power supply |
| Actors | Power supply, sensors |
| Use case description | The sensor detect defect in power supply them backs up the battery and start test |
| precondition | Sensor is testing the power supply permeant |
| postcondition | If the power supply drop down between 10to20% then start backup the battery  If the power supply drops down more than 20% then call police and call tech |
| Normal flow | If the power supply drops down between 10to20% then start backup the battery  If the power supply drops down more than 20% then call police and call tech |
| Alternative flow | The system drop down immediately and the sensor can’t detect the fast change |
| Extend points | Power supply failure , intruder interrupt |

|  |  |
| --- | --- |
| Use case name | Alarm araise |
| Actors | Sensor, Alarm |
| Use case description | When the sensor detect change in the power supply level |
| precondition | Sensor doesn’t a raise the alarm |
| postcondition | Sensor a raises the alarm |
| Normal flow | Sensor detects change in the power supply then it wakes up the alarm |
| Alternative flow | Alarm dose’s response to the senor |
| Extend points | Reports, turn on lights |

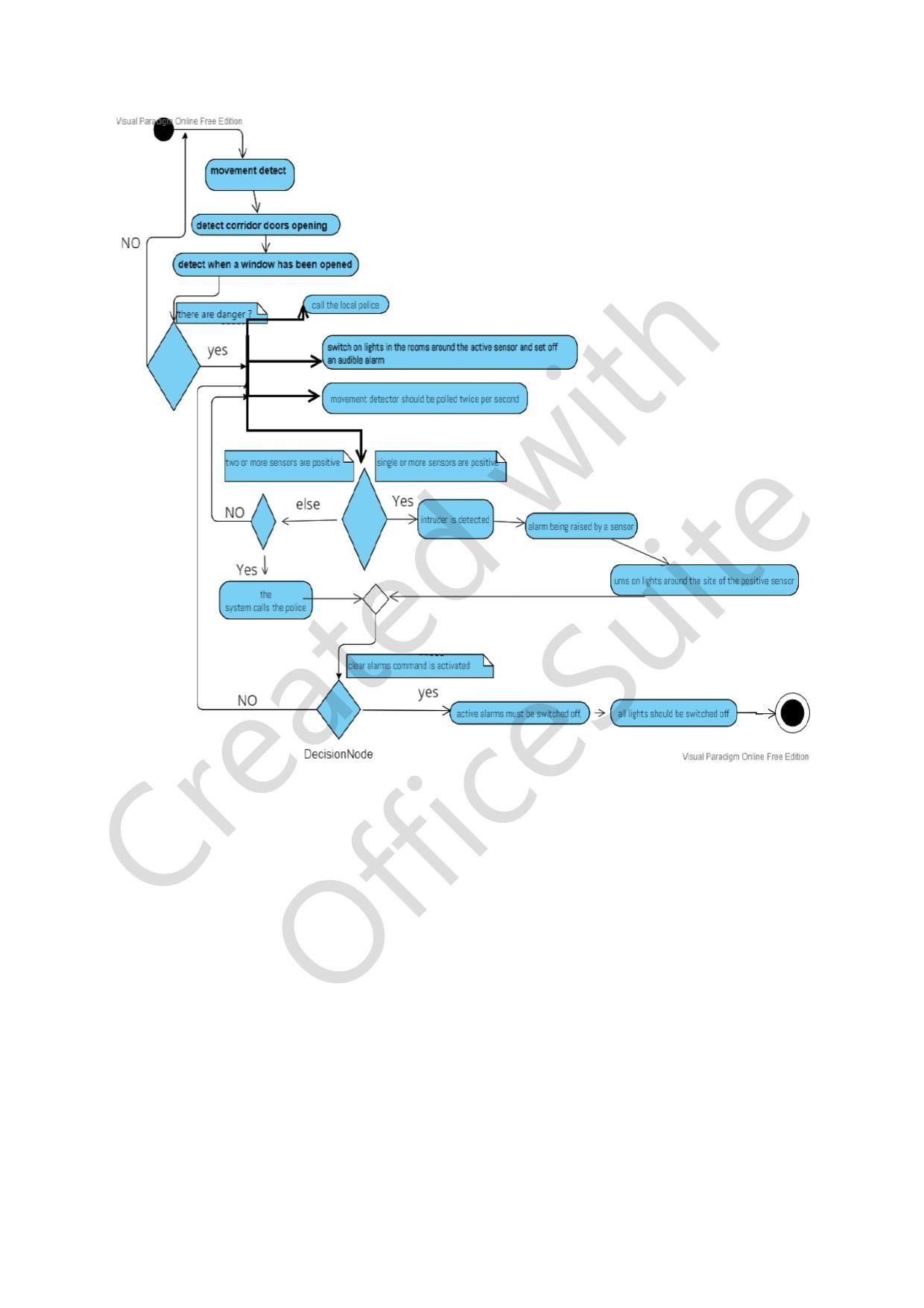
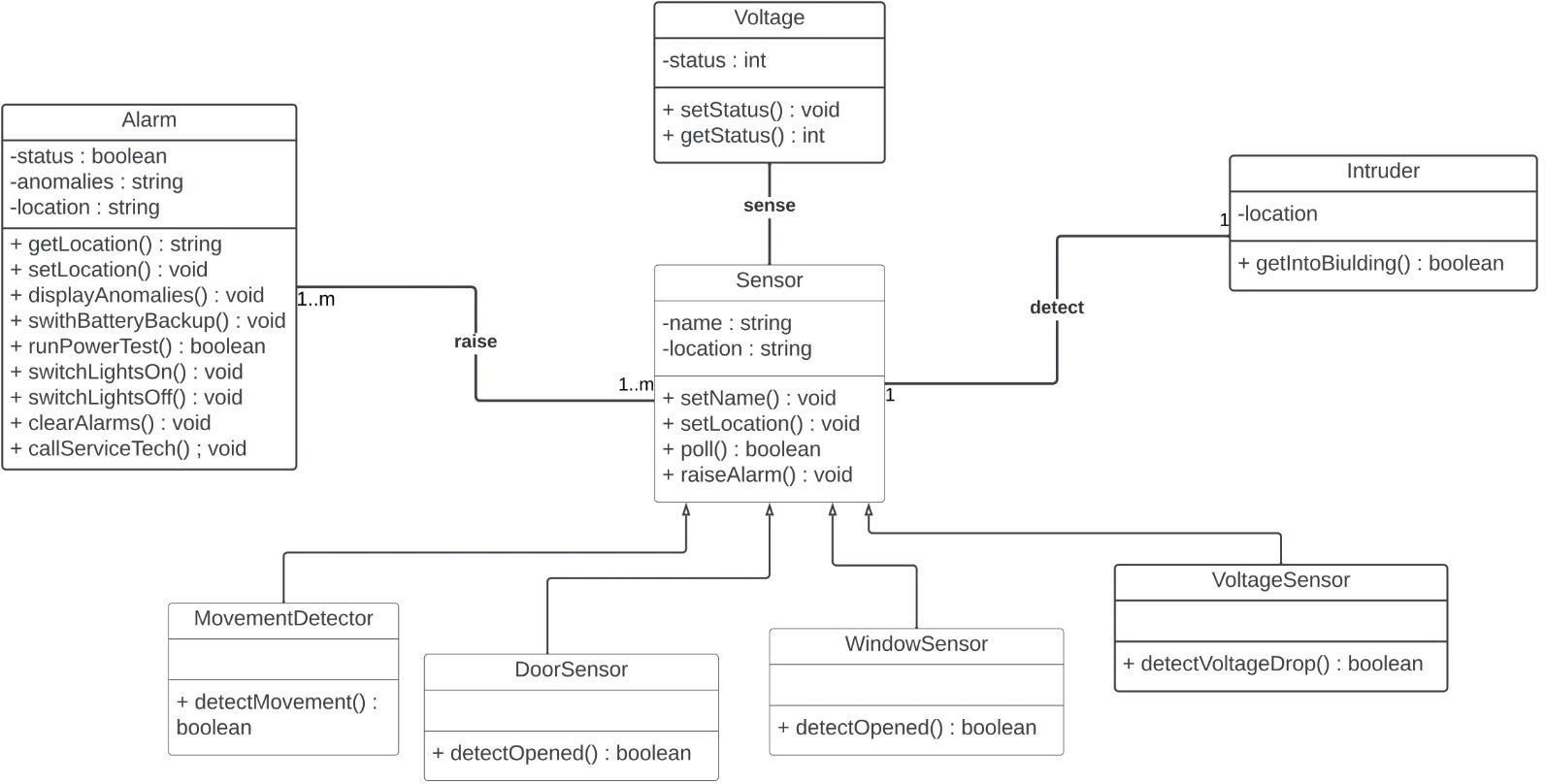
|  |  |
| --- | --- |
| Use case name | Power supply failure |
| Actors | Battery, voltage sensor |
| Use case description | Power supply is drop more than normal and can’t wake up the system |
| precondition | Voltage detector should continuously manage the power supply level |
| postcondition | Power supply failure then the alarm is wake up and call the tech |
| Normal flow | When power supply failure the alarm system is a raised and arise the battery then call the technical |
| Alternative flow | Power supply failure and alarm can’t wake up the alternative battery |
| Extend points | Call technical |

|  |  |
| --- | --- |
| Use case name | Clear\_alarm\_ command |
| Actors | User, alarm |
| Use case description | Users want to stop the activities of the system |
| precondition | System run normally, alarm is switched on, lights is switched on |
| postcondition | alarm is switched off; lights is switched off |
| Normal flow | Users enter the command the order to close the system |
| Alternative flow | System doesn’t response to the command of the user |
| Extend points | Switch off lights, switch off alarm |

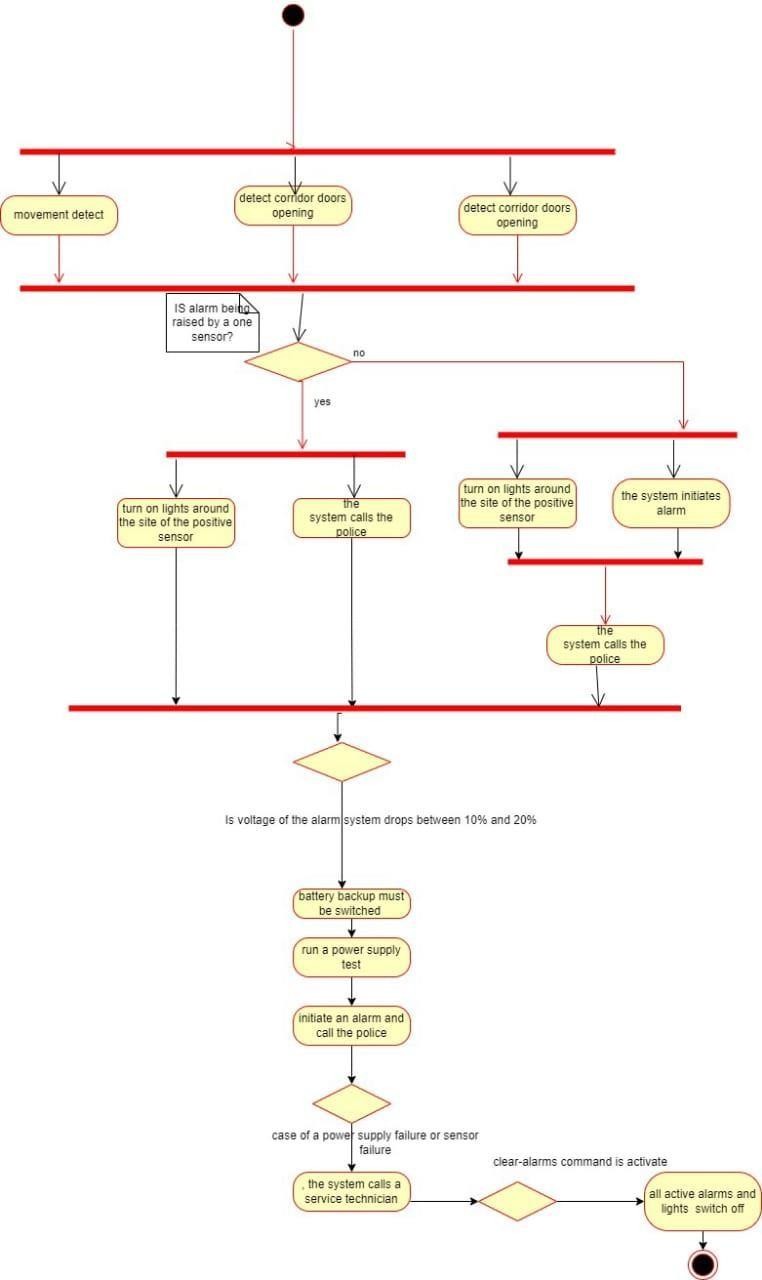
|  |  |
| --- | --- |
| Use case name | safe |
| Actors | Sensor |
| Use case description | The sensors don’t detect any abnormalities in the system |
| precondition | Sensors continuously measure their tasks |
| postcondition | Sensors continuously measure their tasks |
| Normal flow | Sensors continuously measure their tasks |
| Alternative flow | Sensor failure to read and always repot normal status |
| Extend points |  |

|  |  |
| --- | --- |
| Use case name | Battery back up |
| Actors | battery |
| Use case description | If power supply failure the battery is backed up |
| precondition | Battery is charged and ready to be backed up |
| postcondition | Battey is backed up and run the system instead of the power supply |
| Normal flow | Power supply failure arise the alarm then alarm makes battery to be used |
| Alternative flow | Battery is damaged and can’t run the system |
| Extend points |  |
| Nonfunctional | The battery must be waked up within 30 second by alarm |

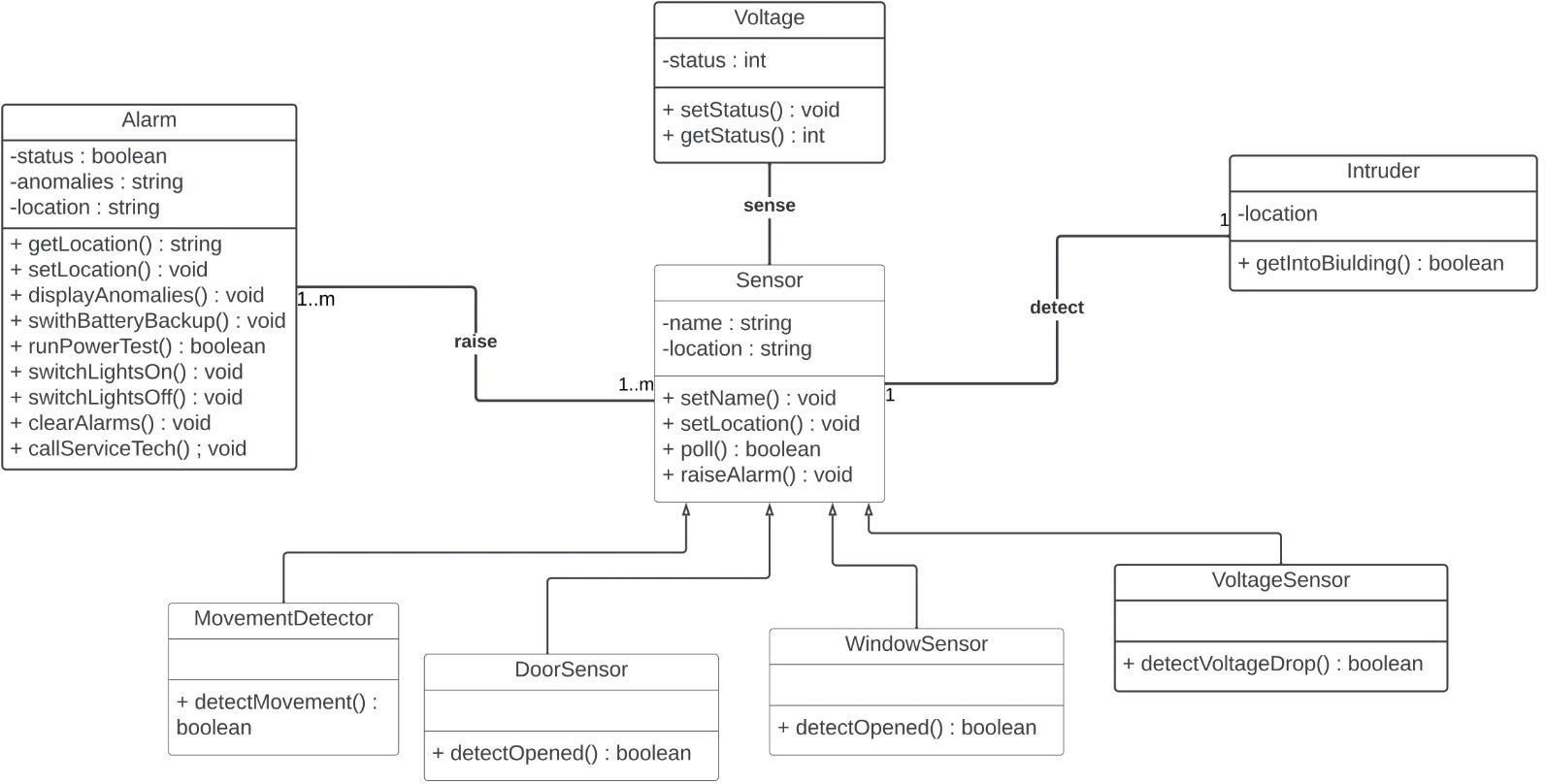
**1st versio**

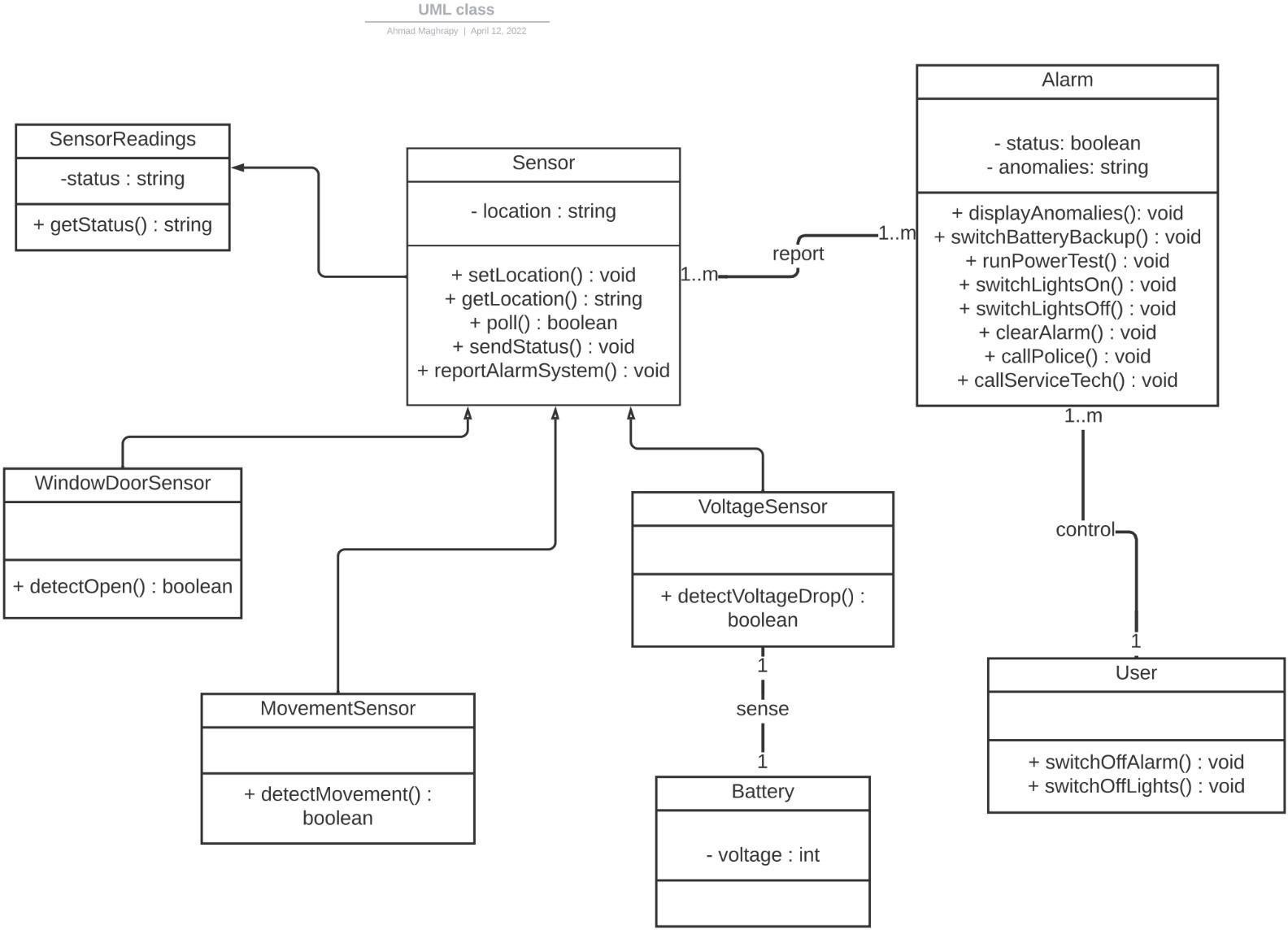


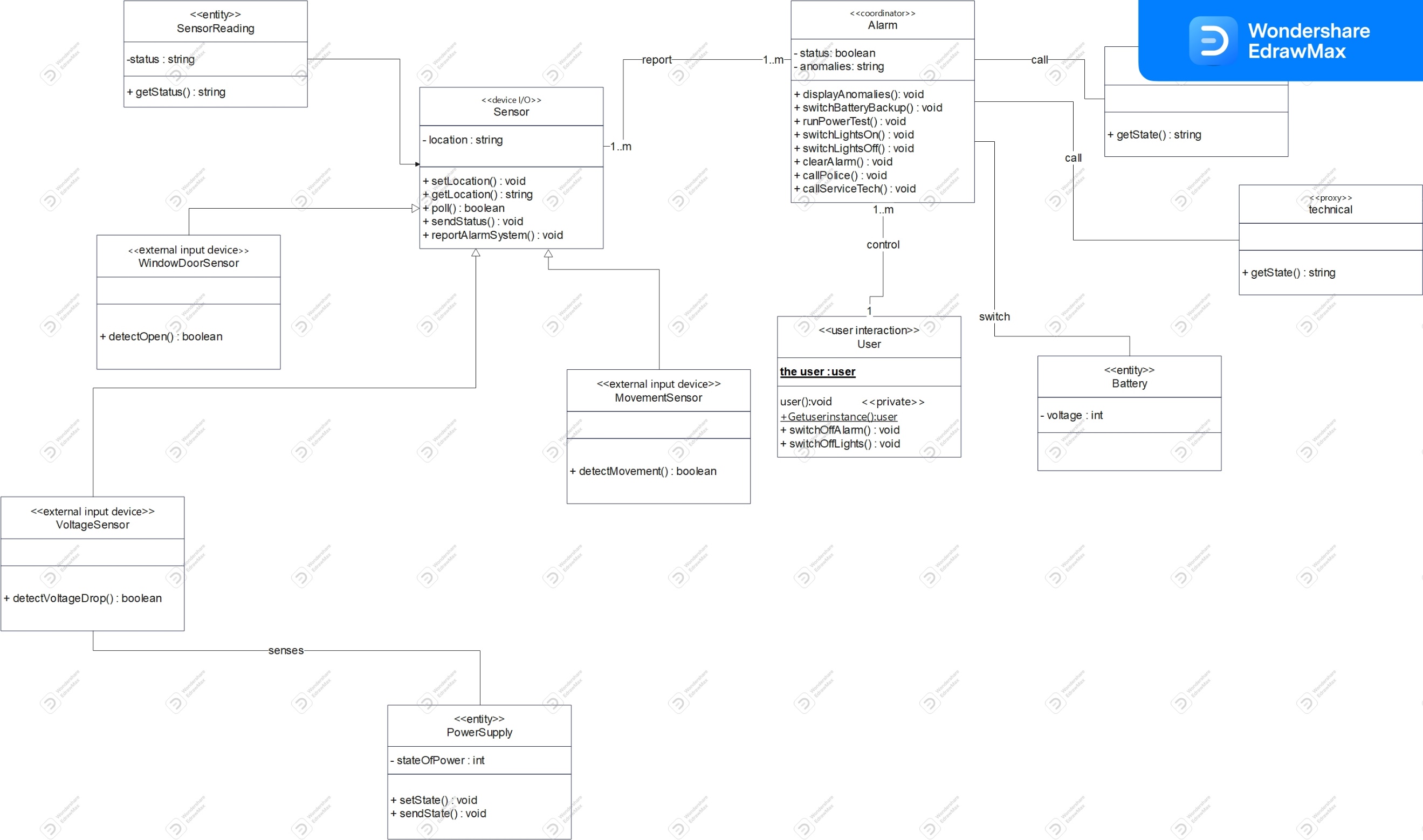
**2nd version**

Page **9** of **2**

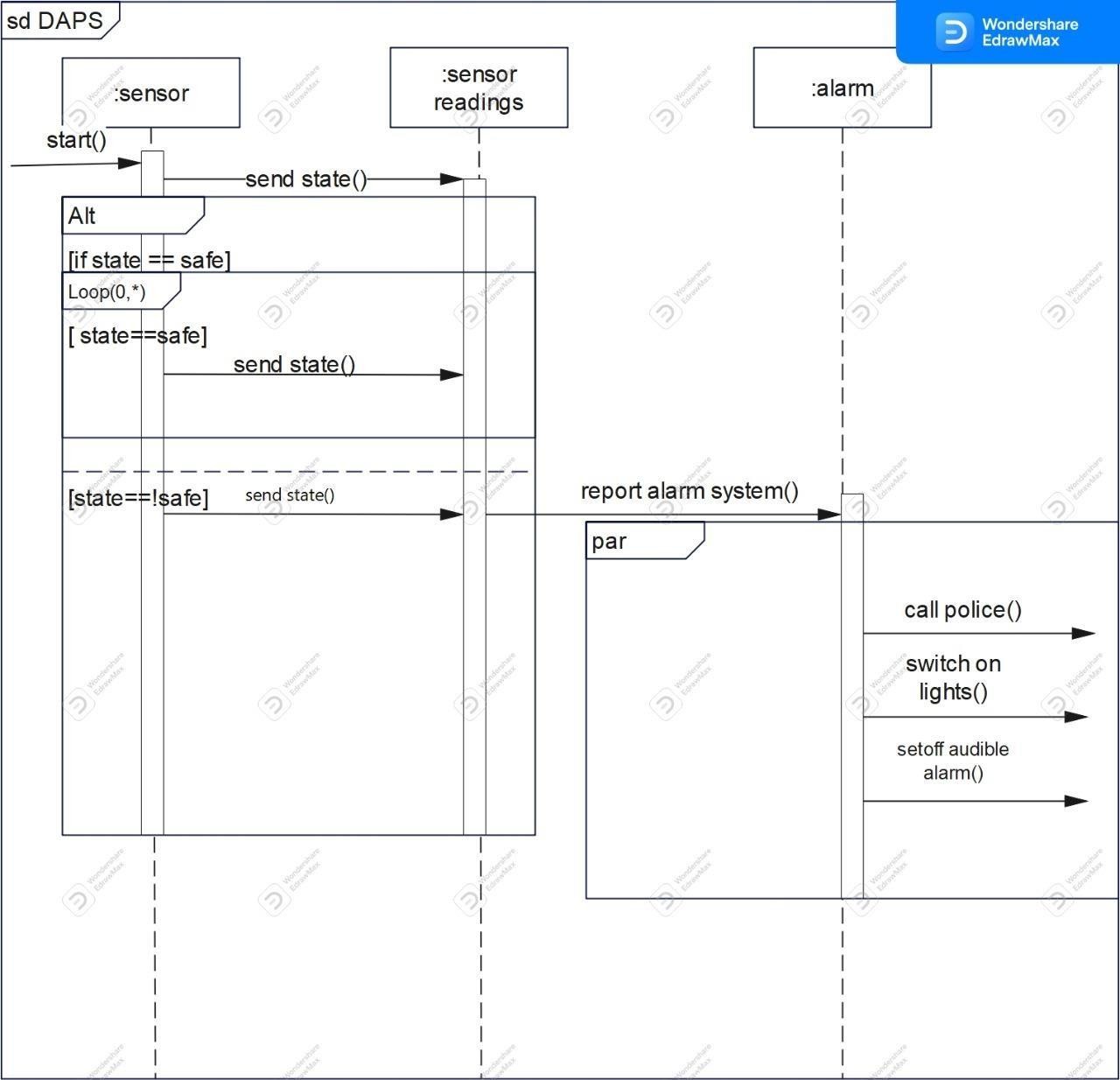
***Class digrams***

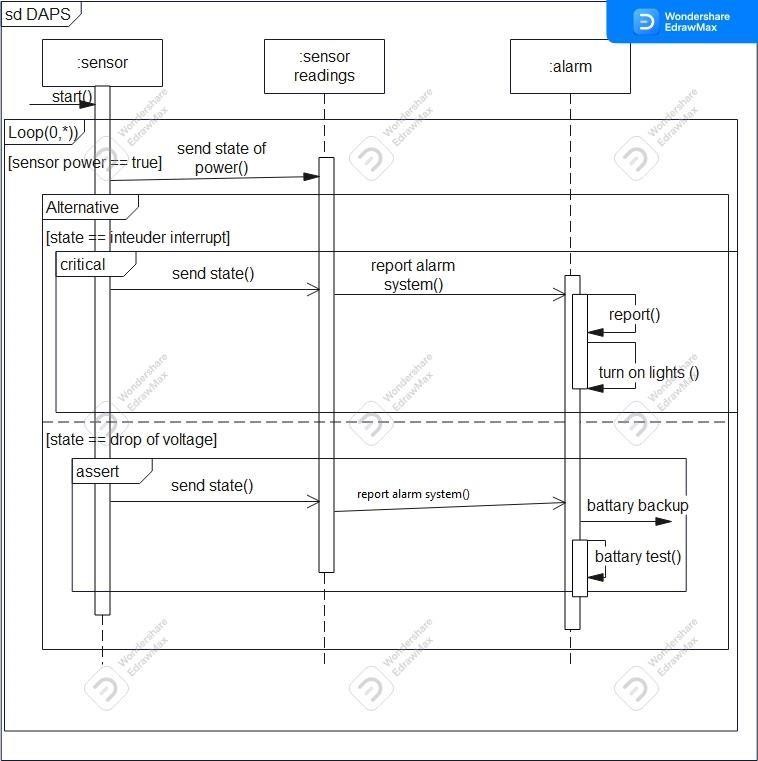
**1st version**

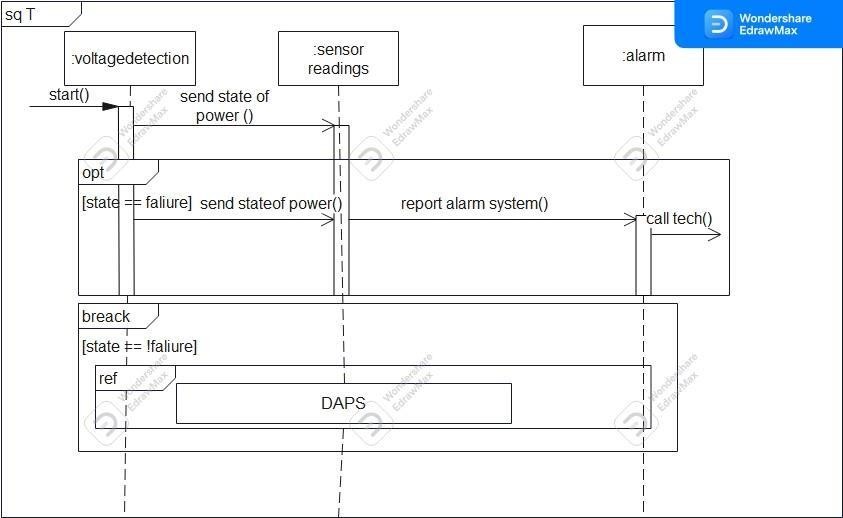
**2nd version**

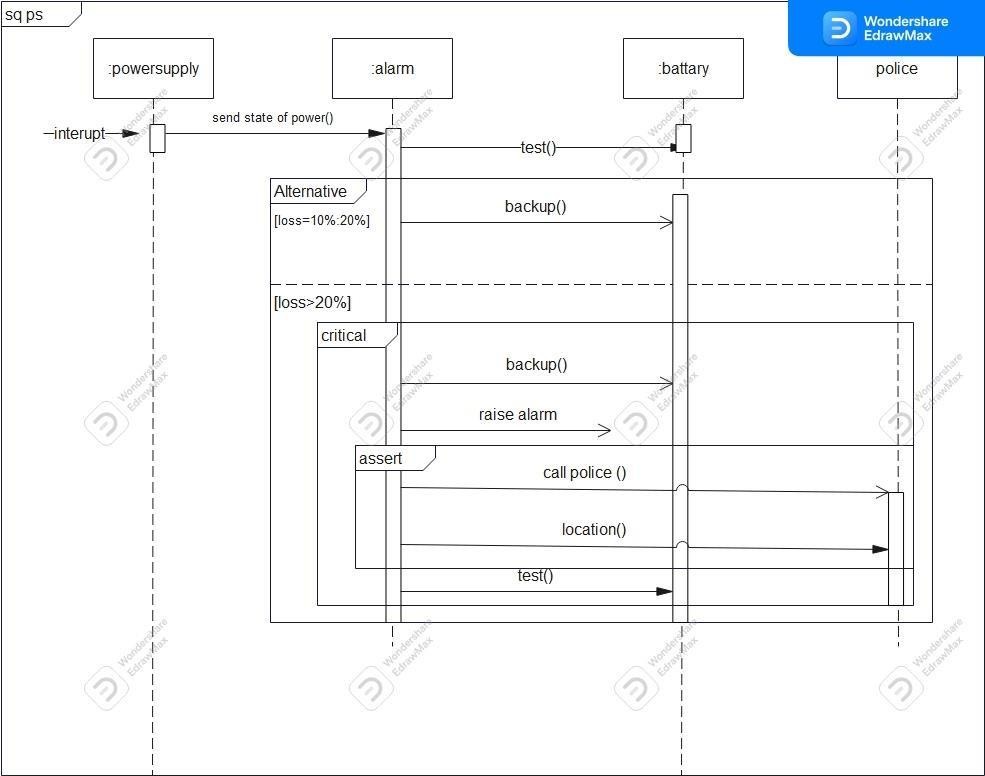
1. **version**

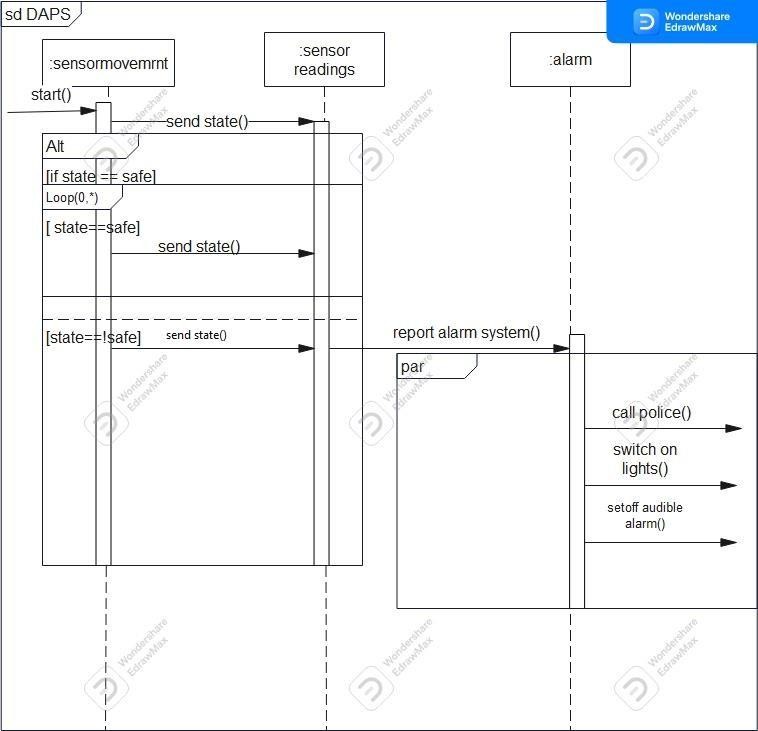
**Sequence diagrams**

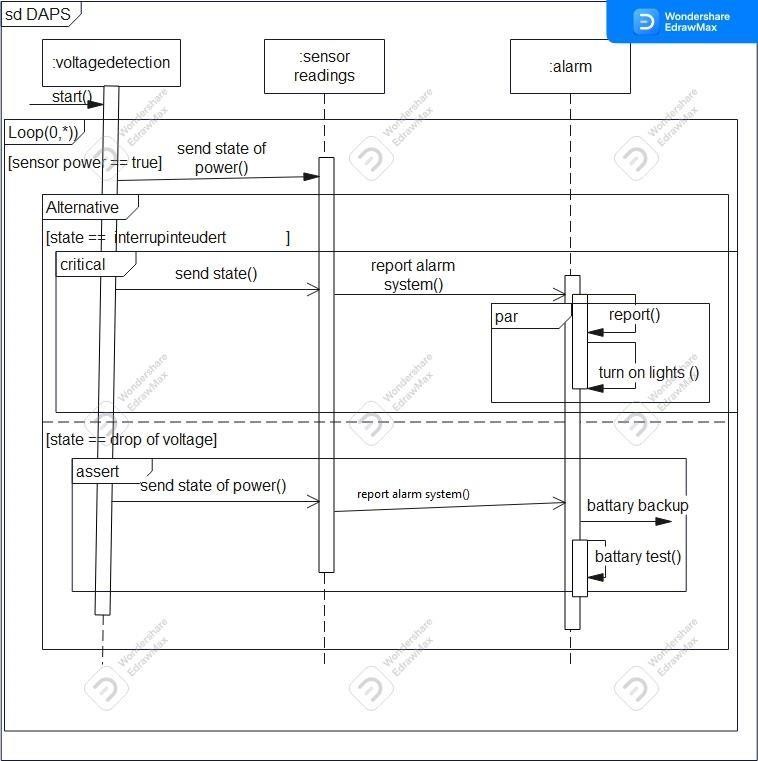


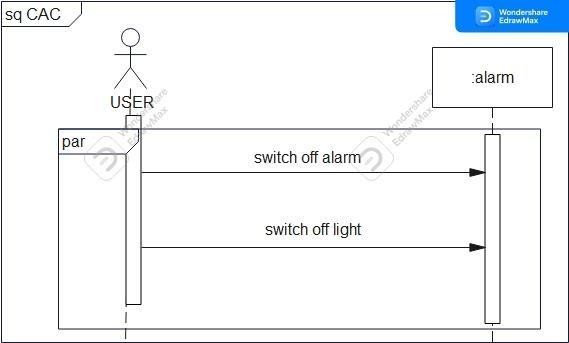




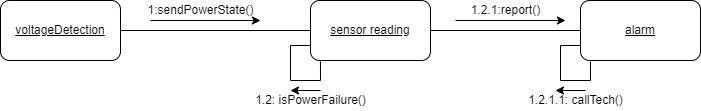


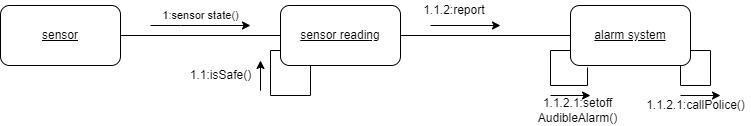


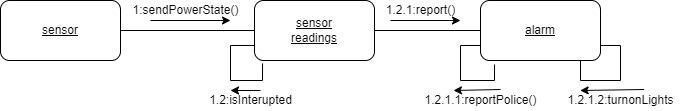


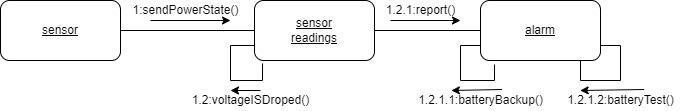


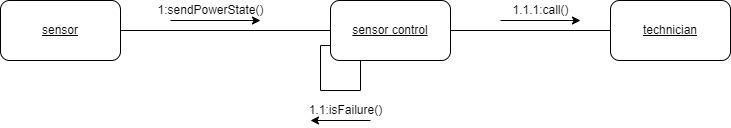
**Calibration diagrams**

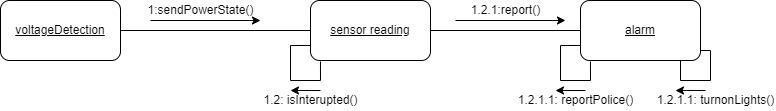


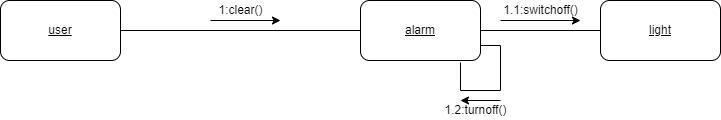


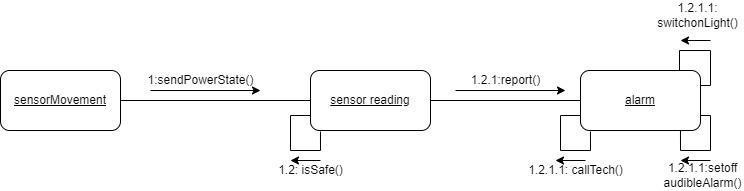






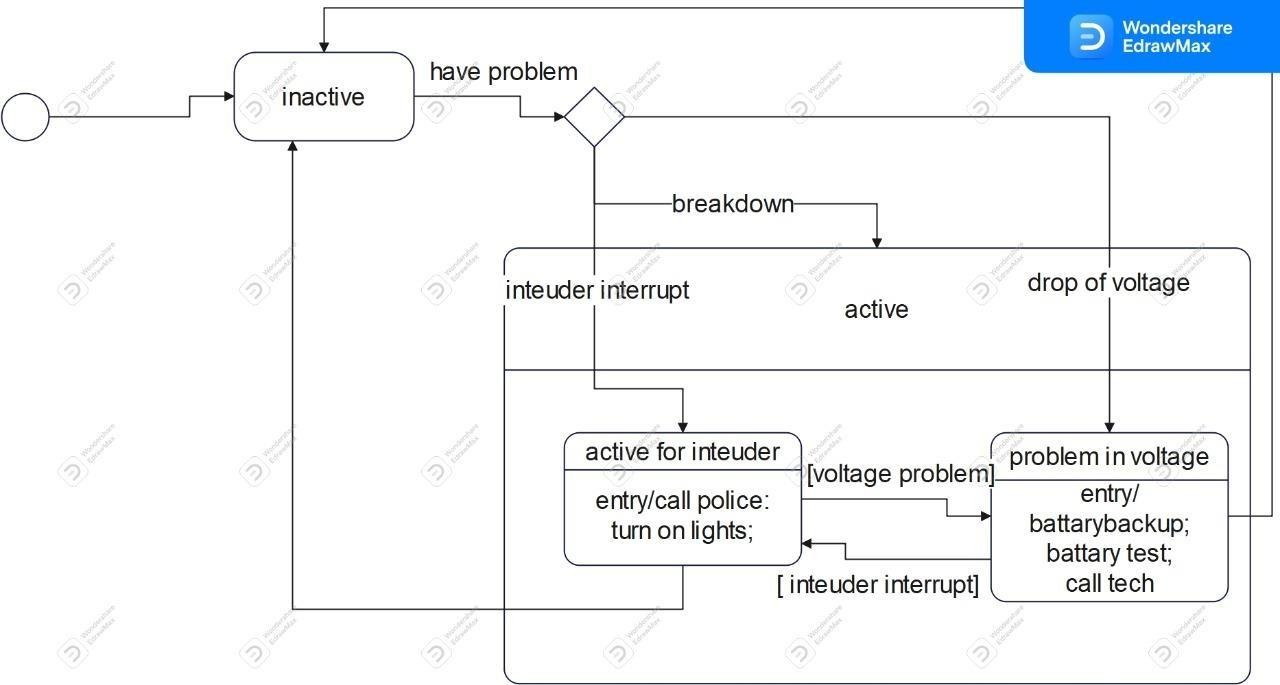




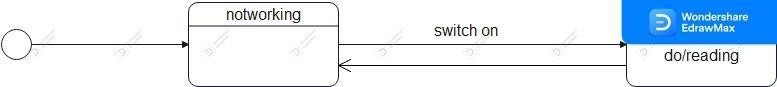


State machine digrams

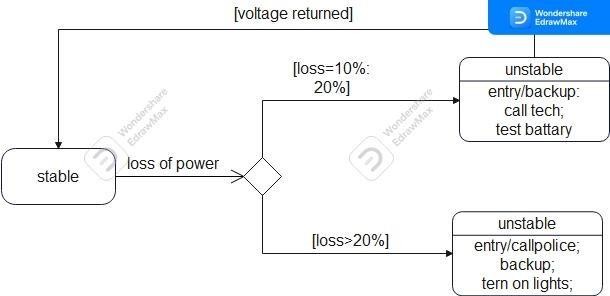
# State machine for alarm



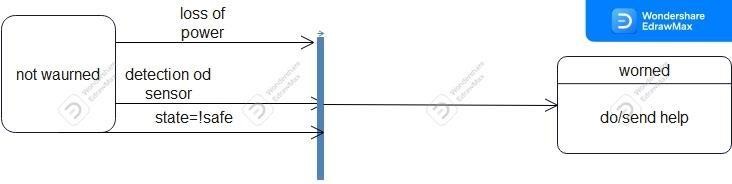
***State machine for sensor***



***State machine for power supply***

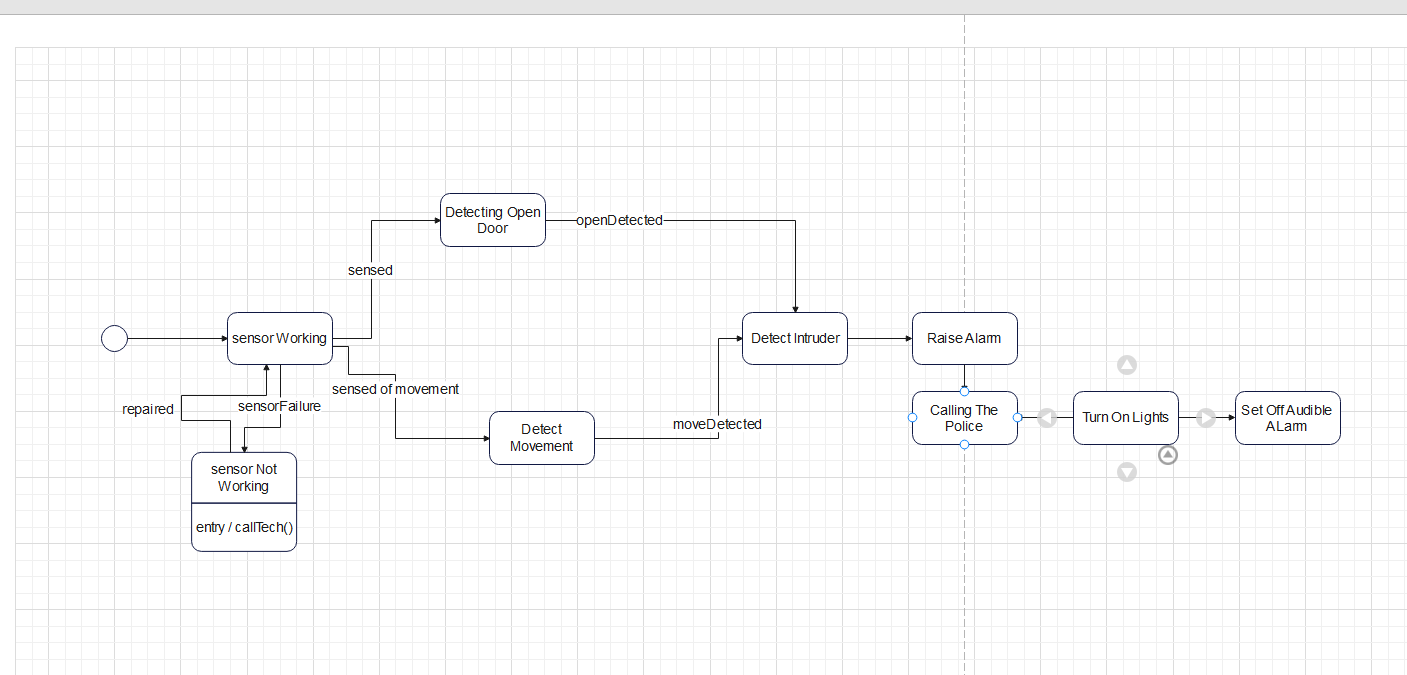


***State machine for police***



***Update state machine diagrams***

**Use case: detect**

****

**Diagram

Description automatically generated**

|  |  |
| --- | --- |
| S1 | sensor Working |
| S2 | sensor Not Working |
| S3 | Detecting Open Door |
| S4 | Detect Movement |
| S5 | Detect Intruder |
| e1 | sensed of movement |
| e2 | sensed |
| e3 | open Detected |
| e4 | move Detected |
| e5 | sensor Failure |
| e6 | repair |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **State/event** | E1 | E2 | E3 | E4 | E5 | E6 |
| S1 | S4 | S3 | ------ | ------ | S2 |  |
| S2 | ------- | -------- | ------- | -------- | -------- | S1 |
| S3 | -------- | -------- | S5 | -------- | -------- | ------- |
| S4 |  |  |  | S5 |  |  |
|  |  |  |  |  |  |  |

**Diagram

Description automatically generated**

|  |  |
| --- | --- |
| S1 | Working |
| S2 | Not Working |
| S3 | Detecting Voltage Drop |
| S4 | Switching Battery Backup |
| S5 | Actions |
| e1 | sensor failure |
| e2 | poll |
| e3 | voltage drop > 20 |
| e4 | voltage drop > 10 |
| e5 | power supply failure |
| E6 | repair |
| E7 | user command |

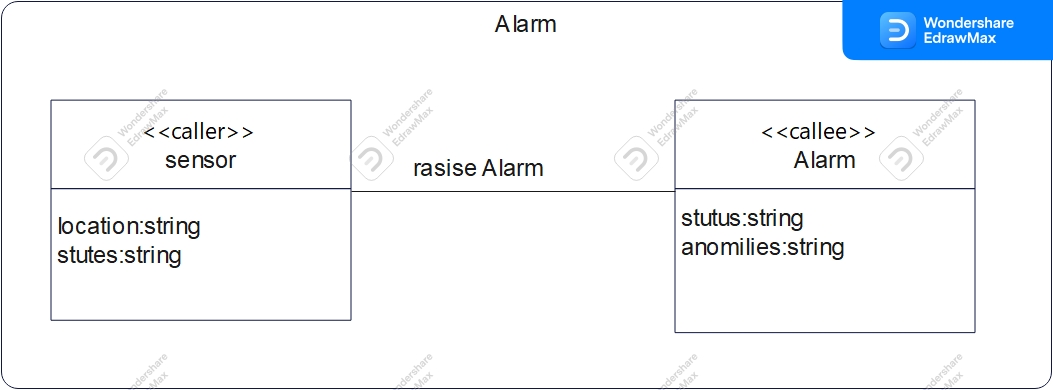
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **State/f state** | S1 | s2 | s3 | s4 | s5 | s6 |
| S1 | -------- | E1 | E2 | ------ | -------- | ------- |
| S2 | e6 | -------- | ------- | -------- | -------- | ------- |
| S3 | -------- | -------- | -------- | E4 | E5, e3 | ------- |
| S4 | -------- | -------- | -------- | -------- | -------- | ------- |
| S5 | E5 | -------- | -------- | -------- | -------- | ------- |

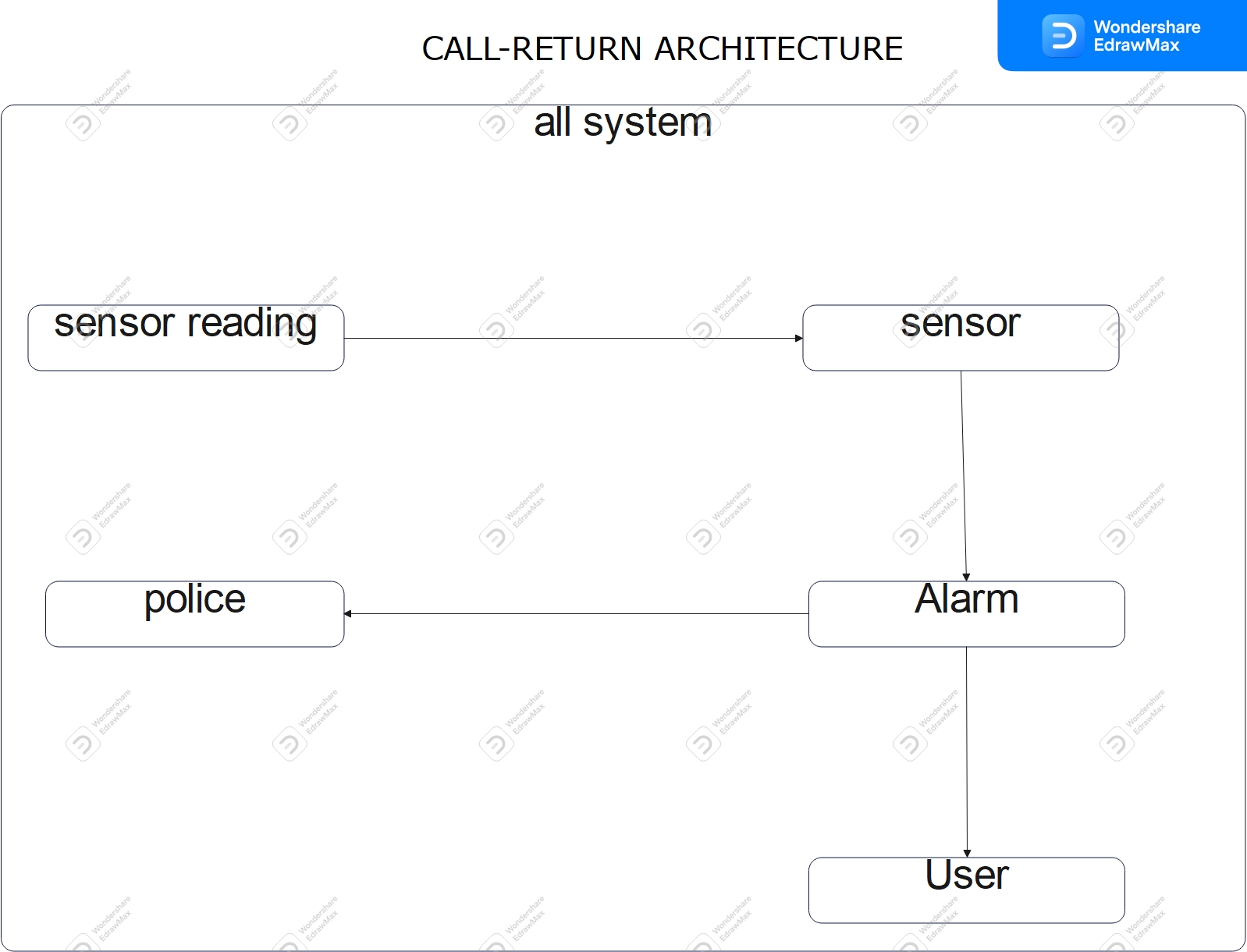
**Diagram

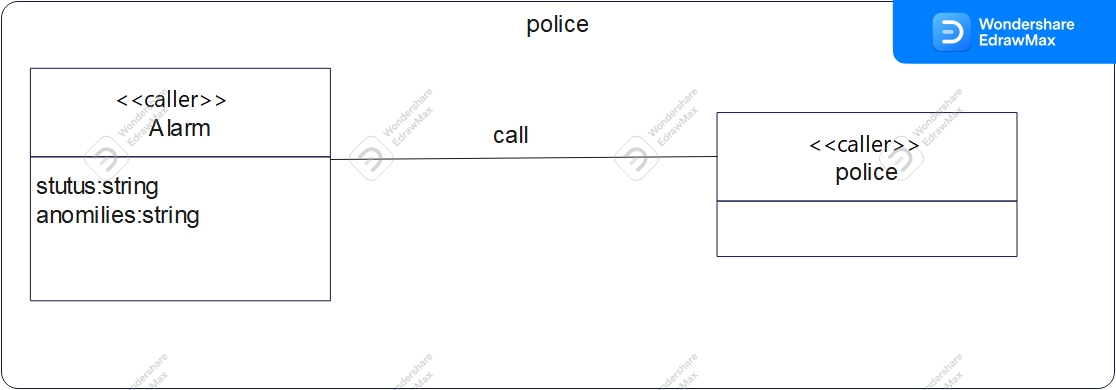
Description automatically generated**

**architecture patterns**

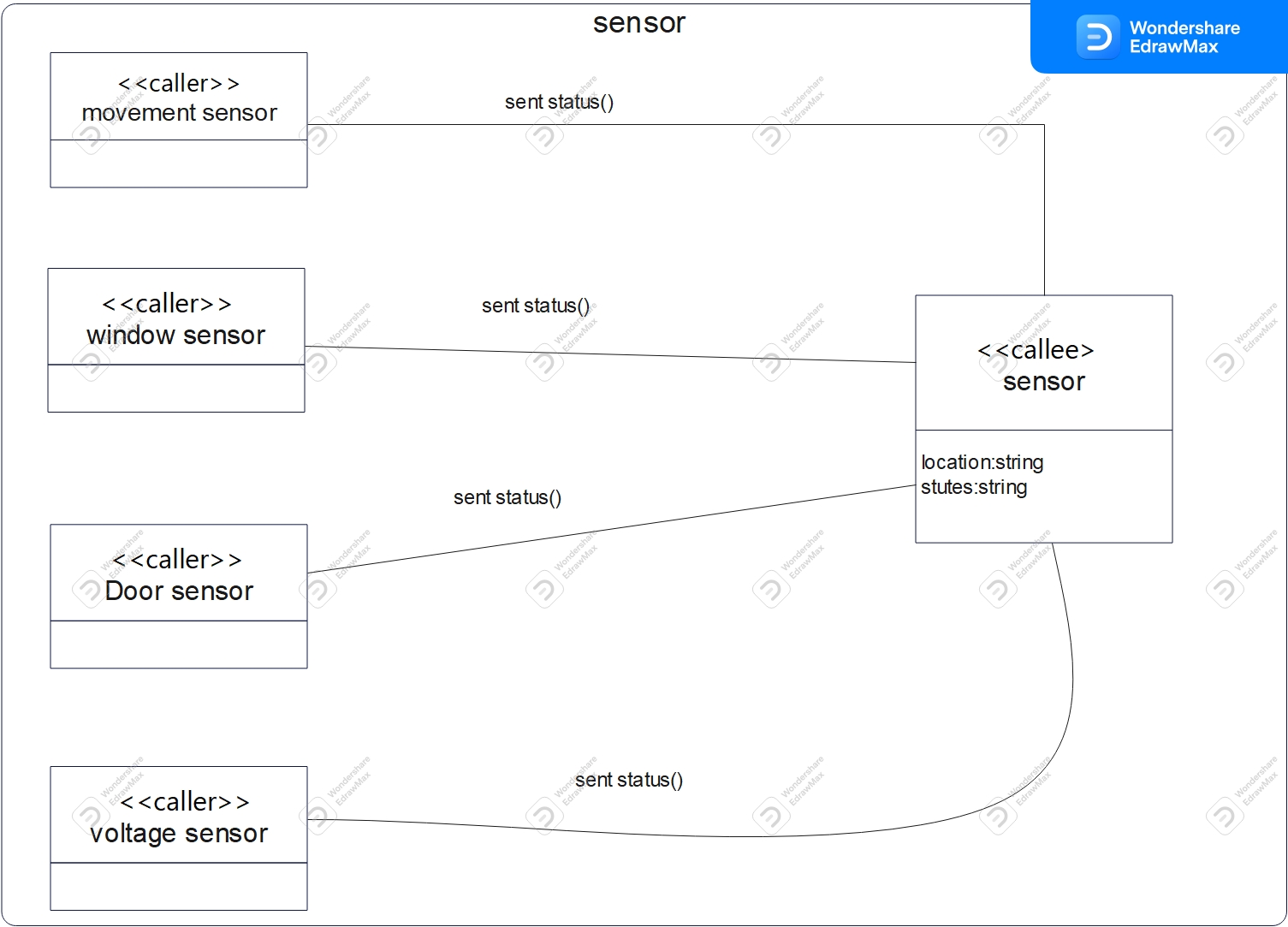
**Call-Return**







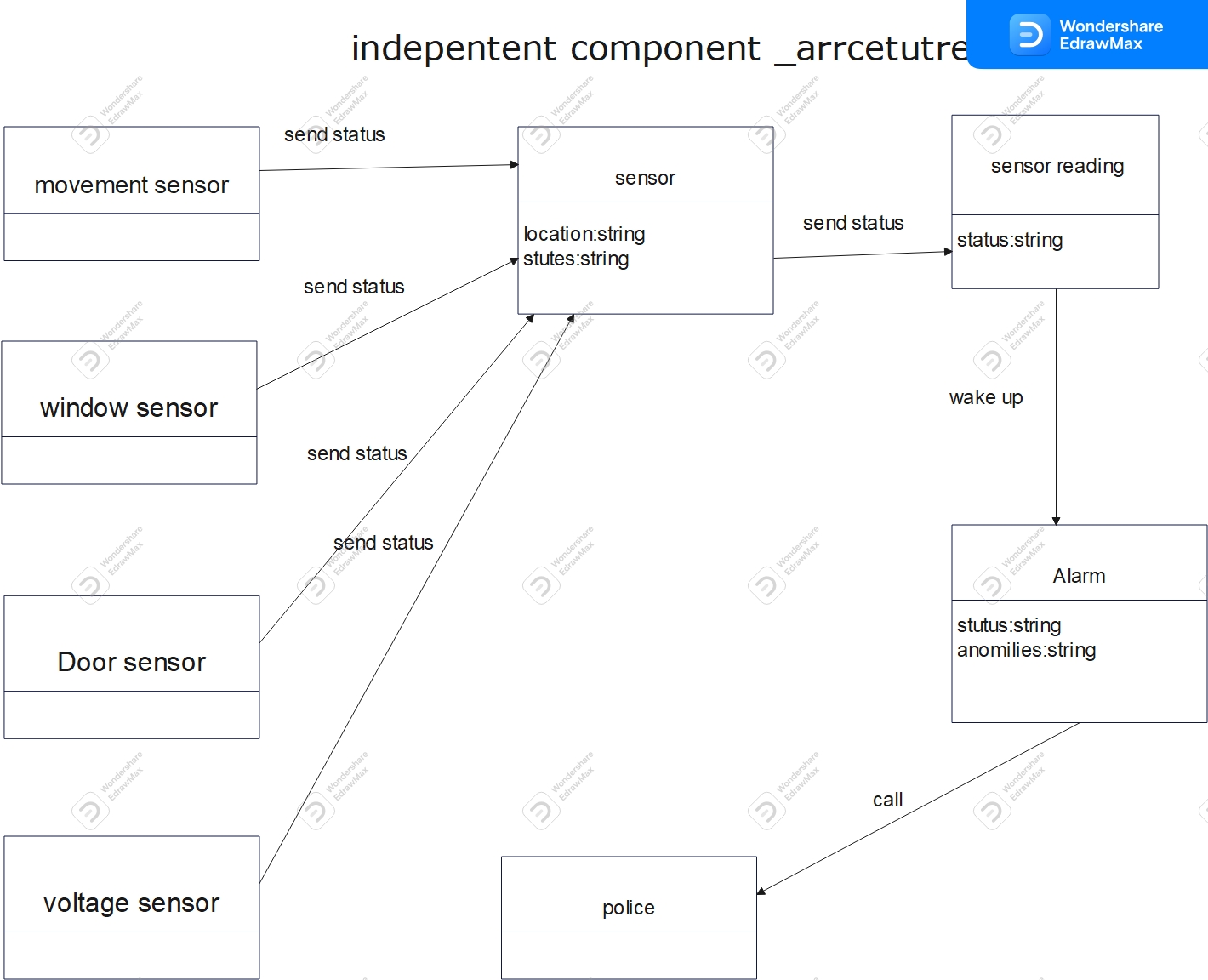




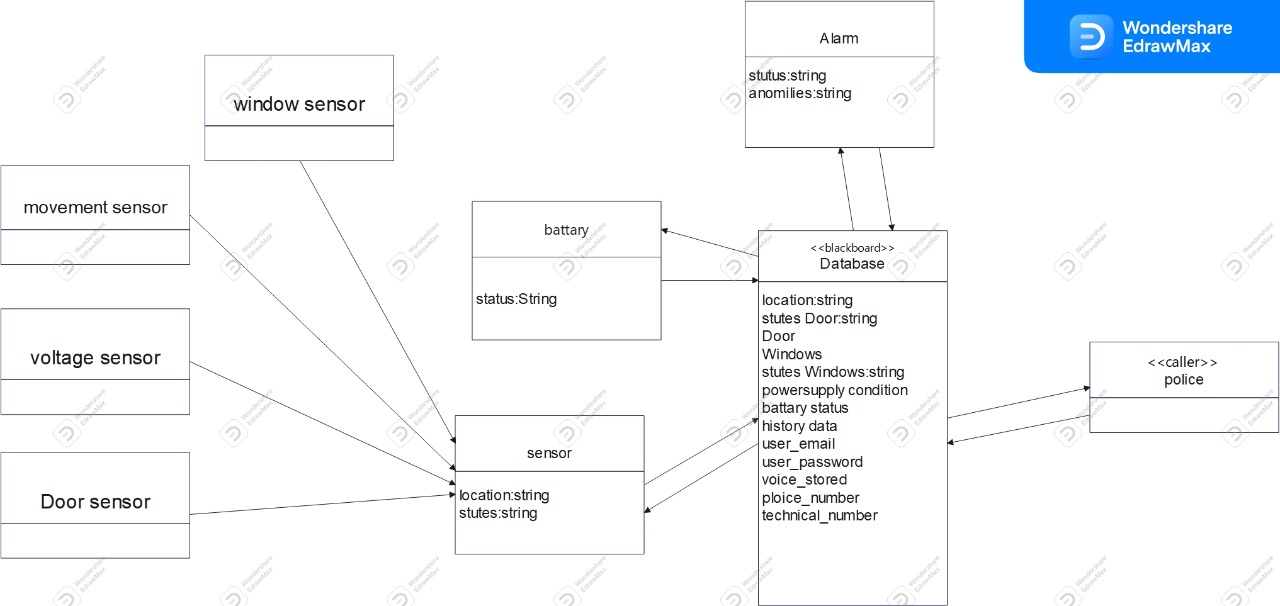
**Diagram

Description automatically generated**

***Independent Components***



***Data-flow***



**Design patterns**

Design patterns description

1) Singleton pattern (creational pattern):

Context: It is very common to find classes for which only one instance should exist (singleton).

Problem: How do you ensure that it is never possible to create more than one instance of a user class. And provide a global point of access to it.

Forces: The use of a public constructor cannot guarantee that no more than one instance will be created. user instance must also be accessible to all classes that require it; therefore it must often be public.

Solution: Have the constructor private to ensure that no other class will be able to create an instance of the class user. Define a public static method called “Get user access ()”, The first time this method is called , it creates the single instance of the class “user” and stores a reference to that object in a static private variable called “the user

\

Example : Applying on user Class

If (the User==null)

the User=new User;

If (the User If (the User==null)

the User=new User;

return the User;

null)

the User=new User;

return the User;

return the User;

|  |
| --- |
| User |
| theUesr:user |
| -user ():void <<private>>  +GetUseInstance():Amin |

2) Delegation pattern (structrual): “applying on many classes in the project”

Context: You are designing a method in a class. You realize that another class has a method which provides the required service. Inheritance is not appropriate (e.g., because the is-a rule does not apply).

Problem: How can you most effectively make use of a method that already exists in the other class?

Forces: You want to minimize development cost by reusing methods.

Solution: The delegating method in the delegator class in our example (sebsor,sensor reading , power supply classes ) calls a method in the delegate class in our example (status class)to perform the required task. An association must exist between the delegator and delegate classes

Example:

Delegate class Delegator classes

|  |
| --- |
| Status |
|  |
| setstatus (string):boolean |

|  |
| --- |
| Sensor reading |
|  |
| sendstatus (string):boolean |

|  |
| --- |
| Power supply |
|  |
| sendstatus(string):boolean |

3) OBSERVER

Context: ▪When partitioning a system into individual classes you want the coupling between then to be loose so you have the flexibility to vary them independently.

Problem: ▪A mechanism is needed to ensure that when the state of an object changes related objects are updated to keep them in step.

Forces: ▪The different parts of a system must kept in step with one another without being too tightly coupled

Solution: ▪ One object has the role of the subject/publisher and one or more other objects the role of observers/subscribers.

The observers register themselves with the subject, & if the state of the subject changes the observers are notified & can the update themselves.

▪ The are two variants: ▪the Push Model where the subject send the observers detailed information about the change that has occurred, and

▪the Pull Model where the subject simple notifies the observers that there have been changes, and it's the responsibility of the observers to find out the details they need to update themselves

Example:

Observable observer

|  |
| --- |
| sensor |
|  |
| setstatus (string):boolean |

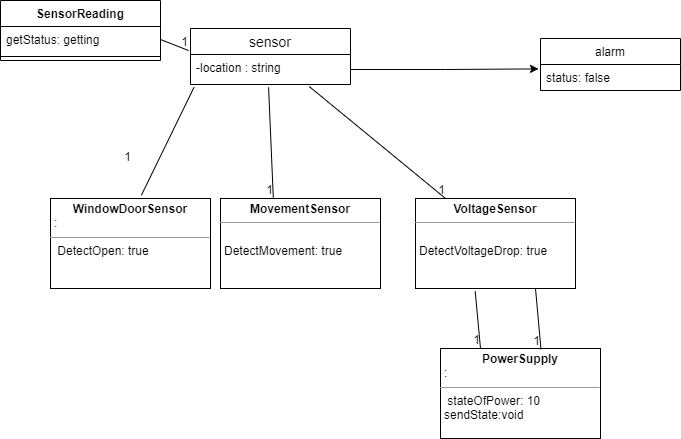
|  |
| --- |
| Alarm |
|  |
|  |

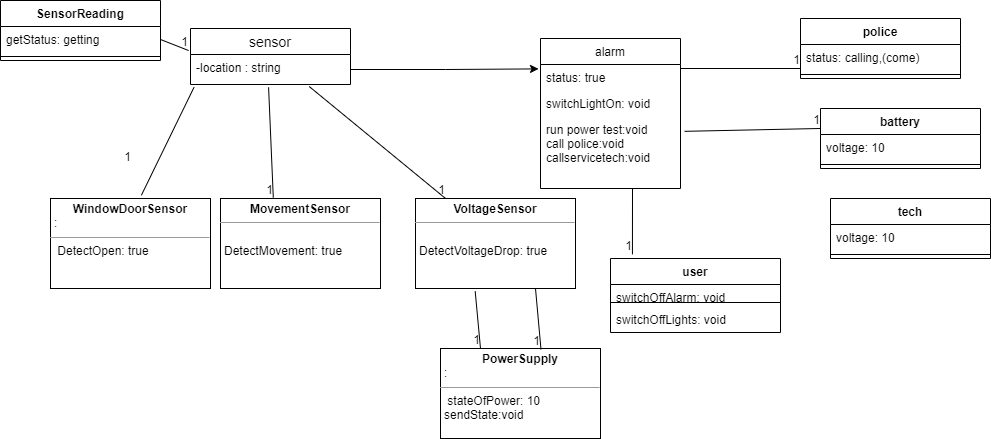
|  |
| --- |
| Movement sensor |
|  |
|  |

|  |
| --- |
| voltage sensor |
|  |
|  |

|  |
| --- |
| Door sensor |
|  |
|  |

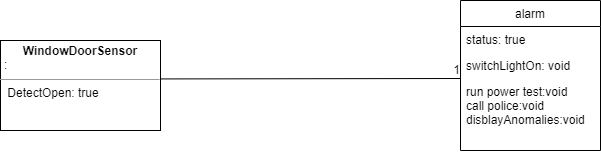
Object digrams





Shape

Description automatically generated with medium confidence



Shape

Description automatically generated with medium confidence

A picture containing shape

Description automatically generated

A picture containing shape

Description automatically generated