# "Software Engineering"
# Course
# a.a. 2018-2019
## Template version 1.0
## Deliverable #1

**Lecturer: Prof. Henry Muccini (henry.muccini@univaq.it)**

# Dashboard Monitoraggio Ambientale

| | |
|---|---|
| **Date** | 03/12/2018 |
| **Deliverable** | 1 |
| **Team (Name)** | 5 Curly Brackets |



| Team Members | | |
|---|---|---|
| **Name & Surname** | **Matriculation Number** | **E-mail address** |
| Marco Ovidi | *247986* | *marco.ovidi@student.univaq.it* |
| Matteo Di Paolo | 243581 | *matteo.dipaolo@student.univaq.it* |
| Carlo Antonio Venditti | 248254 | *carloantonio.venditti@student.univaq.it* |
| Dima Mhrez | 247052 | *dima.mhrez@student.univq.it* |
| Andrea D'Angelo | 248817 | *andrea.dangelo6@student.univaq.it* |

# Project Guidelines

[do not remove this page]

*This page provides the Guidelines to be followed when preparing the report for the Software Engineering course. You have to submit the following information:*

- *This Report*
- *Diagrams (Use Case, Component Diagrams, Sequence Diagrams, Entity Relationships Diagrams)*
- *Effort Recording (Excel file)*

***Important:***

- ***document risky/difficult/complex/highly discussed** requirements*
- *document decisions taken by the team*
- ***iterate**: do not spend more than 1-2 full days for each iteration*
- ***prioritize** requirements, scenarios, users, etc. etc.*

# Project Rules and Evaluation Criteria

## *General information:*

- *This homework will cover the 80% of your final grade (20% will come from the oral examination).*
- *The complete and final version of this document shall be **not longer than 40 pages** (excluding this page and the Appendix).*
- *Groups composed of five students (preferably).*

*I expect the groups to submit their work through GitHub*

***Use the same file to document the various deliverable.***
***Document in this file how Deliverable "i+1" improves over Deliverable "i".***

## Project evaluation:

*Evaluation is not based on "quantity" but on "quality" where quality means:*

- *Completeness of delivered Diagrams*
- *(Semantic and syntactic) Correctness of the delivered Diagrams*
- *Quality of the design decisions taken*
- *Quality of the produced code*

# Table of Contents of this deliverable

## Sommario

**A) Requirements Collection**

A.1 Functional requirements 5
A.2 Non Functional Requirements 14
A.3 Excluded Requirements 14
A.4 Assumptions 14
A.5 Prioritization 15

**B) Software Architecture**

B.1 Static view of the system 16
B.2 Dynamic view of the system 17

C) **Design Decisions** **20**
D) **Effort Recording**

D.1 PERT Diagram 21
D.2 Statistics 22

# List of Challenging/Risky Requirements or Tasks

*<In this section, you should describe using the table below the most challenging or discussed or risky design tasks, requirements, or activities related to this project. Please describe when the risk arised, when and how it has been solved.>*

| Challenging Task | Date the task is identified | Date the challenge is resolved | Explanation on how the challenge has been managed |
|---|---|---|---|
| Identifying the right technologies | **19/11** | **30/11** | Testing performances on own computers or test servers. |
| Learning how to implement NetData efficiently | **19/11** | **//** | Studying a lot. |
| Using new DB technologies we're not familiar with | **23/11** | **//** | Studying a lot. |

# A. Requirements Collection

*In this section, you should describe both the application* ***features/functional*** *requirements as well as the* ***non functional*** *ones. You shall also document* ***constraints*** *and* **rules**, *if they apply.*

## *A.1 Functional Requirements*

- Unusual values must be shown with appropriate colours based on their priority
- Values over the defined threshold must be shown explicitly
- There are 3 types of manager
- At least 150.000 signals per minute must be handled
- Dashboard must show all sensors
- Managers must be able to change the defined threshold values
- The system must be able to detect sensors' failures and display a warning. If there are backup sensor for a measured property, the warning will have a lower priority
- Different managers are given information with different levels of detail, based on a hierarchic relationship
- The user must be able to select an area

### *A1.1 Use Case Diagrams*

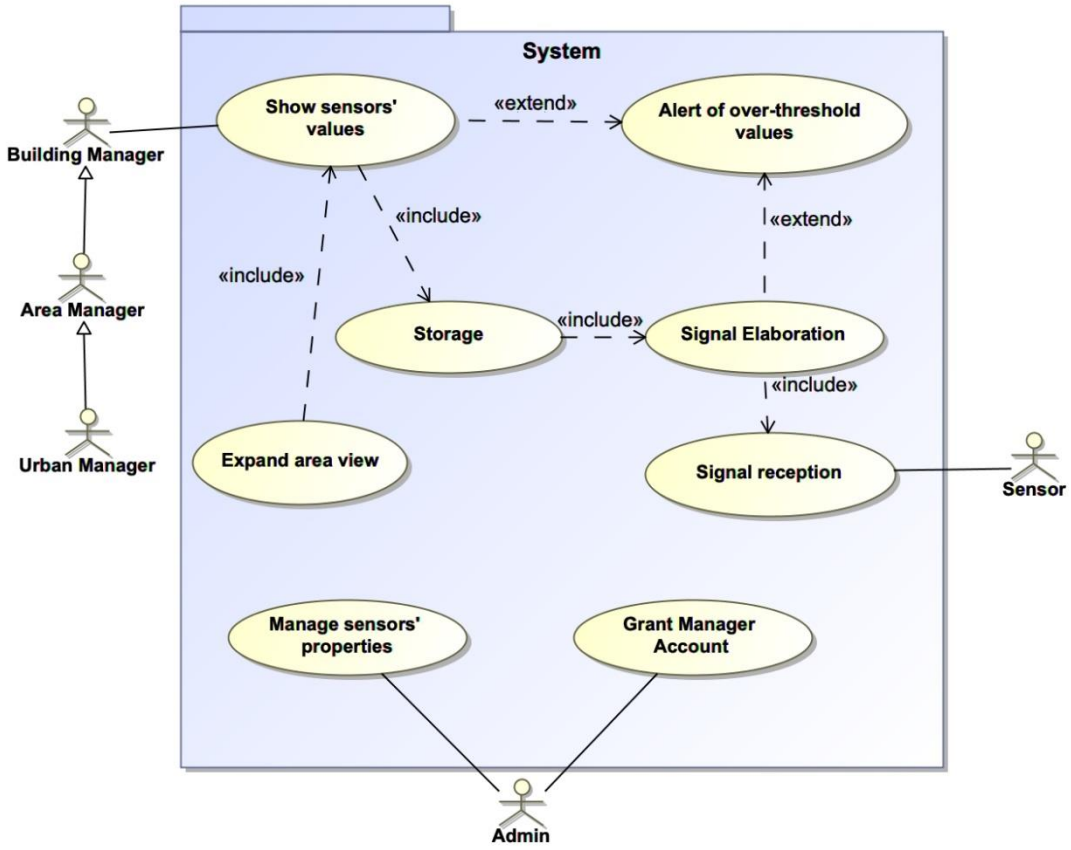



*Figure 1 : Use Case Diagram*

We identified two types of users, the managers and the admins.
Managers access the dashboard for environmental monitoring while admins can change threshold values of sensors and add new managers.

### *A1.2 Tabular description of the most relevant use cases*

| USE CASE 1 | Signal Reception | |
|---|---|---|
| Goal in Context | Signals from the sensors are received correctly. | |
| Preconditions | Sensors are installed and the system is running. | |
| Success End Condition | Values are received and processed. | |
| Primary actor | Sensor | |
| Trigger | No trigger | |
| DESCRIPTION | Step | Action |
| | 1 | Sensors send the detected values to the system. |
| | 2 | The system correctly receives and process those values. |

| RELATED INFORMATION | Signal Reception |
|---|---|
| Priority: | Very High |
| Performance | The system must be able to handle at least 150000 signals per minute. |
| Frequency | Every minute. |
| Superordinates | Signal Elaboration |

| USE CASE 2 | Signal Elaboration | |
|---|---|---|
| Goal in Context | Understand if the values are usual or above the threshold. | |
| Preconditions | Signal Reception is working correctly. | |
| Success End Condition | The correct values are sent to the client. | |
| Primary actor | Sensor. | |
| Trigger | The system receives the data. | |
| DESCRIPTION | Step | Action |
| | 1 | The system receives the data. |
| | 2 | The sensor that sent the values is identified and its set threshold values are found. |
| | 3 | The newly received values are compared to the thresholds. |
| EXTENSIONS | Step | |
| | 1a | Some values are over the threshold:<br>Alert of values over the threshold. |

| RELATED INFORMATION | Signal Elaboration |
|---|---|
| Priority: | Very high |
| Performance | Must be able to elaborate at least 150000 signals per minute. |
| Frequency | Every time a signal is received. |
| Superordinates | Storage |

| USE CASE 3 | Alert of over-threshold values | |
|---|---|---|
| Goal in Context | Make sure the Manager is alerted of the critical conditions | |
| Preconditions | Manager is logged in dashboard | |
| Success End Condition | Alert is given to manager in case of dangerous situations in a certain area | |
| Failed End Condition | System alerts Manager that it isn't able to detect dangerous situations | |
| Primary | Building Manager, Area Manager, Urban Manager | |
| Trigger | A value over threshold is detected | |
| DESCRIPTION | Step | Action |
| | 1 | A value over threshold is detected |
| | 2 | A warning is displayed |
| SUB-VARIATIONS | | Branching Action |
| | 1 | One value type over threshold is shown ORANGE |
| | 2 | Two value types over threshold are shown RED |
| Priority: | Very High | |
| Frequency: | Every time that an unusual value is detected | |

| USE CASE 4 | Storage | |
|---|---|---|
| Goal in Context | Signal values are correctly stored. | |
| Preconditions | Values have been received and handled correctly. | |
| Success End Condition | The system will know the last known values from that sensor. | |
| Primary actor | Sensor | |
| Trigger | A signal is received. | |
| DESCRIPTION | Step | Action |
| | 1 | The signal is received and handled. |
| | 2 | The signal is stored to complete the story of the values of a certain sensor. |

| RELATED INFORMATION | Storage |
|---|---|
| Priority: | High |
| Performance | Must be able to store at least 150000 signals per minute. |
| Frequency | Every time a signal is received. |

| USE CASE 5 | Show sensors' value | |
|---|---|---|
| Goal in Context | System shows sensors' value to the manager | |
| Preconditions | Manager is logged in dashboard | |
| Success End Condition | The Manager is correctly shown current sensors' value | |
| Failed End Condition | Error message | |
| Primary | Building Manager, Area Manager, Urban Manager | |
| Trigger | Manager Login | |
| DESCRIPTION | Step | Action |
| | 1 | Manager logs in dashboard |
| | 2 | System evaluates current sensors' value and label it with different colours considering the treshold value |
| | 3 | System shows elaborated data to Manager |
| EXTENSIONS | Step | Branching Action |
| | 1a | Values over the imposed threshold detected : Alert of threshold value |

| USE CASE 6 | Expand area view | |
|---|---|---|
| Goal in Context | Manager can see a section of the area of expertise in detail | |
| Preconditions | Manager is logged in and dashboard is showing data | |
| Success End Condition | Manager can evaluate detail detected from sensors of an area | |
| Failed End Condition | Manager can’t focus on a particular area | |
| Primary | Manager | |
| Trigger | Manager select an area of interest | |
| DESCRIPTION | Step | Action |
| | 1 | Manager logs in dashboard |
| | 2 | System shows entire area sensors’ value |
| | 3 | Manager selects a section of area |
| | 4 | System shows details of selected area |

| USE CASE 7 | Manage Sensors' Properties | |
|---|---|---|
| Goal in Context | Admin can be able to change threshold value fo each sensor | |
| Preconditions | Admin logged in dashboard | |
| Success End Condition | Admin have modified threshold values of sensors | |
| Failed End Condition | Error Message | |
| Primary | Admin | |
| Trigger | Admin clicks on settings panel in dashboard | |
| DESCRIPTION | Step | Action |
| | 1 | Admin logs in dashboard |
| | 2 | Admin opens setting panel |
| | 3 | Admin selects a sensor |
| | 4 | Admin change threshold value for the selected sensor |

| RELATED INFORMATION | Manage sensor properties |
|---|---|

## A.2 Non Functional Requirements

*1) Availability. The system shall always work, with close to no downtime. It is extremely important that managers are able to check the status of the sensors at any given time*

*2) Reliability. The information the system provides need to be accurate, precise and correct, based on the values the sensors are sending.*

*3) Security. It is important nobody, beside the authorized users, are able to log in or edit parameters.*

*4) Scalability. The system must be able to grow efficiently with new sensors being placed.*

## A.3 Excluded Requirements

*1) Emergency Management. The system does not provide any functionality to interact with emergencies (e.g., calling firefighters). It is only used to check the data the sensors are sending.*

## A.4 Assumptions

*1) We assume the sensors are already in place and their positioning is reasonable. A sensor placed near a fireplace would probably show unusual values when there is no need to. We assume the maximum rate at which a sensor sends data being 1 every 5 seconds.*

*2) "Alert" and "Warning" have very different meanings in our system.*

*We have an alert when one type of parameter (e.g.: temperature) has a value that is over the threshold.*

*A warning is issued when two or more types of parameters are above the threshold, and the manager must be alerted immediately.*

## *A.5 Prioritization*

Here is the list of the *functional* requirements in priority order:

1. Unusual values must be shown with appropriate colours based on their priority
2. Values over the defined threshold must be shown explicitly
3. There are 3 types of manager
4. At least 150.000 signals per minute must be handled
5. Dashboard must show all sensors
6. Managers must be able to change the defined threshold values
7. The system must be able to detect sensors' failures and display a warning. If there are backup sensor for a measured property, the warning will have a lower priority
8. Different managers are given information with different levels of detail, based on a hierarchic relationship
9. The user must be able to select an area

# B. Software Architecture

*<Report here both the static and the dynamic view of your system design, in terms of a Component Diagram, Class Diagrams and their related Sequence Diagrams >*

## *B.1 The static view of the system: Component Diagram*

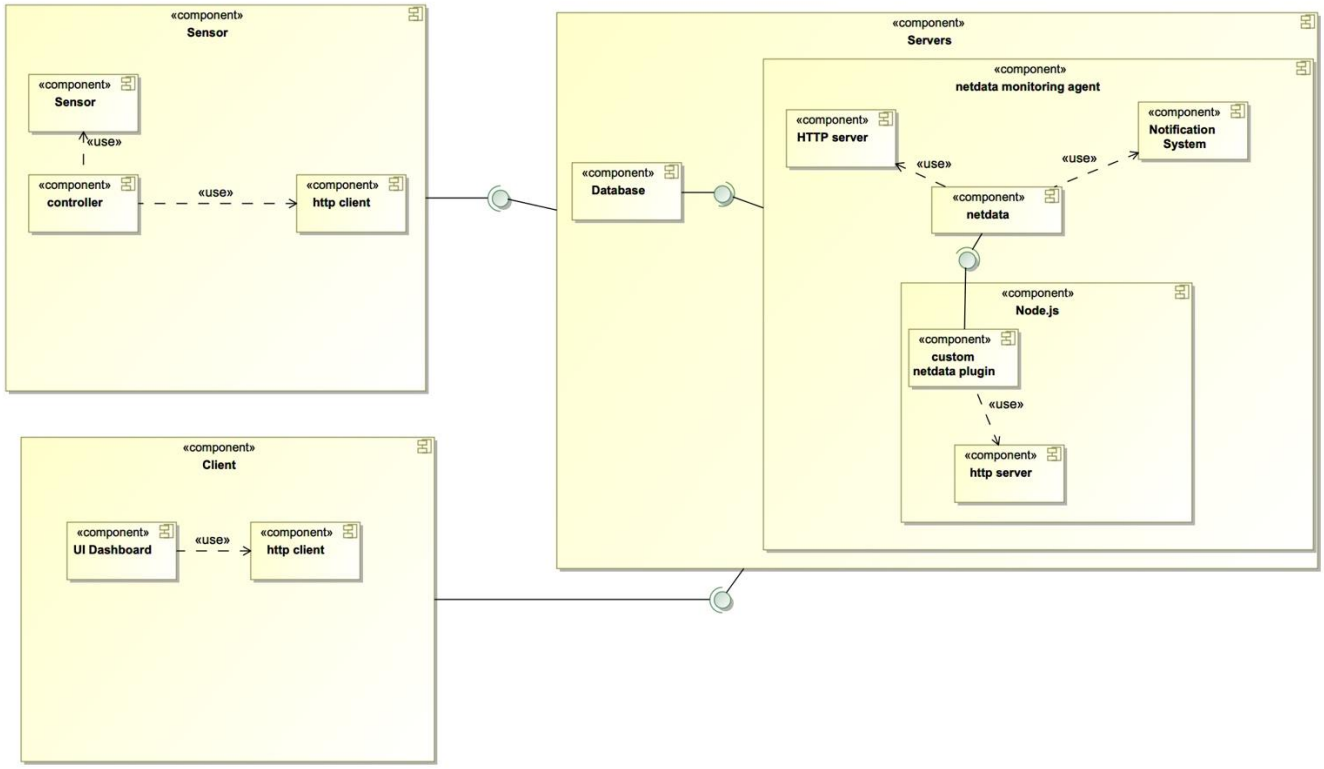



*Figure 2 : Component Diagram*

An overview of the entire system, showing the interfaces used by the various network nodes to communicate and the importance of the decision to use netdata as a foundation for the system. The interactions are simplified to show the logical subdivision without too much technical details.

## *B.2 The dynamic view of the software architecture: Sequence Diagram*

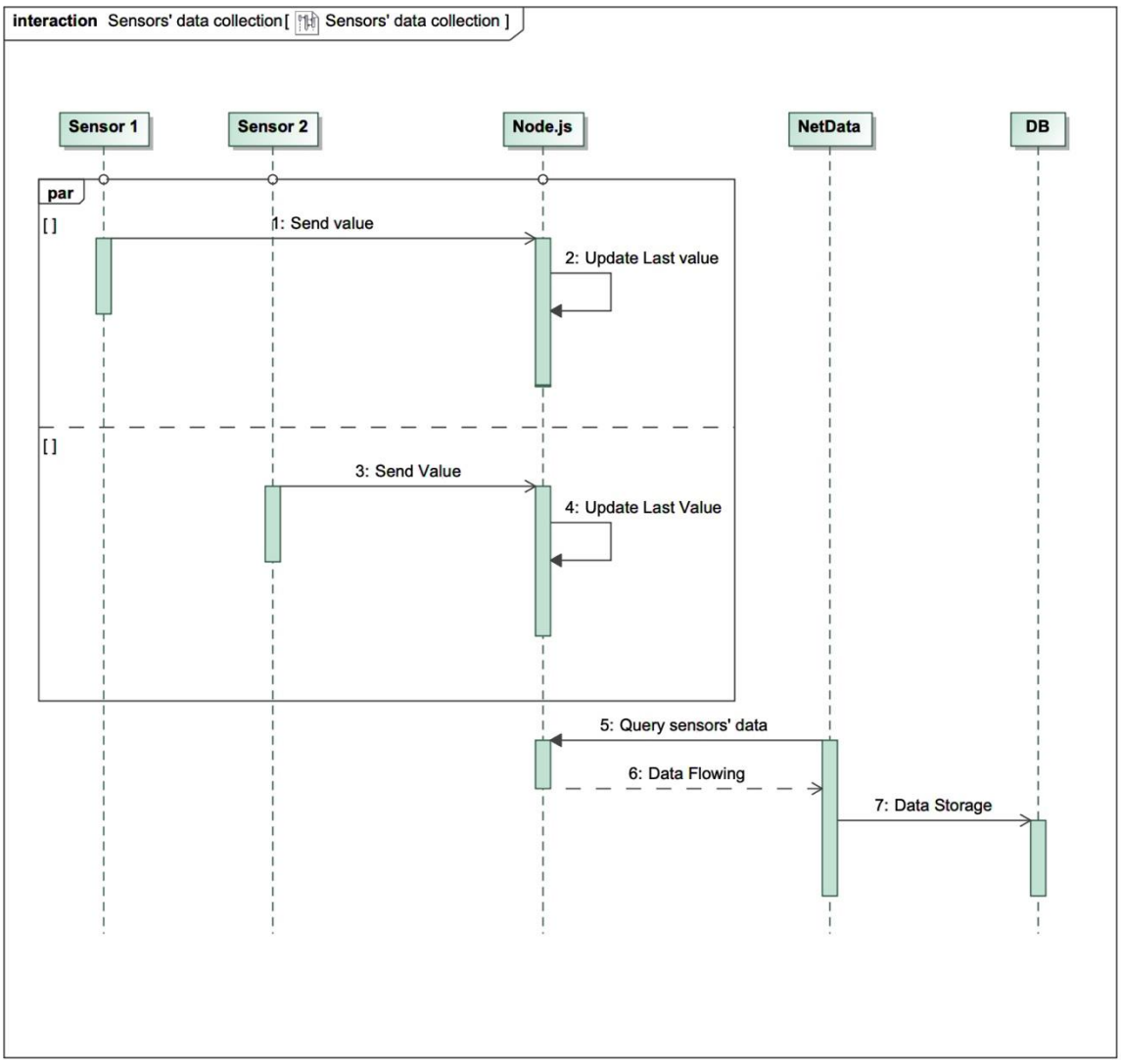



*Figure 3 : Sequence Diagram Sensors' data collection*

This diagram shows the parallel data collection from the sensors, focusing on how the node.js asynchronously receives data from the sensors and makes this available to netdata.
The plugin stores in memory the last value received from each sensor (which is updated every time it gets a new value) and every 5 seconds netdata queries for a new value.

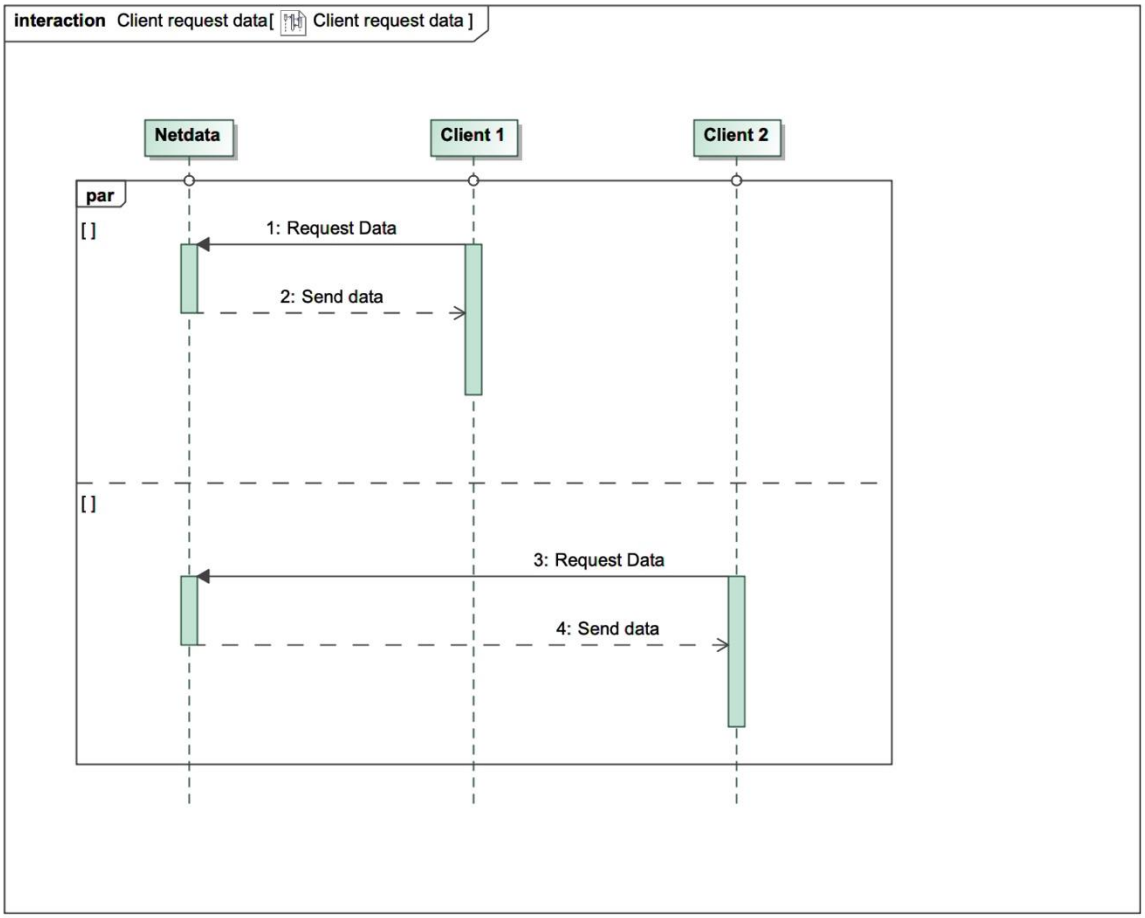



*Figure 4 : Sequence Diagram Users request Data*

The clients are dynamic web pages that interface with the netdata integrated web server. Communication between server and clients is done using HTTP.

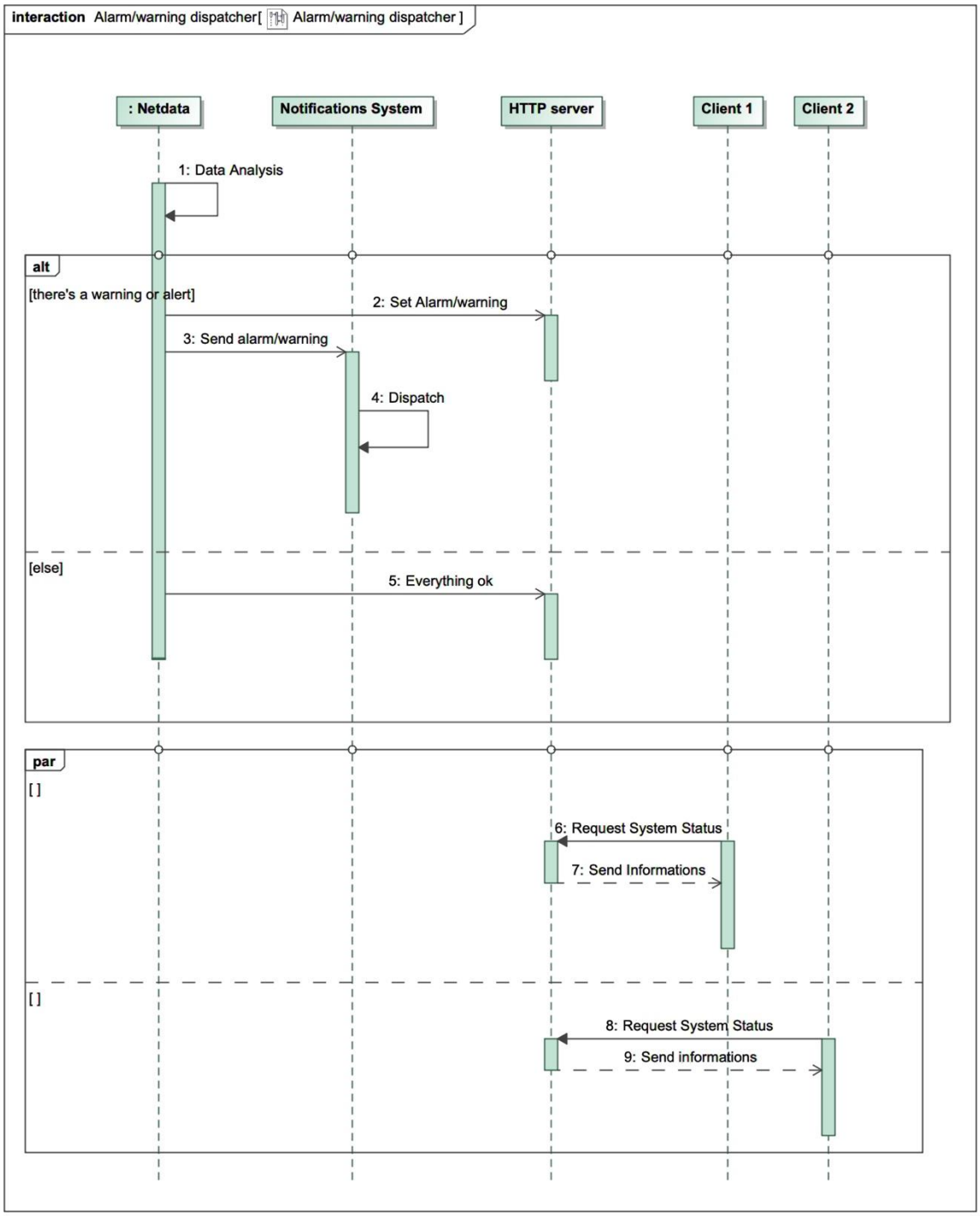



*Figure 5 : Sequence diagram warning/alert dispatcher*

netdata has an integrated notification system that allows us to send alarm and warning not only to the web clients but also using email, irc or even Slack and Telegram (and many others). The notification system is modular and should work out of the box with minimal configuration from our side.

# C. Design Decisions

1- netdata Usage

The netdata open sysm monitoring tool records and shows data efficiently and is visually appealing. It also satisfies all of our needs, so we chose to use it as the main part of our system.

2- Time Series Database

This choice was made in order to maximize efficiency, even though we're mostly knowledgeable about MySQL.

3- HTTP Protocol for the sensors

We tested HTTP request handling both locally and with remote servers with the tool Vegeta and a simple Rust HTTP server. The performance exceeded the requirements by far and we found no need to use a custom protocol. Using HTTP is easy and guarantees good enough performance.

4- Node.js for the netdata Plugin

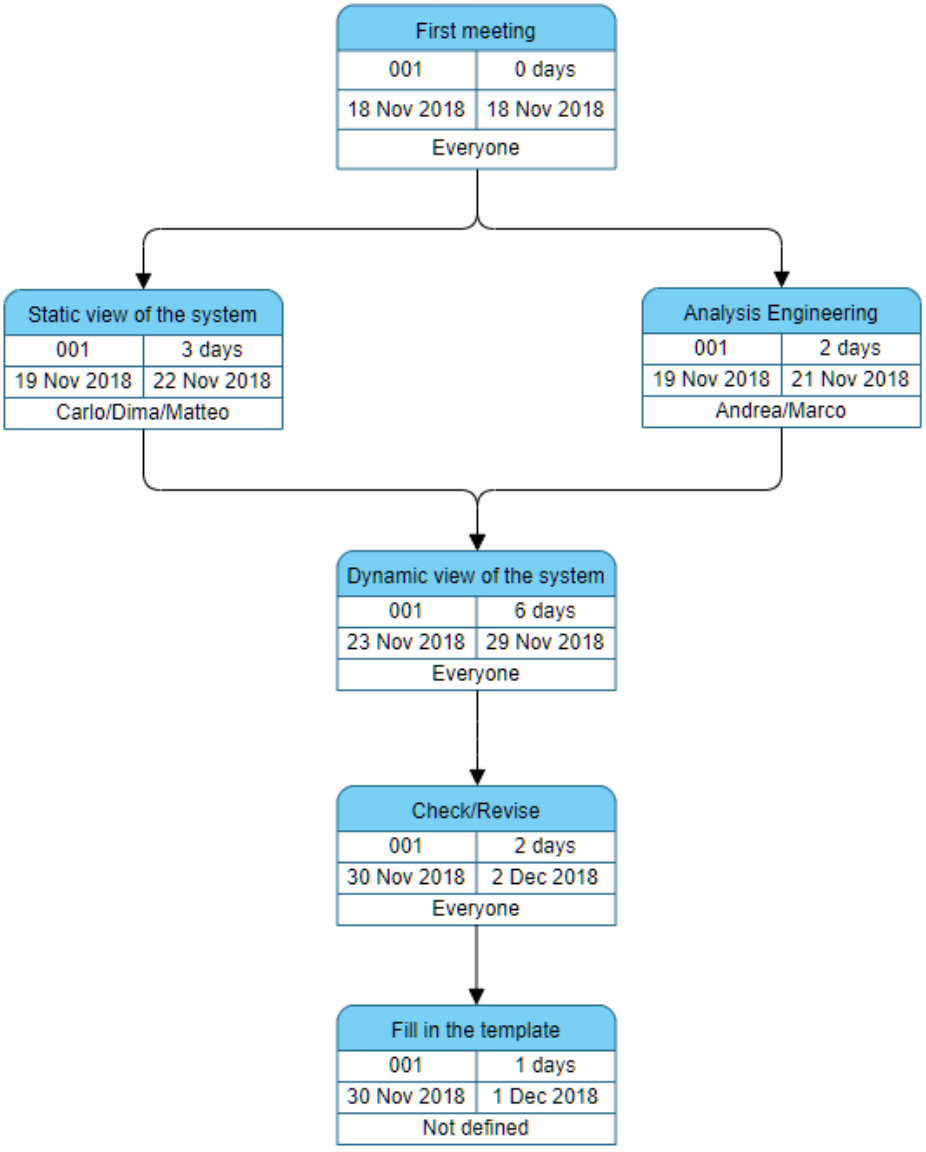
We chose node.js because it is JavaScript based and we all have some basic notions of JavaScript and are willing to learn more.
A simple API for netdata node.js plugins exists and it should be very performant for asynchronous and not CPU-intensive workloads (as it is in our case). Anyway, more specific performance testing is needed.

5- Admin user

We also chose to insert an Administrator as a separate user from the managers, in order to keep the parameters control and edit permissions completely separated from the "normal" users of the system.

# D. Effort Recording

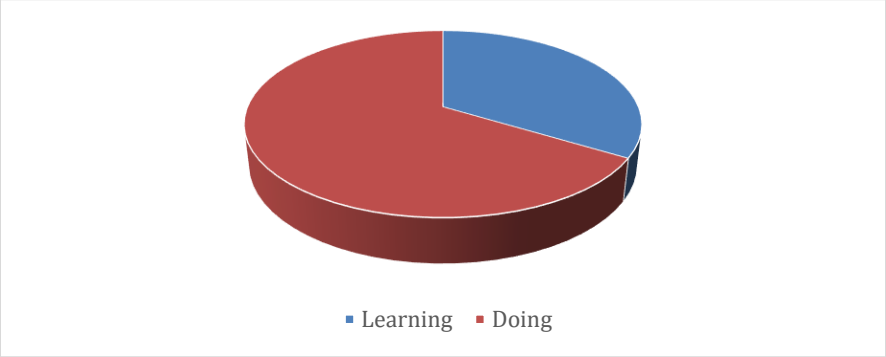
***PERT***



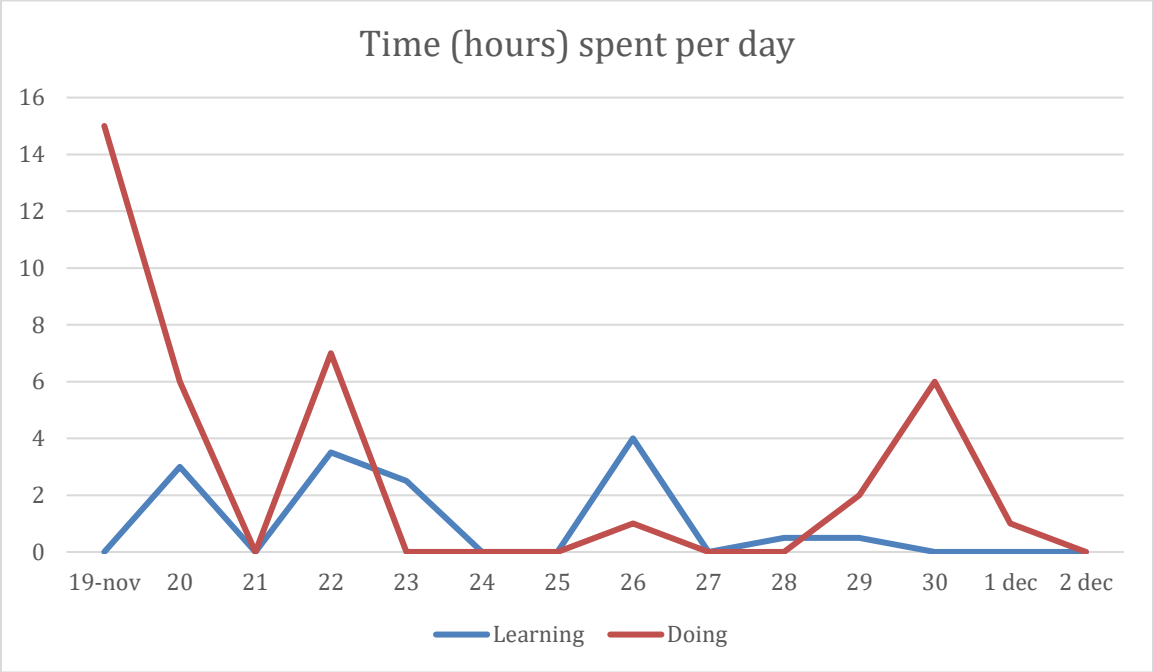
***Summary Statistics***

***Time Spent for LEARNING: 18.5 hours***
***Time spent for Doing: 36.5 hours***
***Total time spent: 55 hours***

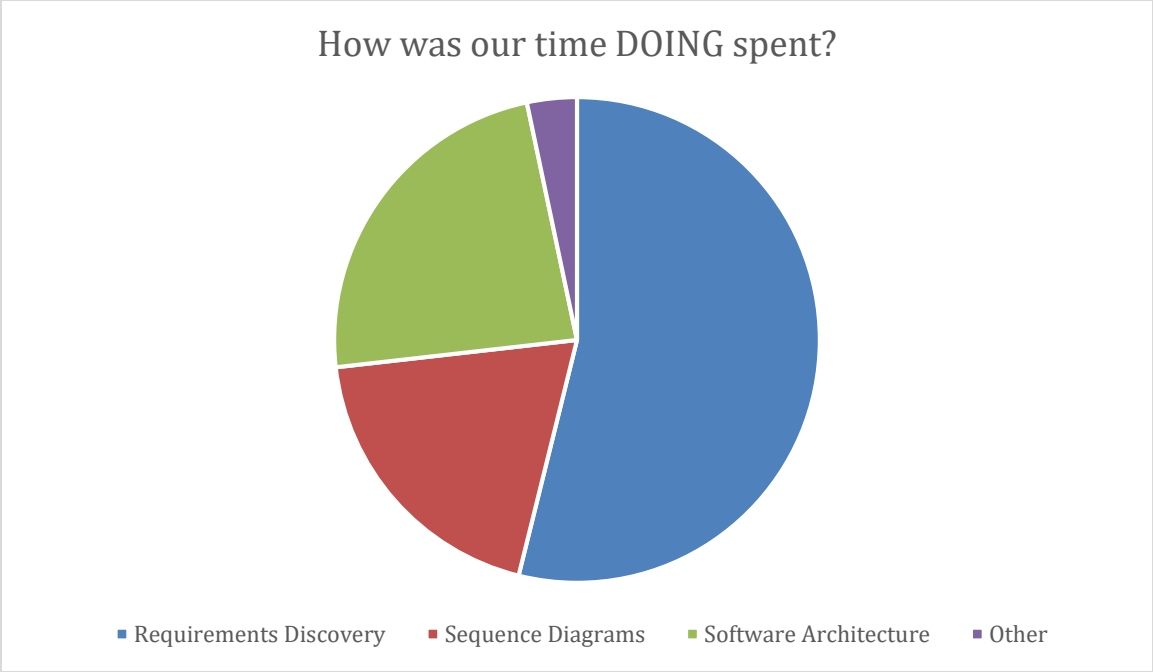
## *Time Partitioning*




66.36% of the time spent on the project was spent on DOING.
33.64% of the time spent on the project was spent on LEARNING.




We need to stop working with "bursts" of efficiency and find a more consistent schedule and line of work.




From this diagram, and the PERT diagram, we notice how we overestimated the necessary time to make Sequence Diagrams and grossly underestimated the time necessary for the Requirements Discovery.
It is also true that the Requirements Discovery is biased by the Initial Planning we all participated in (15 person-hours) , but the statement still stands and it's something we have to consider. The making of scenarios and the deep understanding of the requirements took more time than we initially considered.

## *Pictures' index*

Figure 1 : Use Case Diagram6
Figure 2 : Component Diagram16
Figure 3 : Sequence Diagram Sensors' data collection17
Figure 4 : Sequence Diagram Users request Data18
Figure 5 : Seuqence diagram warning/alert dispatcher19