

Automatic management of INSA's rooms



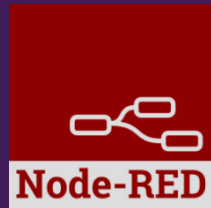
Florian LEON & Teo GENEAU & Walid KHALED

Encadré par :
Mme GUERMOUCH & Mme GHADA

Salle autonome

Table des matières

| | |
|--|---|
| Méthode agile | 2 |
| SCRUM Process | 2 |
| Our user stories | 3 |
| Overview | 4 |
| Scenario | 5 |
| Microservices created | 5 |
| Global Architecture..... | 6 |
| Presence Sensor | 7 |
| Light Sensor | 7 |
| CO2-Humidity Sensor | 7 |
| Temperature Sensor | 7 |
| Controller..... | 7 |
| Tests..... | 8 |
| Jenkins..... | 9 |
| Project build with Jenkins Git | 9 |
| We notice that all the builds are successful. | 9 |
| War file creation | 9 |



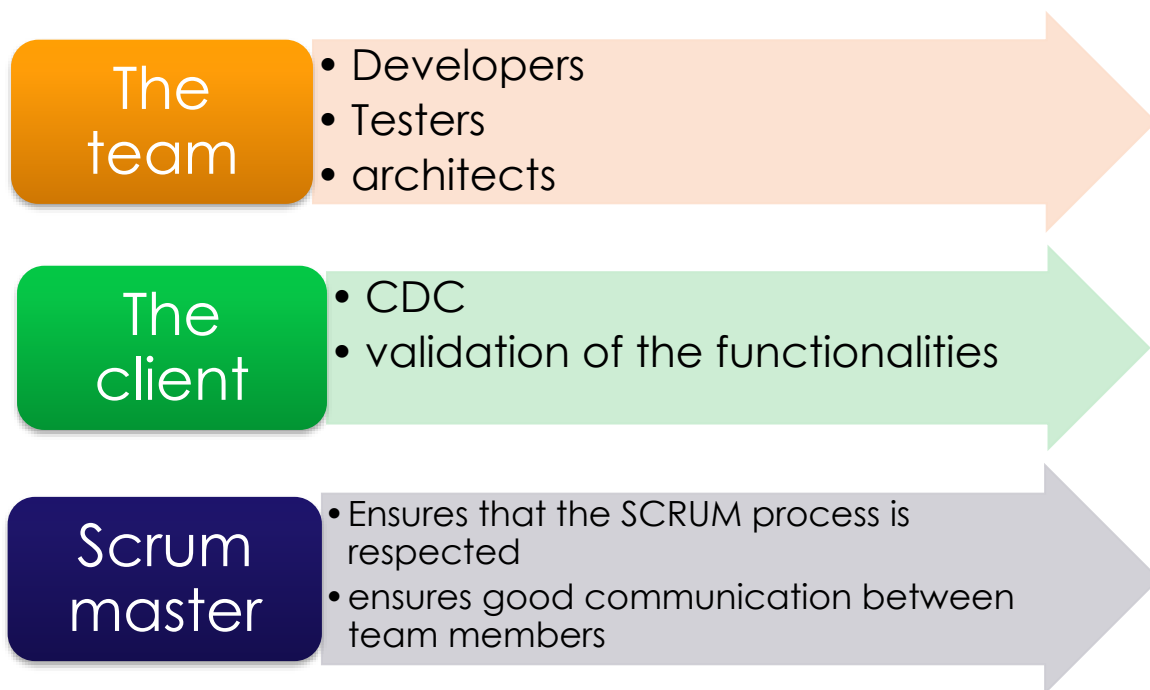
Project management

Méthode agile

We chose to use the agile method "SCRUM" in order to manage our project well and to be able to structure the ideas and their development over time.

Indeed, a classic project management imposes a static structure with three pillars : the deadline, the budget and the execution. If ever during the project a change occurs, it impacts all the parts and so the work must be redone.

The agile method is based on the following concepts:



SCRUM Process

It starts with a user story, which is a description of the user experience using the user's language.

The product backlog is the list of tasks to be done to achieve a user story.

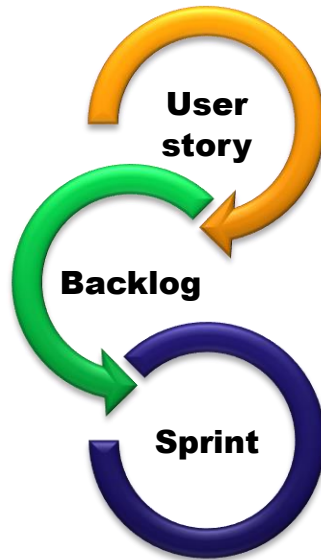
Once the team has agreed on the requirements backlog and the user stories, they proceed to the sprint.

A sprint starts with a planning meeting, during this step the team members choose the priority elements of the product backlog that will be developed in the sprints.

Meetings are held frequently to monitor the progress of the project.

After this meeting the Scrum master distributes the problems encountered to the team members and if the problems persist he communicates them to the client during another meeting.

Any change in the development is integrated in the product backlog.



Our user stories

| | |
|---|--|
| 1 | As a student, I want the system to detect my presence in the room and measure the temperature and turn on the heaters when it is cold |
| 2 | As a student, I want the system to detect my presence in the room and measure the temperature and turn on the air conditioning when it is hot. |
| 3 | As an administrator, I would like to retrieve the value of any presence/CO2 sensor of any rooms so that the system switches on the alarm (risk of fire departure) automatically. |
| 4 | As a teacher, I want the system to detect my presence in the room and measure the CO2 level and open the windows and doors in case of high CO2 level to ventilate. |
| 5 | As a teacher, I want the system to detect my presence and turn on the lights if the brightness value is low |
| 6 | As a teacher, I want the system to detect my absence and turn OFF the lights if the brightness value is high |
| 7 | As an administrator, I would like to get incident and activity reports so that I could do statistic about periodic events and incidents. |

Salle autonome



We chose to use the JIRA tool to apply the SCRUM method to our project.

This tool allowed us to create user stories and the backlog as well as the different sprints.

It also allowed us to distribute the tasks and to see the evolution of the project in real time.

The screenshot displays a JIRA project board for 'Salle autonome'. It features a list of user stories (PS-11 to PS-17) with their respective priorities, assignees, and statuses. Below the stories is a 'Créer un ticket' button. The 'Backlog' section shows three tickets: PS-19 (Test avec jenkins), PS-20 (Database creation), and PS-21 (Test avec node-red). The backlog also includes a 'Créer un sprint' button.

| User Story | Priority | Assignee | Status |
|---|----------|----------|---------|
| PS-11 As a student, I want the system to detect my presence in the room and measure the temperature and turn on the... | 3 | TG | TERMINÉ |
| PS-16 As a student, I want the system to detect my presence in the room and measure the temperature and turn on the... | 2 | TG | TERMINÉ |
| PS-12 As an administrator, I would like to retrieve the value of any presence/CO2 sensor of any rooms so that the syste... | 7 | FL | TERMINÉ |
| PS-15 As a teacher, I want the system to detect my presence in the room and measure the CO2 level and open the wind... | 6 | FL | TERMINÉ |
| PS-14 As a teacher, I want the system to detect my presence and turn on the lights if the brightness value is low | 5 | K | TERMINÉ |
| PS-13 As a teacher, I want the system to detect my absence and turn OFF the lights if the brightness value is high | 5 | K | TERMINÉ |
| PS-17 As an administrator, I would like to get incident and activity reports so that I could do statistic about periodic event... | 4 | FL | A FAIRE |

+ Créer un ticket

▼ Backlog (3 tickets) 0 0 0 Créer un sprint

| User Story | Status | Assignee |
|--------------------------|----------|----------|
| PS-19 Test avec jenkins | EN COURS | K |
| PS-20 Database creation | TERMINÉ | FL |
| PS-21 Test avec node-red | TERMINÉ | FL |

Micro services architecture

Overview

We were requested to develop a Web application (Proof-of-Concept) for managing INSA's rooms. This application must allow automatic closing windows, doors, turning on heating, turning off lights ... etc. This application relies on software services, sensors, and actuators. The goal is to retrieve data from sensors and analyze them to enable taking decisions.

Through software services (microservices), the application retrieves data of sensors (temperature, presence, ...), and according to the values of the retrieved data, actions on actuators can be triggered. The application is based on microservices and implements the followings scenarios.

Scenario

We imagined 7 scenarios using microservices from three points of view: Student, Teacher and Administrator. The first six scenarios were tested and worked fine. The test sequence is detailed later in the report.

Scenarios

1. As a student, I want the system to detect my presence in the room and measure the temperature and turn on the heaters when it is cold
2. As a student, I want the system to detect my presence in the room and measure the temperature and turn on the air conditioning when it is hot
3. As an administrator, I would like to retrieve the value of any presence/CO2 sensor of any rooms so that the system switches on the alarm (risk of fire departure) automatically
4. As a teacher, I want the system to detect my presence in the room and measure the CO2 level and open the windows and doors in case of high CO2 level to ventilate
5. As a teacher, I want the system to detect my presence and turn on the lights if the brightness value is low
6. As a teacher, I want the system to detect my absence and turn OFF the lights if the brightness value is high
7. As an administrator, I would like to get incident and activity reports so that I could do statistic about periodic events and incidents

Microservices created

Each sensor and actuator is a micro service and thus, we have identified 3 categories of microservices: Sensors, Actuators and others.

Microservices created

Sensors

Temperature
Presence
CO2-Humidity
Light

Actuators

AC - Heater
Doors
Windows
Alarms
Lights

Others

Controller
Database
Clock




We use the presence sensor to determine if we should activate an actuator or not. For instance, we won't turn on the lights if no one is in the room. Every scenario are led by the presence sensor.

For now, doors and windows are controlled together. We will give more details about how it works together in the next part.

The most important microservices is the controller that will managed every scenarios. We also have a clock that was supposed to give us readable timestamp to manage scenarios. Finally, we have the database microservice that will communicate with a MySQL database that we were supposed to use for a tests. Both last microservices have been tested and worked fined. Unfortunately, we did not had enough time to integrate them into our final solution.


Global Architecture

Each actuator is based on the following architecture:



```
1 private int id; //Unique id
2 private String batiment; //In which building is the sensor located
3 private String room; //Room of that building the sensor located
4 private int position; //Sensor's position within a room
5 private int state; //Does it still works
6 private type currentStatus; //Is the actuator activated or not
```

Each sensor is based on the following architecture:



```
1 private int id; //Unique id
2 private String batiment; //In which building is the sensor located
3 private String room; //Room of that building the sensor located
4 private int position; //Sensor's position within a room
5 private int state; //Does it still works
6 private int battery; //How much battery is left
7 private int value; //What was the last value registered
```

Presence Sensor

The presence sensor store a boolean value for « Is there someone in the room? ». It is central in our architecture and determine how every actuators work. Indeed, one of the first condition to enable AC for instance, is to check if someone is in the room. So in the next, explanation it will not be mentioned but keep in mind it is always use. So, if no one is in the room, we turn the lights and the AC-Heater off and close doors and windows.

Light Sensor

The light sensor store a percentage value where 0% means there is no light in the room and 100% is the maximum luminosity the sensor can detect. So here, if the luminosity is less that 50% we turn on the lights and if it is greater than 80% we turn them off. The zone between 50 and 80% is called the buffer zone. In this zone, we keep the last state. We use to avoid the lights to turn on and off every time we are close to the 50% threshold.

CO2-Humidity Sensor

The CO2-Humidity sensor store both the CO2 level and the humidity level of the room. The humidity value is not use in our project because we did not have enough time. It works exactly as the light sensor meaning that we have a threshold value with a buffer zone. Here, this sensor is used with three actuators. If someone is in the room and the CO2 level is too high we open all doors and windows of the room but close them if the room is empty. On the other hand, if the room is empty and the CO2 level is too high we trigger the alarm for the administrator.

Temperature Sensor

The temperature sensor store temperature of the room and acts on the heater or AC of the AC-Heater module. Here, we have two thresholds, one for the AC and one for the heater and thus, two buffer zones. When the AC is turned on, the heater is automatically turned off and vice-versa.

Controller

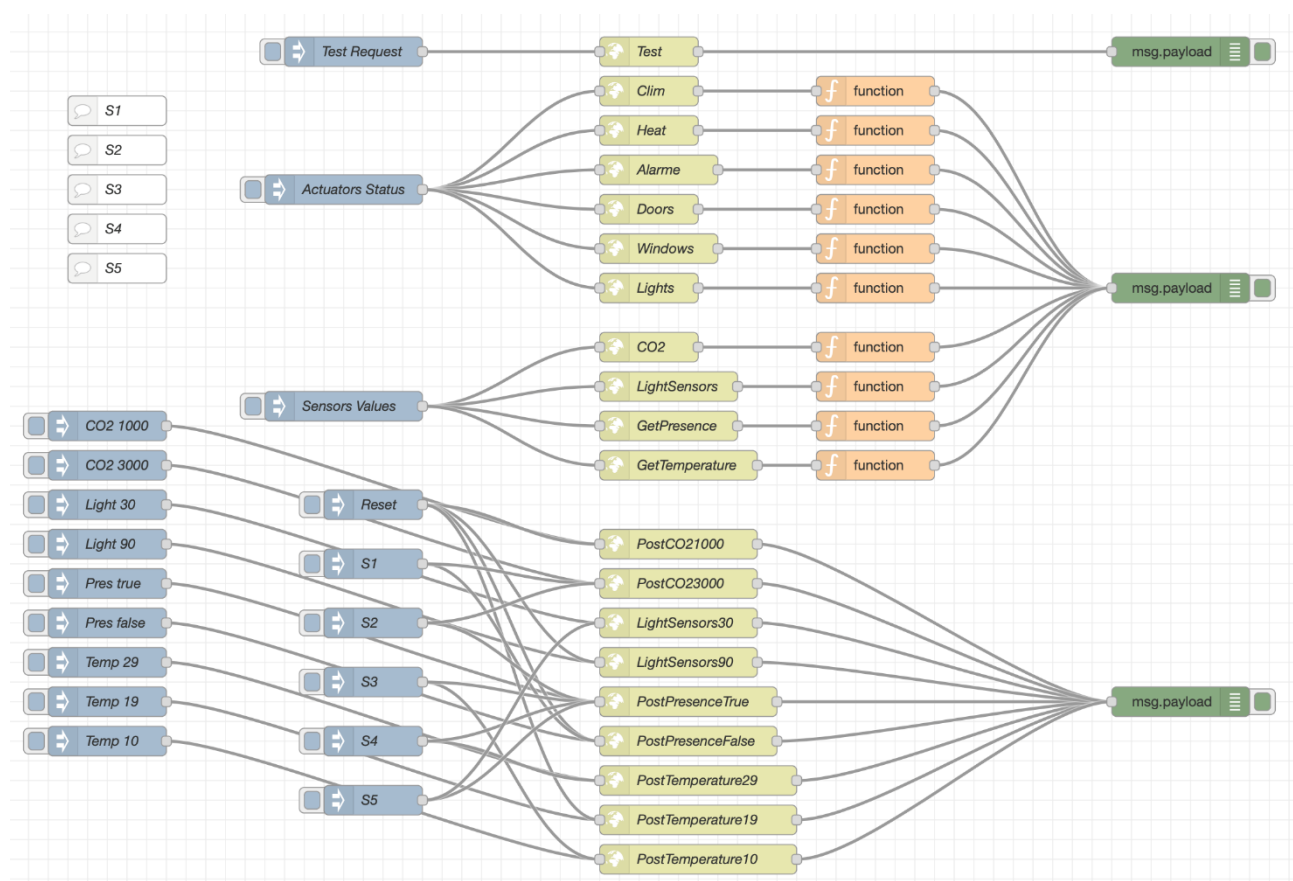
The controller centralizes all sensors and actuators in one place. Functions are created to create an element, retrieve the list of created elements and finally act on the actuators or retrieve the values of the sensors. All these functions use HTTP requests to exchange information. We also use the controller for our tests. We initialize a room and several sensors/actuators to test our scenarios.

Tests

We use a simple node-red flow to tests our scenarios. Between each scenarios we reset our sensors. We defined 5 scenarios in our flow:

- We set CO2 level to 3000ppm and the presence to false → it should trigger the alarm
- We set CO2 level to 3000ppm and the presence to true → doors and windows should be open
- We set the temperature to 10° and the presence to true → Heater should be on and AC off
- We set the temperature to 29° and the presence to true → AC should be on and heater off
- We set the lights sensors to 30% and the presence to true → lights should be on

Here is the flow used:



Finally, here you can watch how this flow work. The flow is also available in our GitHub repo.

Jenkins

Jenkins is an open source software which is used in server's automation, with Jenkins, we can do build, test and deploy our micro services on a container.



Project build with Jenkins Git

| Jenkins | | | | | | |
|---|---|-------------------|----------------|---------------|----------------|--|
| Tableau de bord | | | | | | |
| <div> <div> <div>Nouveau item</div> <div>Utilisateurs</div> <div>Historique des constructions</div> <div>Relations entre les builds</div> <div>Vérifier les empreintes numériques</div> <div>Administrer Jenkins</div> <div>Mes vues</div> <div>Créer une Vue</div> </div> <div> <div>File d'attente des constructions</div> <div>État du lanceur de compilations</div> </div> </div> | | | | | | |
| S | M | Nom du projet | Dernier succès | Dernier échec | Dernière durée | |
| ✓ | ☁ | AC | 19 mn - #2 | 22 mn - #1 | 20 s | |
| ✓ | ☁ | Alarm | 10 mn - #1 | s. o. | 19 s | |
| ✓ | ☁ | Cozum | 5 mn 39 s - #1 | s. o. | 23 s | |
| ✓ | ☁ | Contrôlier | 43 s - #1 | s. o. | 33 s | |
| ✓ | ☁ | Doors | 8 mn 39 s - #1 | s. o. | 24 s | |
| ✓ | ☁ | Lights | 6 mn 39 s - #1 | s. o. | 22 s | |
| ✓ | ☁ | LightSensor | 3 mn 39 s - #1 | s. o. | 26 s | |
| ✓ | ☁ | PresenceSensor | 2 mn 44 s - #1 | s. o. | 25 s | |
| ✓ | ☁ | TemperatureSensor | 1 mn 43 s - #1 | s. o. | 25 s | |
| ✓ | ☁ | Windows | 7 mn 39 s - #1 | s. o. | 26 s | |

We notice that all the builds are successful.

War file creation

After following the steps we got a packaging jar instead of a war and so we had to add a line in packaging that allows us to create a war and not a jar package.

```
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.7/commons-lang3-3.7.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/google/j2objc/j2objc-annotations/1.3/j2objc-annotations-1.3.jar (8.8 kB at 891 B/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/failureaccess/1.0.1/failureaccess-1.0.1.jar (4.6 kB at 456 B/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/28.2-android/guava-28.2-android.jar (2.6 MB at 260 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/google/errorprone/error_prone_annotations/2.3.4/error_prone_annotations-2.3.4.jar (14 kB at 1.4 kB/s)
[INFO] Replacing main artifact with repackaged archive
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 51.904 s
[INFO] Finished at: 2022-01-26T20:18:06+01:00
[INFO] -----
En attente que Jenkins finisse de récupérer les données
[JENKINS] Archiving C:\Users\User1\.jenkins\workspace\AC\AC\pom.xml to fr.insa.twf.actuators/AC/0.0.1-SNAPSHOT/AC-0.0.1-SNAPSHOT.pom
[JENKINS] Archiving C:\Users\User1\.jenkins\workspace\AC\AC\target\AC-0.0.1-SNAPSHOT.jar to fr.insa.twf.actuators/AC/0.0.1-SNAPSHOT/AC-0.0.1-SNAPSHOT.jar
channel stopped
Finished: SUCCESS
```



• • •

Unfortunately, once I got to the stage of launching Tomcat I had problems with the JRE HOME environment variable so I could not continue the deployment of the project.

https://github.com/florianleon/TP_SOA.git

https://github.com/florianleon/TP_SOA/blob/main/node-red/demo.mp4