

Multi-sensor data analysis to support nursing home care management

Andreu Ortega Blasi

In partial fulfilment of the requirements for the degree in Menció en Sistemes Electrònics at Unieversitat Politecnica de Catalunya

Supervisors: Hannelore Strauven and Miguel Jesús García Hernández

Katholieke Universiteit Leuven
June 2018



Abstract

This thesis focuses on incontinence nursing homes residents. As the society is aging as the population studies show, new problems are conceded: the older the population becomes, the more nursing homes are needed. One of the nursing home main care tasks is the management of the residents incontinence.

Currently, the incontinence management is done manually by caregivers of the nursing home, with the delays and unnecessary checks that this process entails. So the focus of this thesis is to automatize this process with a monitoring system to support the incontinence management of the nursing homes.

This project has previous stages: the sensor module that measures ambient parameters such as temperature, humidity or gas concentration, needed to evaluate that incontinence cases, is already developed.

This thesis will focus on study the raw sensor data and convert it in a useful data for monitoring incontinence. As a first part, this raw data will be ordered, classified and plotted in graphs. Afterwards, a characterization of the sensors in the module will be applied in order to obtain the conversion rates that will be used to convert this data, that comes from the output of the microcontroller, into real ammounts of gas concentration. Also the measurements of the sensors need to be calibrated by programming to avoid the effects of external factors. Finally the results obtained in this thesis will be studied and compared with a smart incontinence application that is already on the market.

Resum

Aquest TFG es centra en la incontinència dels residents de llars d'avis. A mesura que la societat envelleix, com mostren els estudis de població, es necessiten més residències d'ancians. I una de les tasques principals d'una residència d'ancians és la gestió de la incontinència dels seus pacients.

Actualment, el maneig de la incontinència es realitza manualment pels cuidadors dels residents, amb les demores i les comprovacions innecessàries que aquest procés implica. Llavors, l'enfocament d'aquesta tesi és automatitzar aquest procés i fer un sistema de monitorització per donar suport al maneig de la incontinència en les llars d'avis.

Aquest projecte té etapes prèvies: el mòdul sensor que mesura paràmetres ambientals com la temperatura, la humitat o la concentració de gas, necessaris per avaluar els casos d'incontinència, ja està desenvolupat.

Aquest TFG es centrarà en estudiar les dades que provenen del mòdul del sensor per convertir-los en dades útils per a la monitorització de la incontinència. Com a primera part, aquestes dades sense processar s'ordenaran, classificaran i representaran en gràfics. Posteriorment, es farà una caracterització dels sensors en el mòdul per obtenir les taxes de conversió que s'utilitzaran per convertir aquestes dades. que provenen de la sortida del microcontrolador, en quantitats reals de concentració de gas. També cal calibrar les mesures dels sensors mitjançant programació per evitar els efectes de fets externs. Finalment, els resultats obtinguts en aquesta tesi seran estudiats i comparats amb una aplicació de detecció intel·ligent de la incontinència que ja està en el mercat.

Resumen

Esta tesis se centra en la incontinencia de los residentes de hogares de ancianos. A medida que la sociedad envejece, como muestran los estudios de población, se necesitan más residencias de ancianos. Y una de las tareas principales de una residencia de ancianos es la gestión de la incontinencia de sus pacientes.

Actualmente, el manejo de la incontinencia se realiza manualmente por los cuidadores de los residentes, con las demoras y las comprobaciones innecesarias que este proceso implica. Entonces, el enfoque de esta tesis es automatizar este proceso y hacer un sistema de monitoreo para apoyar el manejo de la incontinencia en las residencias de ancianos.

Este proyecto tiene etapas previas: el módulo sensor que mide parámetros ambientales como la temperatura, la humedad o la concentración de gas, necesarios para evaluar los casos de incontinencia, ya está desarrollado.

Esta tesis se centrará en estudiar los datos que provienen del módulo del sensor para convertirlos en datos útiles para la monitorización de la incontinencia. Como primera parte, estos datos sin procesar se ordenarán, clasificarán y representarán en gráficos. Posteriormente, se aplicará una caracterización de los sensores en el módulo para obtener las tasas de conversión que se utilizarán para convertir estos datos, que provienen de la salida del microcontrolador, en cantidades reales de concentración de gas. También es necesario calibrar las medidas los sensores mediante programación para evitar los efectos de hechos externos. Finalmente, los resultados obtenidos en esta tesis serán estudiados y comparados con una aplicación de detección inteligente de incontinencia que ya está en el mercado.

*I would like to dedicate this thesis to my family and friends that
ever trusted in me.*

*Also I would like to dedicate this thesis to everyone that has suffered
incontinence patterns. I feel identified with them because I had
incontinence problems related with milk protein and fructose
intolerance.*

Acknowledgements

First and foremost I would like to thank my supervisor Hannelore Strauven, for advising and helping me with the thesis. She helped me to obtain laboratory acces and also showed the nursing home where the measurements were done. Also thanks to e-Media Research Lab of the Faculty of Engineering Technology for being the promotors of this thesis and the university of KU Leuven for letting me to use their chemistry laboratory.

I would like to thank as well to my co-supervisor in Spain, Miguel Jesús García Hernández for helping me with the correction of the written documents of my thesis.

Also I would like to thank to my friends and family. In special I would like to thank my parents, who supported me to do this thesis and Erasmus; my uncle Francesc Blasi who gave me ideas for the laboratory experiment; my friend Oscar Muñoz, who checked and corrected the document of my thesis and my erasmus friends who were close in stressful moments.

Revision history and approval record

Revision	Date	Purpose
0	20/11/2018	Document creation
1	10/01/2019	Document Revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Andreu Ortega	aortegablas@gmail.com
Hannelore Strauven	hannelore.strauven@kuleuven.be
Miguel Jesus Garcia Hernandez	miguel.j.garcia@upc.edu

Written by:		Reviewed and approved by:	
Date	20/11/2018	Date	10/01/2019
Name	Andreu Ortega	Name	Hannelore Strauven
Position	Project Author	Position	Project Supervisor

Contents

Abstract	I
Resum	II
Resumen	III
Dedication	IV
Acknowledgements	V
Revision history and approval record	VI
Contents	IX
List of figures	XI
List of tables	XII
1. Introduction	1
1.1. Situation	1
1.2. Statement of purpose	2
1.3. Requirements and specifications	3
1.4. Methods and procedures	4
1.4.1. Structuring the data and plotting it	4
1.4.2. Characterization and calibration of the sensors used	5
1.4.3. Comparison with the product already on the market	6
1.5. Work plan	6
1.6. Existing solutions	9
2. State of the art of the technology used or applied in this thesis	10
2.1. Programming with Python	10
2.2. Electronic module	10
2.2.1. Microcontroller	11
2.2.2. Electrochemical sensor	11
2.2.3. Semiconductor sensor	11
2.2.4. Ammonia sensors	12
2.2.5. Hydrogen sulfide sensors	12
2.2.6. Control sensors	12

2.3.	Conversion Rates	13
2.3.1.	The incubator	13
3.	Methodology / project development	14
3.1.	Structuring the data and plotting it	14
3.1.1.	Structuring the data	14
3.1.2.	Plotting the data	15
3.2.	Conversion rates and external factors compensation	17
3.2.1.	The experiment	17
3.2.2.	Decreasing the concentration of ammonia in a solution	20
3.3.	Market application comparison	21
3.3.1.	Visual comparison	23
3.3.2.	Mathematical comparison	23
4.	Results	25
4.1.	Structuring the data and plotting it	25
4.1.1.	Structuring the data	25
4.1.2.	Plotting the data	27
4.2.	Conversion rates	28
4.2.1.	Experiment results	28
4.3.	Market application comparison	32
4.3.1.	Visual comparison	32
4.3.2.	Mathematical comparison	35
5.	Conclusion	37
5.1.	General Conclusion	37
5.2.	Future work	37
5.2.1.	Improvements to the sensor module	37
5.2.2.	Another improvements	38
Annexes		A-1
A.	Data Structure	A-1
A.1.	Code to structure the data	A-1
A.2.	Script flow diagram	A-2
A.3.	Script results	A-3
B.	Data graphs	B-4
B.1.	Code to plot the data	B-4
B.2.	Id identification	B-6
B.3.	Plot all the sensor module data	B-7
B.4.	Focused plot	B-10
B.5.	Script flow diagram	B-11
B.6.	Existing solutions	B-13
B.7.	Code to digitalize Tena Identifi results	B-15
B.8.	Plotting the digital values of Tena identifi	B-16
C.	Mathematical comparison	C-17
C.1.	Comparison script	C-17
C.2.	Script to select the data intervals	C-20

C.3.	Script to find nearest data	C-21
C.4.	Script to select the data in the range required	C-22
C.5.	Script to plot the comparison	C-23
C.6.	Comparison script flow diagram	C-24

List of Figures

1.1.	An example of the Data_DB including the values of some sensors	4
1.2.	An example of the pdf_info	5
1.3.	Tena Identifi Sensor Wear	9
2.1.	Python programing enviroment	10
2.2.	Particle Photon microcontroller	11
2.3.	Heraeus Incubator	13
3.1.	Update to the structuring script with the time addition	15
3.2.	Experiment Notes	19
3.3.	Example for testing the experiment	19
3.4.	Placement of both modules with ammonia solutions inside the incubator	20
3.5.	Equipment for the dilution process	21
3.6.	Excel with the Tena Identifi relevant values	22
4.1.	Local folder where all the measurements files are placed	25
4.2.	First sensor values of the new Excel	26
4.3.	Last sensor values of the new Excel	26
4.4.	Execution of the update	26
4.5.	Local folder where all the plot files are placed	27
4.6.	An example of the plotting results	27
4.7.	An example of the focused plot	28
4.8.	Experiment result with 0ppm and 25 ppm	29
4.9.	Experiment result with 50ppm and 75ppm ammonia dilutions	30
4.10.	Experiment result with 100ppm, 150ppm and 200ppm ammonia dilutions	31
4.11.	Tena Plot with all the values	32
4.12.	John Connor module measurement values	33
4.13.	Kate Brewster module measurement values	33
4.14.	Kyle Reese module measurement values	34
4.15.	Ambient parameters measured	34
4.16.	Result of the comparison between the ammonia sensor MICSNH3 and the tena levels	35
4.17.	Result of the comparison between the ammonia sensor SGXNH3 and the tena levels	35
4.18.	Result of the comparison between the ammonia sensor MQS-137 and the tena levels	36

1.	Code to create new Excel with the information provided in other Excels	A-1
2.	Flow diagram of the data structuration	A-2
3.	Code to structure the data with the outputs once executed	A-3
4.	Code to plot the data values	B-5
5.	Code to separate all the data from the different sensor modules	B-6
6.	Code to plot the data values	B-9
7.	Code to plot only the desired sensor values	B-10
8.	Flow diagram of the data plotting	B-12
9.	Tena measurement example	B-13
10.	Tena measurement example	B-14
11.	Code to digitalize the Tena Identifi values	B-15
12.	Code to plot the Tena Identifi values	B-16
13.	Code to compare the applications	C-19
14.	Code to select the data intervals	C-20
15.	Code to find the nearest data	C-21
16.	Code select the data in the range required	C-22
17.	Code to obtain the traces of the desired plot	C-23
18.	Flow diagram of the data comparison	C-24

List of Tables

1.1. Tables with all the main tasks of this thesis described	8
1.2. Milestone with all the tasks	8
2.1. Basic characteristics of ammonia sensors	12
2.2. Basic characteristics of hydrogen sulfide sensors	12

Chapter 1

Introduction

1.1. Situation

Studies show that the population is aging (Lorant, 2005), so in the future, more nursing homes will be needed to take care of the old people. The ratio of the global population and the working population is decreasing, for the same reason. This means that in the future, more old people will need to take care of them and, in proportion, less workers will be.

The process to monitorized incontinence patterns in residents of nursing homes is done manually and it has a lot of delays and unnecessary checks. So, in the future, nursing homes will be very busy if they keep with this manual process and will need a faster system to monitorize all the residents incontinence in order to help better them.

The goal of this project is automatize the entire process of incontinence management in the nursing homes, making it faster, easier to control and more comfortable for the resident due to the benefits of the better response of an automatic system against manual systems. Knowing the incontinence profile of each resident and providing incontinence care at the right moments will be easily done with this system. Looking it with perspective, this thesis will be strongly useful and needed in the near future.

The way to do it is by creating a sensor module to gather, from an incontinence pad, data and store it in a software platform where will be managed to find incontinence patterns. Also this patterns will give advises to the caregivers to change the patient incontinence pad at the right moment.

The sensor module should have as little direct impact on the resident as possible. The dispositive should not be wearable or implantable, so this is why is placed near the resident patient bed in his room of the nursing home. This mean that the module needs to measure gas concentrations from the ambient to detect urine or stool from the resident.

At the moment this thesis is started, there are already parts of the project done: the sensor module that gathers ambient parameters such as temperature, humidity and gas concentration, transforms into raw data and sends it to a database is already built. Also there is a mobile application created for controlling the results of the sensors measurements (Simon Kupers, 2019).

As the urine is a solution mainly based on ammonia (Putnam, 1971) and stool with hydrogen sulfide, this were the gases measured ¹.

At this point, this thesis starts and tries to convert this raw data into useful data for managing incontinence by the next steps: structuring, plotting it, applying the conversion rates and comparing them with the smart incontinence application Tena Identifi, that will be further detailed in this thesis.

This thesis is developed at the e-Media Research Lab of the Faculty of Engineering Technology (E-media), at the university of KU Leuven, in Leuven, Belgium. It has one developer, Andreu Ortega, that is a student of IT engineering specialized in electronics in the UPC, from Barcelona, Spain. It also has two supervisors, Hannelore Strauven from KU Leuven and Miguel Jesus Garcia Hernandez from UPC.

1.2. Statement of purpose

The objectives of this thesis are described in the following list:

- Extract the raw data from the database and classify it by exporting it in different Excels containing the data of each measurement related with different test in residents from the nursing home.
- Use this classified data and plot it to observe the performance of the sensors. This will allow to observe the overall sensor response to evaluate its usefulness in a certain measurement.
- Do the sensors characterization and apply the conversion rates between the input and the output values of the sensors. This will convert the digital values of the microcontroller into values of real gas concentration, temperature...
- Compensate the deviation that the gas sensors have due to external facts. This will compensate the effects, in the sensor measurements, of external factors.
- Evaluate the results by comparing it with an application already in the market. So this thesis will prove the level that this project has against other existing applications, comparing them and observing similarities and differences.

The way to achieve this objectives will be described in this chapter as methods and procedures. The application of this methods will be explained in chapter 3 and their results in chapter 4.

¹This explanations about the module decisions already done can be found in the previous thesis (Simon Kupers, 2019).

1.3. Requirements and specifications

As this thesis has different stages, each stage has its own requirements and specifications.

Requirements and specification of the data structuration:

- The data placed in the new file, must contain the same aspects as the data in the database: this means that this data must have the measurement sensor values, the timestamp from the measurement, the id of the sensor and the resident measured...
- Also it needs to be fixed the time shifting from the time in the mcu and the real time of the measurements.
- This data also must be in the intervals of time that involves the group of measurements. Any data from another time interval will be a part of another group of measurements.

Requirements and specification of the data structuration:

- The observation of the performance of the measurements though time has to be clear, so graphs must be easy to understand by eye.
- This graphs must show the performance of all 9 available sensors that are working in the module (some of them are implemented but not working, so is not necessary to observe them).

Requirements and specification of the characterization of the sensors:

- The conversion rates shown in each gas sensor datasheet must be implemented. Also it is needed to obtain extra information to compensate external facts effect (because in the datasheets isn't explained).
- If there is a lack of information, a detailed test should be done in order to characterize the sensors, relating the gas concentration measured and the output values from the mcu obtained.

Requirements and specification of the comparison with the application in the market:

- The pictures containing the values of this application must be digitalized by creating an Excel with the same information and after using it in a program to manage it.
- Data compared must be the sensor gas concentration measurements against the incontinence levels of that application.

1.4. Methods and procedures

The project of the entire system to manage incontinence has a wide variety of different aspects to fulfill, all related with the each other. The data used in this thesis comes from the database connected with the online platform of the micro-controller. This data is the result of detecting gas concentration by the sensors and digitalizing this values by the microncontroller in order to make available to work with them by software. The database is written in MySQL which makes it possible to export the data to csv files ² that have all the measurements that have been done since this project started.

Focusing on this thesis, the main work is programming in Python. Several packages with their functions are used to develop the entire program. The most important are: Pandas (Pandas), that allows the user to manage data by dataframes; and Plotly (Plotly) , that allows the programmer to plot the data in different ways.

So, for structuring the data the main package used will be Pandas. Afterwards, Plotly will allow to plot the data to observe the performance of all the sensors.

Also, the program needs to apply the compensation of the external facts that influence the gas sensors. So first the method to compensate the data will be developed and after it will be programmed as a script of the entire program.

Another part of the program will be the script that compares the results obtained in this thesis Tena Identifi results, not only observing the results in the graphs, but also with mathematical tools.

1.4.1. Structuring the data and plotting it

The data from the database is exported to an excel file named Data_DB. that contains each measurement with the output values of all the sensors measured at the certain time of the measurement:

id	timestamp	resident_id	device_id	nr_of_measurements	MICSNH3	MICSNO2	MICSCO	SGXNH3	SGXH2S	SPH2SH2S	SPH2STEMP
160	04/03/18 15:38	1360059000f51353532343635		1	3NULL	NULL		1	4	1766	1939
161	04/03/18 15:38	1360059000f51353532343635		1	3NULL	NULL		2	5	1829	1939
162	04/03/18 15:38	1360059000f51353532343635		1	2NULL	NULL		2	4	1878	1938
163	04/03/18 15:38	1360059000f51353532343635		1	3NULL	NULL		2	4	1910	1938
164	04/03/18 15:41	1360059000f51353532343635		1	3NULL	NULL		1	4	1872	1922
165	04/03/18 15:41	1360059000f51353532343635		1	3NULL	NULL		2	5	1911	1922
166	04/03/18 15:41	1360059000f51353532343635		1	3NULL	NULL		2	4	1948	1922
167	04/03/18 15:41	1360059000f51353532343635		1	3NULL	NULL		2	4	1969	1922
168	04/03/18 15:41	1360059000f51353532343635		2	4NULL	NULL		3	5	1991	1921
169	04/03/18 15:41	1360059000f51353532343635		2	4NULL	NULL		2	4	2023	1921
170	04/03/18 15:41	1360059000f51353532343635		5	4NULL	NULL		2	4	2040	1920
171	04/03/18 15:41	1360059000f51353532343635		5	4NULL	NULL		2	5	2046	1920
172	04/03/18 15:41	1360059000f51353532343635		5	4NULL	NULL		1	4	2049	1919
173	04/03/18 15:42	1360059000f51353532343635		5	5NULL	NULL		2	4	2045	1919
174	04/03/18 15:42	1360059000f51353532343635		5	4NULL	NULL		3	4	2045	1919
175	04/03/18 15:42	1360059000f51353532343635		5	5NULL	NULL		2	5	2047	1919
176	04/03/18 15:42	1360059000f51353532343635		5	5NULL	NULL		1	5	2050	1920
177	04/03/18 15:42	1360059000f51353532343635		5	5NULL	NULL		1	4	2053	1922
178	04/03/18 15:43	1360059000f51353532343635		5	5NULL	NULL		1	4	2054	1922
179	04/03/18 15:43	1360059000f51353532343635		5	5NULL	NULL		2	5	2056	1923
180	04/03/18 15:43	1360059000f51353532343635		5	4NULL	NULL		1	4	2053	1925
181	04/03/18 15:43	1360059000f51353532343635		5	5NULL	NULL		1	4	2049	1927
182	04/03/18 15:43	1360059000f51353532343635		5	4NULL	NULL		2	4	2046	1929
183	04/03/18 15:43	1360059000f51353532343635		5	4NULL	NULL		3	5	2049	1930
184	04/03/18 15:44	1360059000f51353532343635		5	4NULL	NULL		3	5	2056	1933
185	04/03/18 15:44	1360059000f51353532343635		5	5NULL	NULL		0	4	2052	1935
186	04/03/18 15:44	1360059000f51353532343635		5	5NULL	NULL		1	5	2045	1938
187	04/03/18 15:44	1360059000f51353532343635		5	4NULL	NULL		1	5	2052	1939
188	04/03/18 15:44	1360059000f51353532343635		5	5NULL	NULL		2	5	2055	1941
189	04/03/18 15:45	1360059000f51353532343635		5	5NULL	NULL		1	4	2046	1943

Figure 1.1: An example of the Data_DB including the values of some sensors

²All the data in the database is sorted by time.

Each row of the excel, related with each measurement, has the following data: the id related with the measurement, the date when the measurement was done, the id of the resident that was with the sensor module, the module sensor id, because there is more than one sensor module; and the digital values obtained from the microcontroller that are related with the sensor measurements. Also there is another excel, that will be named pdf_info, that contains the interval of time that a focused group of measurements had been done, where is written the interval of time of each group of measurements.

id	start	end	remarks	device_id	MICSNH3	SGXNH3	SGXH2S
68	2018-03-27 09:19:51	2018-03-27 09:30:04	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
69	2018-03-27 09:30:12	2018-03-27 09:35:52	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
70	2018-03-27 09:36:00	2018-03-27 09:53:41	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
71	2018-03-27 09:53:49	2018-03-27 09:58:14	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
72	2018-03-27 09:58:22	2018-03-27 10:14:57	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
73	2018-03-27 10:15:05	2018-03-27 10:20:12	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
74	2018-03-27 10:20:20	2018-03-27 10:37:11	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
75	2018-03-27 10:37:19	2018-03-27 10:42:59	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
76	2018-03-27 10:42:59	2018-03-27 10:51:16	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
77	2018-03-27 14:09:13	2018-03-27 14:25:31	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
78	2018-03-27 14:25:39	2018-03-27 14:31:19	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
79	2018-03-27 14:31:35	2018-03-27 14:49:57	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
80	2018-03-27 14:50:06	2018-03-27 14:55:04	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
81	2018-03-27 14:55:37	2018-03-27 15:10:46	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
82	2018-03-27 15:10:54	2018-03-27 15:16:21	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
83	2018-03-27 15:16:29	2018-03-27 15:23:55	Lab measurement 1 NH3	380054000d51353532343635	1	1	1
84	2018-03-29 18:01:39	2019-03-21 14:56:28	Test measurement 7	380054000d51353532343635	1	0	0
85	2018-03-29 22:44:33	2018-03-29 23:20:32	Test measurement 7	340019001147343000000000	1	0	0
86	2018-03-29 22:44:33	2018-03-29 23:13:18	Test measurement 7	380054000d51353532343635	1	0	0
87	2018-03-29 23:22:54	2018-03-29 23:49:07	Test measurement 7	440060000d51353532343635	1	0	0
88	2018-03-29 23:21:52	2018-03-29 23:48:13	Test measurement 7	32002f000b51353432383931	1	0	0
89	2018-03-29 23:50:02	2018-03-30 03:02:55	Test measurement 7	380054000d51353532343635	1	0	0
90	2018-04-01 14:53:56	2018-04-02 15:15:43	Toilet measurement 6	380054000d51353532343635	1	1	1
91	2018-04-02 15:26:11	2018-04-03 17:31:38	Toilet measurement 7	380054000d51353532343635	1	1	1
92	2018-04-04 19:46:54	2018-04-05 00:33:01	Test measurement 8	32002f000b51353432383931	1	1	1
93	2018-04-04 19:46:54	2018-04-05 00:33:01	Test measurement 8	440060000d51353532343635	1	0	0
94	2018-04-04 19:46:54	2018-04-05 00:33:01	Test measurement 8	340019001147343000000000	1	1	1
95	2018-04-06 17:38:04	2018-04-09 04:30:09	Toilet measurement 8	380054000d51353532343635	1	1	1

Figure 1.2: An example of the pdf_info

So the first part of this thesis was considering these intervals described in pdf_info, create new Excels separating the different group of measurements, and save them in a local folder with the title corresponding to the measurements that are related.

In some cases, it will be more that one cell in the same group of measurements, so it needs to be considered the range from the start date of the first cell to the end date from the last cell.

In the pdf_info there are a lot of different group of measurements but mainly this thesis will work with the lab measurements and Tena measurements, that are the measurements done in the lab and with the residents of the nursing home.

After, all the data must be plotted, and these plots should be stored in a local folder in order to monitoring the variations of the gas during the time the measurements were done.

1.4.2. Characterization and calibration of the sensors used

The sensor module converts the gas concentrations into digital values that the software platform can use them. So, in order to observe the data as real gas concentration values instead of digital values, a conversion rate has to be applied to convert the data into gas concentrations.

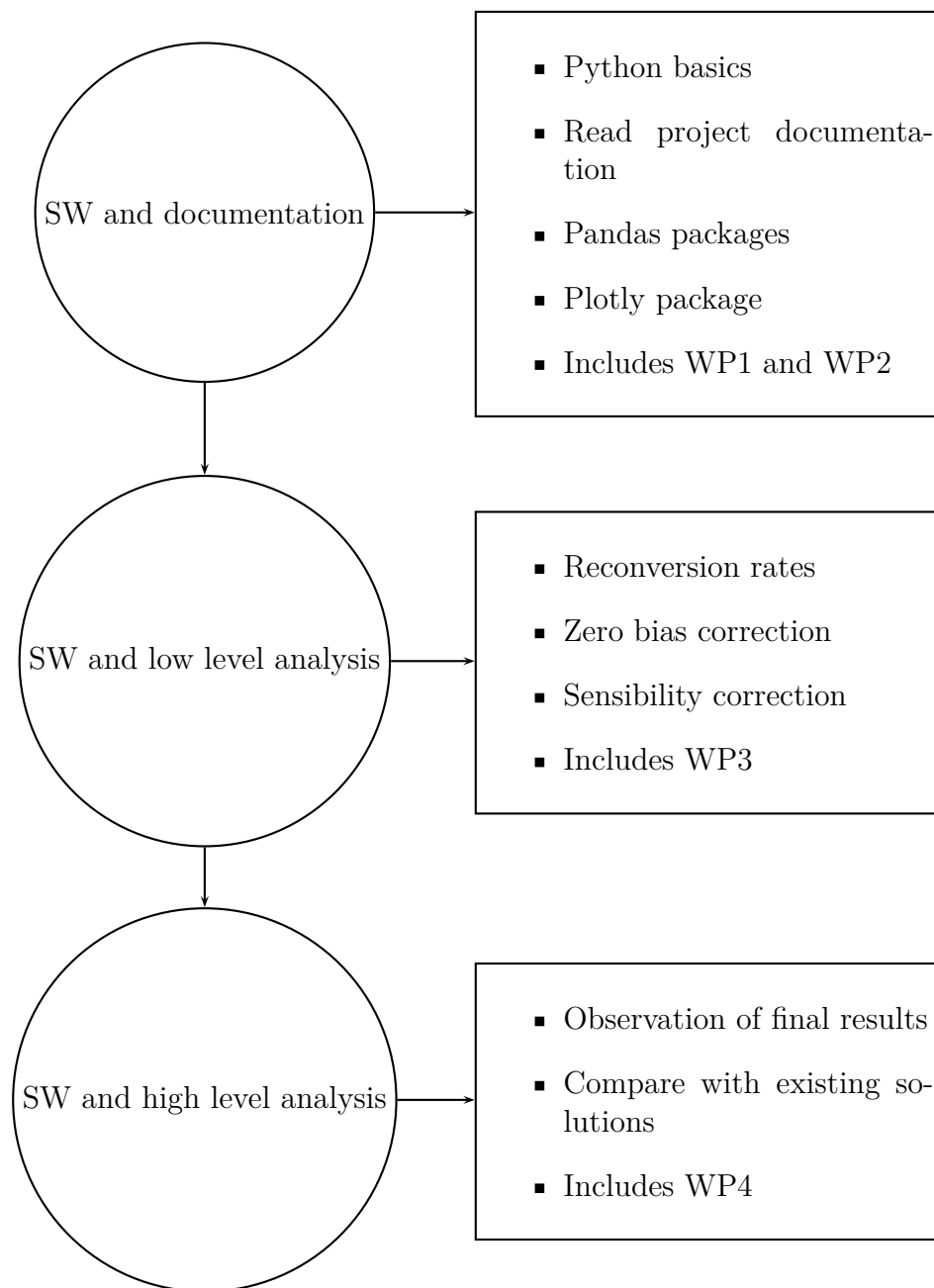
Also the sensors output is influenced by external factors, such as temperature, pressure or relative humidity. So they must be characterized and calibrated in order to obtain the compensated and correct values.

1.4.3. Comparison with the product already on the market

Finally, a comparison between the results of this thesis with the results of Tena Identifi measurements will be done, visually by comparing the plots of their results in time and mathematically trying to prove these first impressions.

1.5. Work plan

For a fast understanding of this thesis work a short work plan diagram is displayed:



And now more focused in extended tables:

Project: Multi-sensor data analysis	WP ref: 1
Major constituent: Software and documentation, structure raw data.	Sheet n of m
Short description: Learn the basics of Python whereas reading and understanding the already existing documentation of the project. After, learn Python focused in data managing and plotting.	Start date: 24/09/2018 End date: 26/10/2018
	Start event: initial package End event: WP2
Internal task T1: Python basics Internal task T2: Read project documentation Internal task T3: Pandas packages Internal task T4: Plotly package	Deliverables: Program that structure and plots the data.

Project: Multi-sensor data analysis	WP ref: 2
Major constituent: Software and analysis 1	Sheet n of m
Short description: Using the tools of Python learned, focus on low level data analysis in order to remark interesting events to study afterwards in a high level	Start date: 26/10/2018 End date: 22/01/2019
	Start event: WP1 End event: WP3
Internal task T1: Characterize the gas sensors. Internal task T2: Analyze in a low level data: compensate the external facts that influence the performance of the gas sensors.	Deliverables: Program that applies the conversion rates and the compensation studied

Project: Multi-sensor data analysis	WP ref: 3
Major constituent: Software and analysis 2	Sheet n of m
Short description: Using the tools of Python learned, focus on extracting some conclusions about the performance of the electronic sensor module and compare our data to the data of smart incontinence sensor wear	Start date: 23/11/2018 End date: 23/01/2019
	Start event: WP2 End event: Final package
Internal task T1: Compare the data obtained with the incontinence material TENA; compare the performance of our project and the TENA performance.	Deliverables: Data analysis document with incontinence episodes

Project: Multi-sensor data analysis	WP ref: 4
Major constituent: Writing report	Sheet n of m
Short description: Once I finish the work with data I need to fulfill the thesis report.	Planned start date: 21/12/2018 Planned end date: 15/01/2019
	Start event: WP2 End event: final package
Internal task T1: Finish the thesis report	Deliverables: Thesis report

Table 1.1: Tables with all the main tasks of this thesis described

The reader will notice here that the two last packages started and finished at the same time. This is because the final steps of the comparison were done once studied the characterization of the sensors to take better conclusions. The available lab hours to do WP3 were very limited so that why this work plan has been followed.

Once everything is described in the work packages it can be summarized in a milestone:

WP#	Task#	Short title	Milestone / deliverable	Date (week)
1	1	Structure the data	Program that structures the data and plots it	5
2	2	Analyze in a low level the data	Data has to be cleaned of things that don't work well in order to proceed with out study	After classes
3	3	Analyze in a high level the data	Data analyzed has to be related with the data of smart incontinence sensor wear	After classes
4	4	Finish the thesis report	Thesis report	After classes

Table 1.2: Milestone with all the tasks

1.6. Existing solutions

The idea of monitoring the incontinence of residents in nursing homes has already been tested and there is one application available in the market. This system is called Tena Identifi (Identify) that consists in a software platform that manages a disposable that absorbs incontinence product with a reusable logger. Tena Identifi tracks resident's voiding patterns in real time, by humidity sensors, called Tena Identifi Sensor Wear. The process to monitoring the incontinence needs that the residents wear the sensor, where the humidity sensors are placed. A reusable logger gathers data during 72h through this sensors and sends it to a software platform where it is used to generate incontinence reports. After, the logger needs to be cleaned with non corrosive disinfectant.



Figure 1.3: Tena Identifi Sensor Wear

The cost of the Tena Identifi Sensor Wear is about 142,70€, that includes 12 disposables with the sensors, at the 'Thuiszorgwinkel' (Thuiszorgwinkel, 2019), this price does not include the loan of the logger the use of the software platform, which is 1500€ per year.

The level of saturation is displayed by 9 levels, from level 0 to level 8, a low resolution discretization about the saturation measured by the sensors. So small changes are not detected by the final user. Also with the Tena Identifi is very difficult to control the slow loss of urine spread over a longer amount of time.

Another problem that Tena Identifi has is that it does not measure the stool loss, so is not a complete incontinence monitoring system. Also it does not provide a real time feedback.

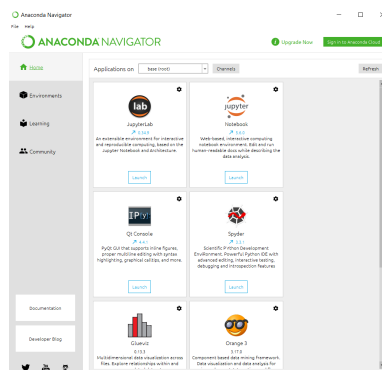
Also the values obtained by the Tena Identifi are affected by another facts, suchs as the change of the logger, if the resident goes to the bathroom... So they are also attached and are important to consider. This documents can be consulted in Annex B.6.

Chapter 2

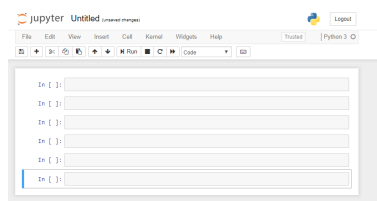
State of the art of the technology used or applied in this thesis

2.1. Programming with Python

Python is and easy to learn programming language, so that is why it is used for developing a wide variety of projects. Python 3.7 is the version used in this thesis and Anaconda is the Python launcher this thesis will use, with the Jupyter Notebook as a enviroment for programming.



(a) Anaconda launcher



(b) Jupyter programming environment

Figure 2.1: Python programming environment

So with this environment not only is it very easy to work line by line and easy to know where is the programming error, but also can be compiled and executed all together so the process to test the program can be done as fast as another environment such as Spider can be. So, this environment, is perfect for new people users interested in learning Python.

2.2. Electronic module

In this section the electronic module used for measuring ambient parameters will be explained. As it is mentioned before, this module was built. This section

is focused on explaining the sensors and microcontroller integrated in the sensor module and their main aspects.

2.2.1. Microcontroller

The microcontroller used in this module is the Particle Photon (Particle, 2019). It is a 32-bit controller, 3,3V working voltage and also WI-FI module, very useful to send the data recorded by sensors to the database. It has 8 digital ports and 8 analog ports. The analog-to-digital (A/D) inputs, of 12 bits resolution each one (0-4095) will have a big and important impact in the calculus in order to convert back from the output values to gas concentration. It's price is 38€.

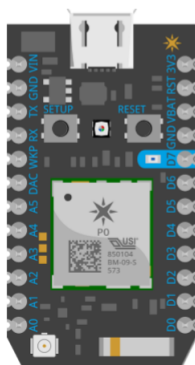


Figure 2.2: Particle Photon microcontroller

2.2.2. Electrochemical sensor

Electromechanical sensors (SGX-Sensortech, 2017) detects gases by its interaction with an electrolyte, producing a current proportional to the concentration of the gas. This sensors have, typically, three pins that are called electrodes: the counter, the sensing and the reference electrode.

The current obtained, as is very small, is converted to voltage by a trans-impedance amplifier.

2.2.3. Semiconductor sensor

Semiconductor sensors detects gases by it sensing layer, composed of a metal oxide, that is heated. When a gas is surrounding the sensor, some chemical components of the gas are absorbed by the surface of the sensing layer, changing the electrical conductivity of the layer, so this means a change of its electrical resistance.

The conversion rate applied by this sensors are calculated by comparing the relative change of the sensing resistance (R_S) against the baseline resistance (R_0). This baseline resistance is the preheated resistance value when, in the surroundings of the sensor, there is no gas.

$$R = \frac{R_S - R_0}{R_0} \quad (2.1)$$

2.2.4. Ammonia sensors

After some characteristics discussion done by the previous thesis, the chosen sensors to measure ammonia are described in the following table:

Sensor Model	Measurement range (in ppm)	Price (in €)	Power consumption (in mW)
SGX-4NH3	0-100	75,00	390
MiCS-6814	1-500	11,04	100
MQS-137	5-500	30,00	900

Table 2.1: Basic characteristics of ammonia sensors

2.2.5. Hydrogen sulfide sensors

Following the same procedure, the chosen sensors to measure hydrogen sulfide are described in the following table:

Sensor Model	Measurement range (in ppm)	Price (in €)	Power consumption (in mW)
3SP-H2S	0-50	16,96	0,05
SGX-7H2S	0-50	38,04	390

Table 2.2: Basic characteristics of hydrogen sulfide sensors

2.2.6. Control sensors

Also in the module there are more sensors: a temperature sensor LM35, that the output voltage changes linearly with the temperature; the DHT22 sensor, that is a digital temperature and humidity sensor; a motion sensor, IRA-S210ST0, a pyroelectric infrared sensor; and a light sensor NSL-19M51, that is a light dependent resistor or LDR. The motion sensor needs a signal filtering and amplification circuit, both circuit and sensors brought together to one single board, IMX-70. The motion sensor will help to detect if the resident is moving or not and the light sensor is used for detecting if it is night or day. From this sensors, only the ambient parameter sensors, such as temperature or relative humidity, will be used as a support for our tests.

2.3. Conversion Rates

2.3.1. The incubator

For measuring the conversion rates and compensating the external factors of the sensor module, an experiment was developed that will be further explained in the chapter 3. To do that experiment, an environment to control the temperature was needed. This could be done in an incubator of the chemistry lab. This incubator is the Heraeus B6 Incubator with serial number 3789 FS2L (Heraeus, 2019). This incubator has a control panel where the temperature can be selected and, after some time that the incubator needs to warm up, it achieves the desired temperature. Once it achieves it, it should maintain the temperature very stable.



Figure 2.3: Heraeus Incubator

At the moment to do the experiment, the incubator had some unexpected issues:

- The incubator wasn't very precise in order to adjust the temperature.
- For some temperatures in the range from 20°C to 30°C, the incubator didn't respond well to set the temperature and remain as it was.
- The incubator was overheated by a previous experiment so it was very difficult to lower the temperature, to the temperatures expected for our purpose.

As far as the thesis writer and supervisor concerns, this problems of the oven could not be solved and this will influence the way to do the experiment.

Chapter 3

Methodology / project development

3.1. Structuring the data and plotting it

This thesis uses Pandas library from Python in order to structure the data. This part is divided in two main functions: the one that has to create new Excels as explained in the introduction and the other that needs to plot the data from the new Excel.

3.1.1. Structuring the data

The main idea is, by selecting the time range in the pdf_info, first select the start time of the first cell of the entire group of measurements and after select the end time of the last cell of the same group of measurements. Then looking in the other excel, the generalPDF to measurements with the same times, the start and the end, so it can be chosen as the range of measurements between these times that the data from the database has. After it creates the Excel, named plotPDF because the data in the excel will be plotted in further stages, in the local folder specified in the function.

Now, let's focus on the main functions available in Pandas library that have been used in this script ³:

- `pd.read_csv`, `pd.read_excel`: This function opens the Excel file (.csv or .xlsx) placed on an specific path and converts its data into a dataframe.
- `df.loc`: Returns the cell, column or row specified as parameter.
- `df.size`: This function returns the number of elements in the dataframe ⁴

³Df mean dataframe, used in Pandas library for managing the data, and pd is referred to Pandas library.

⁴In this thesis case, as the dataframe used only has one column, returns the number of the rows.

- `df.reset_index`: Resets the index of the dataframe⁵.
- `df.index.values`: This function returns the index value of the condition applied (in this script, it returns the cells that have the same time than the start time and the end time of the group of measurements)
- `df.to_csv`: Converts the dataframe into a Excel file, placing it in the specific path that is selected.

Following the requirements and the specifications that this thesis had, the code can be consulted in Annex 1.

Once the thesis started working in the comparison with Tena Identifi, this script was updated. Measurements of our module are time shifted, because the time set in the microcontroller, so in summertime is shifted 2h and in wintertime 1h. For this reason, this time has to be added to all the data of the database in order to take the desired data in the range of Tena Identifi data.

So in the script, once imported the data from `Data_DB` file and before selecting the range of cells to export as a classified data, this time is added. The data needs to be converted in order to make available the time addition, so the steps of this update are: convert the data into usable data, add the time shift necessary and again insert the new timestamps into the dataframe that contains all the data.

```

In [ ]: generalPDF = pd.read_csv(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20181008 - Data_DB.
<
In [ ]: rlist = generalPDF["timestamp"].to_csv(None, header=False,
index=False).split('\n')
rlist = rlist[:-1]

In [ ]: clist = list()
i=1
for index, item in enumerate(rlist):
    clist.append(datetime.strptime(rlist[index], '%Y-%m-%d %H:%M:%S'))

In [ ]: nlist = list()
for item in clist:
    nlist.append(pd.to_datetime(item.date()).strftime('%Y-%m-%d') + ' ' + item.time().strftime('%H:%M:%S'))
<
In [ ]: nnlist = list()
for item in nlist:
    nnlist.append(item+timedelta(hours=2))

In [ ]: generalPDF["timestamp"] = pd.DataFrame({'timestamp':nnlist})

In [ ]: cellToStart = generalPDF.index.values[generalPDF['timestamp'] == dateStart]
cellToStart = int(cellToStart)
cellToStart

```

Figure 3.1: Update to the structuring script with the time addition

3.1.2. Plotting the data

This script select `plotPDF` file and opens it and classifies the measurements according to the module that came from. After it applies the conversion rates to

⁵Each row can be identified with an index in order to make it easier to select, very useful in case of creating shorter dataframes from the `generalPDF` dataframe, so they can be reindexed to easily manage them.

the data classified and plot it.

To select the data, the scrip search for the title (or remark in that case) of the new Excel created by selecting the cell of the pdf_info that contains that remark, referred to a specific group of measurements.

After, it imports the file that applies the conversion rates for the sensor values and imports as well the file that separates the data by the identification sequence of each sensor module. Now it will plot all the figures for each device id ⁶, and save them in a local folder.

Plotting all graphs together is a bit confusing, and all the plots are so small that the data is not well recognized. So this script also provides a function that only plots one of the nine graphics in order to have a bigger visualization and study better what it contains.

Following the requirements and the specifications that this thesis had the code of this script is in Annex 4 ⁷.

Conversion rates

Due to the experiment results, that will be further explained in chapter 4, this code couldn't be implemented. As they are conversion rates, changes in measurements that mean change in gas concentrations are still able to see. That's why this thesis can continue with the comparison without having a good characterization of the sensors.

Id identification

As explained before, this function separates all the data from each sensor module (there are 7 sensor module available) and also changes their device id into their names that makes them more recognizable. The code can be consulted in Annex 5.

Plot all the sensor module data

This script needs to define the nine plots and place it in the layout. So it is divided into two parts, the trace creation, that defines how will each plot look, and the layout creation, that defines where these graphs will be placed.

For plotting all the sensors outputs, scatter plots suits best with the objective of observing the evolution of the concentration, temperature or relative humidity levels through the time. So first, the nine traces were created, defining the axis values, as scatter plots, graphs titles and other aspects to fulfill the information of the plots.

After, the layout is chosen where each plot must be placed, treating the space of the layout as a matrix from [0,0] to [1,1]. By the organization of the space defined in the code, all 9 plots can be placed. The problem is that this defined space is

⁶Taking in account than for some groups of measurements, an specific sensor module may not have measured any data due to the fact that is switched off, not working ...

⁷Also this script imports and uses another scripts created in order to make this Python module more modular. The following explanations will be about them.

fixed, so as more graphs need to be plotted, the smaller are the graphs and it increases the difficulty to recognize the plots performances.

Finally with all the traces and the layout is created the figure that it will be plotted in the main script. This code is in Annex 6

Focused plot

This function returns a plot of the sensor values selected by filling in a number between 0 and 8. It saves all the sensor names and values in some lists that have 9 possible options for creating the plot. In this case there is not necessity to distribute the plots because there is only one.

Also it returns the name of the sensor selected in order to show it in the graph. The code is in Annex 7.

3.2. Conversion rates and external factors compensation

By using the datasheets (Pewatron, 2019) and annotations, (SGX-Sensortech, 2007a), (SGX-Sensortech, 2007b) and (SGX-Sensortech, 2015), of the sensors and microcontroller it should be possible to, by the formulas written in the datasheets, find out the conversion rates from the digital values obtained from the microcontroller into the real gas concentrations. As the sensor modules are already built, some information is incomplete so unfortunately this way can solve the objective of this section.

So, for this reason, an experiment has been created in order to do the conversion rates and compensate the external effects in the sensors, that are not provided in the datasheets. As the temperature is the main external factor that modifies the performance of the sensors it therefor is the compensation this thesis will focus on. A part of this experiment is based on the how external factors affect the sensitivity of the sensors. In a scientific article Wei u. a. (2019) it is considered the following relation, considering relative humidity and temperature as external facts:

$$S_{RH,T} = a_1xRH + a_2xT + a_3 \quad (3.1)$$

But also the baseline, the offset value of the sensor when is measuring 0ppm:

$$B_{RH,T} = b_1xRH + b_2xT + b_3 \quad (3.2)$$

So this experiment wil try to find this relation (only with temperature), to apply it.

3.2.1. The experiment

This experiment consists in measuring the output of the sensors in an incubator, each time with a known gas concentration and a range of temperatures from 20°C

to 30°C⁸. So for a certain amount of gas concentration it should be able to observe the performance and the temperature influence.

If the gas concentration starts from 0 ppm, so the gas is not present, it will provide information about the effect of the temperature when the sensor is measuring 0 ppm, so the effect of the zero bias will be able to get compensated. After, as the temperature affects the sensitivity of the sensor, apart from biasing the zero point, it needs to be studied how changes the dependence between the effect of the temperature in the performance and the gas concentration inside the incubator. Knowing the range of gas concentration our sensors are available to measure, this are the following measurements⁹ that will be tested¹⁰:

- For the sensor SGX-4NH3: from 0 to 100 ppm by an increase of 25 ppm each measurement.
- For the sensor MiCS-6814: from 0 to 200 ppm by an increase of 50 ppm each measurement.
- For the sensor MQS-137: from 5 to 200 ppm by an increase of 50 ppm each measurement.

So at the end of the experiment a variety of curves will be plotted by a Python script, showing the relation from the gas concentration, the input of the sensors, and the digital values, the output of the sensors, in the range from 20° to 30°. After, by programming, it should be detected which range of ppm is each measurement¹¹ and apply the inverse of the curve that fits for that amount of ppm, and the temperature that is measuring the sensor module itself.

To observe the data coming from the sensor module placed in the incubator, a program was created that sends all the data from the module by the microcontroller and saves it in a database, also offering to plot the data at a real time. For tracking the time when the measurements were taken and relate it with the temperature in the incubator, the time when the incubator reaches a certain temperature is annotated manually by the researcher for each step..

The values observed in the graph are the output digital values from the microcontroller, remember from 0 to 4095, that represents the converted values, by the sensor stage and the microcontroller, from the gas concentration, temperature, relative humidity... gathered by the sensors. Afterwards, the results obtained can properly separate the sensors that seems to work well, from the sensors that

⁸That is the expected temperature of the nursing home and also what the incubator allow us to work with

⁹Notice that the hydrogen sulfide sensors will no be tested. Further explanations in the following sections will explain why.

¹⁰MiCS and MQS sensor should be tested until 500 ppm, but the laboratory available hours only let to do it until 200. Still the results of this experiment will be very significant.

¹¹As the data has values of the input/output from the sensors, knowing the output means knowing moreless the input so it can be applied the right curve. Also the values of a new measurement can be fitted with a linear approximation between the two nearest curves if it is in the middle between two ranges of ppm

don't¹².

New experiment	
2018-12-11 15:56:00	Temperature set to 30°C
2018-12-11 15:56:00	Open de oven door to start the experiment going from 30 to 20°C (we set the temperature of the oven to 18°C)
2018-12-11 15:58:00	Temperature set to 29°C
2018-12-11 16:00:00	Temperature set to 28°C
2018-12-11 16:04:00	Temperature set to 27°C
2018-12-11 16:19:00	Temperature set to 26°C
2018-12-11 16:25:00	Temperature set to 25°C
2018-12-11 16:29:00	Temperature set to 24°C
2018-12-11 16:36:00	Temperature set to 23°C
2018-12-11 16:43:00	Temperature set to 22°C
2018-12-11 16:57:00	Temperature set to 21°C
2018-12-11 17:22:00	Temperature set to 20°C

Figure 3.2: Experiment Notes

So after a test measurement, that is the real time plot obtained:

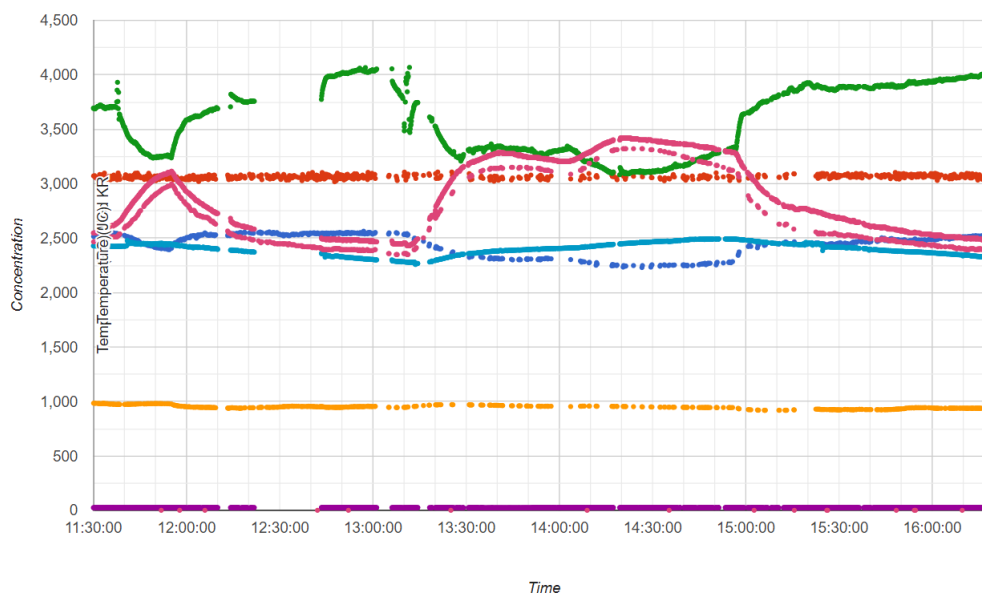


Figure 3.3: Example for testing the experiment

So, for this experiment there is the necessity of some chemical components as well as tools to execute it:

- A wifi connection to gather data from the sensor modules.
- Ammonia solution.
- Some beakers (at least 2).
- A test tube and a funnel.
- An accurate syringe
- The incubator.

¹²For this example, the data plotted was from two different modules, the ammonia sensors and the temperature from each one. As it can be observed, the purple plot of data may reveal that the sensor is not working well or directly is not working.

- Power supply for the sensor module.

During the experiment, the researcher noticed that the temperature setting of the incubator was not stable.

For this reason, the experiment could not consist on selecting the temperatures desired, grade per grade, and obtain accurated response of the effect of the temperature in the sensors performance.

Instead the test was done going from 30°C to 20°C, setting the incubator to 18°C and opening the door for lower the temperature faster, because if not it lasts a lot. Also opening the door of the incubator means that the gas will spread into the aire of the entire lab, so, for avoiding that effect, another module will be closed in a box inside the incubator with a gas solution. Also this will offer to the thesis a first impressions about the performance of the sensors once the gas is very concentrated in a small place or spreads into the air of a big room, the case that happens in a nursing home.



Figure 3.4: Placement of both modules with ammonia solutions inside the incubator

3.2.2. Decreasing the concentration of ammonia in a solution

Some gas concentrations had to be applied inside the incubator for testing with the sensors. This thesis has been provided by KU Leuven with a solution of 250000ppm of ammonia. This thesis was not provided with any hydrogen sulfide

solution, so that why the only tested gas was ammonia.

For obtaining the desired gas concentrations, a dilution has to be applied. A dilution consist on decreasing the concentration of a solute in a solution by adding more water to that solution. The equation that relates the initial and final gas concentrations (c) and volumes (V) is:

$$c_{initial} \times V_{initial} = c_{final} \times V_{final} \quad (3.3)$$

So for example, if 25 ppm of ammonia are needed, the volumn of the solution is, for example with 1 microliter of the solution of ammonia and the rest for water, has to be 10 milliliters:

$$c_{initial} \times V_{initial} = V_{final} = \frac{c_{initial} \times V_{initial}}{c_{final}} = \frac{1ul * 250000ppm}{25ppm} = 10ml \quad (3.4)$$

So for mixing it, it is used a test tube and a funnel. The water and the ancient solution of ammonia are saved in some beakers. For putting low volumes of the solution of ammonia, it is needed a very accurate syringe.



Figure 3.5: Equipment for the dilution process

3.3. Market application comparison

The final step of this thesis consist off comparing the sensor moduke with the similar application that is in the market named Tena Identifi, described in the

introduction of this thesis.

The results of the Tena Identifi measurements is handed by the nursing home on paper and anonymized as shown in the Chapter 1. So the first step is to digitalize the information given in these documents, by introducing all their values into an Excel ¹³.

The values of that columns, ordered from left to right, are: "Date", "Level", "Change logger", "Leakage", "Outlasting in het product", "Toilet niet succesvol" and "Toilet bezoek succesvol" ¹⁴.

2018-07-30 00:00:00	0	0	0	0	0	0
2018-07-30 01:00:00	0	0	0	0	0	0
2018-07-30 02:00:00	0	0	0	0	0	0
2018-07-30 03:00:00	0	0	0	0	0	0
2018-07-30 04:00:00	0	0	0	0	0	0
2018-07-30 05:00:00	0	0	0	0	0	0
2018-07-30 06:00:00	0	0	0	0	0	0
2018-07-30 07:00:00	0	0	0	0	0	0
2018-07-30 08:00:00	0	0	0	0	0	0
2018-07-30 09:00:00	0	0	0	0	0	0
2018-07-30 10:00:00	0	0	0	0	0	0
2018-07-30 11:00:00	0	0	0	0	0	0
2018-07-30 12:00:00	1	0	0	0	0	0
2018-07-30 13:00:00	3	0	0	0	0	1
2018-07-30 14:00:00	4	1	0	0	0	0
2018-07-30 15:00:00	0	0	0	0	0	0
2018-07-30 16:00:00	0	0	0	0	0	0
2018-07-30 17:00:00	0	1	0	0	0	0
2018-07-30 18:00:00	0	0	0	0	0	1
2018-07-30 19:00:00	0	0	0	0	0	0
2018-07-30 20:00:00	0	0	0	0	0	0
2018-07-30 21:00:00	2	0	0	0	0	0
2018-07-30 22:00:00	2	0	0	0	0	0
2018-07-30 23:00:00	2	0	0	0	0	0
2018-07-31 00:00:00	4	0	0	0	0	0
2018-07-31 01:00:00	4	0	0	0	0	0
2018-07-31 02:00:00	5	0	0	0	0	0
2018-07-31 03:00:00	6	0	0	0	0	0
2018-07-31 04:00:00	6	0	0	0	0	0
2018-07-31 05:00:00	6	0	0	0	0	0
2018-07-31 06:00:00	7	0	1	0	0	1
2018-07-31 07:00:00	7	0	0	0	0	0
2018-07-31 08:00:00	0	1	0	0	0	0
2018-07-31 09:00:00	0	0	0	0	0	0

Figure 3.6: Excel with the Tena Identifi relevant values

For making able to manage by the user this data, a code has been created and can be consulted in Annex 11.

So once it's digitalized, a comparison of the measurements of Tena Identifi and the measurement that the thesis sensor module has taken at the same time has to be done. So the first step will be a visual comparison to try to find any similar patterns.

After, the data will be compared statistical to find any significant resultsn.

Plotting the digital values of Tena identifi

To plot all the values of the Tena Identify, it needs to be plotted not only the values of the measurements of the incontinence, but also the other values that the caregivers annotated during the measurement, described in the section before. The code to perform that is placed in the annex 12.

¹³Where the titles of each column of values isn't written because of a problem with the Python script. Them will be put in the Python script instead of putting them in the Excel file

¹⁴Traduction to english: "Date", "Level", "Change logger", "Leakage", "Outlasting in the product", "Toilet visit not successful" and "Toilet visit successful"

It takes the Excel where all the Tena values are digitalized. It creates two plots: one to track the voiding's of the user, and one to control the other important aspects that involves this Tena Identifi product, bot through the same 72h.

With this, this thesis pretends to digitalize the documents shown in chapter 1 for using them as data and compare them with this project.

3.3.1. Visual comparison

The visual comparison consist on comparing the performance of the Tena Identifi measurements and the sensor module measurements by graphs. This graphs will give us information about similarities of differences between both procedures.

3.3.2. Mathematical comparison

This mathematical comparison consist on take all the digitalized Tena Identifi data, and group it by level, taking all the same data with the same level in one group.

In this groups will selected the intervals of time that are related with all the measurements. This intervals will be used to take all the data from the database, convert it to the same data type than the Tena Identifi data, and relate the levels of Tena Identifi measurements and the measurements done by the sensor module. Finllay, a graph showing all the groups of data will not only prove the performance of the first visual impressions, but also will give new information about the performance of the sensor module.

Following this explanations, this code can be found here: 13. Some scripts were created in order to help to achieve the desired performance of this mathematical comparison, that will be explained in the following sections.

Separate dates

This script is responsible of obtaining the data ranges from the measurements of Tena Identifi that previously have been grouped by levels of incontinence.

For this, this script takes in account the position of the data, by looking at the index value of each measurement, so it can prove if they are connected by the indexes, so they are connected in time, or not.

For each one that is not connected, it adds to the list of start dates a new stat date as the actual date in the loop and also adds an end date into the end dates list of the previous list. For this procedure, is needed to add as a start date the time of the first measurement and also the time of the last measurement as an end date.

This code can be consulted in Annex 14.

Find nearest data

This script finds the nearest measurement to the dates intervals provided as a parameter in the function. It needs to find the time of the measurement that is the nearest above the start dates and the time of the measurement that is the

nearest below the end dates.

For doing it, it calculates the time difference and obtains the minimum difference, above or below, of the desired dates.

Also sometimes one Tena Identifi measurement level will be unconnected in time with same measurement levels. For this case, the program is responsible to take the two measurement that cover this date, the next and the previous ones.

This code is provided in Annex 15.

Select Data

This scrip uses an empty dataframe and adds the data that is in the range of cells required, obtained as a parameter of the function. This code can be found in Annex 16.

Print the comparison

This scrip creates a trace of data to plot pear each available group of data related with the level of Tena Identifi measurements. It plots in the x axis the level of the group of measurements and in the y axis the value of the desired sensor of the group of measurements. It returns the list of all the traces to plot them in the main script.

Previously to this script, in the main script, has to be added to the data the level related with the data group to use it as x axis.

This code can be consulted in Annex 17.

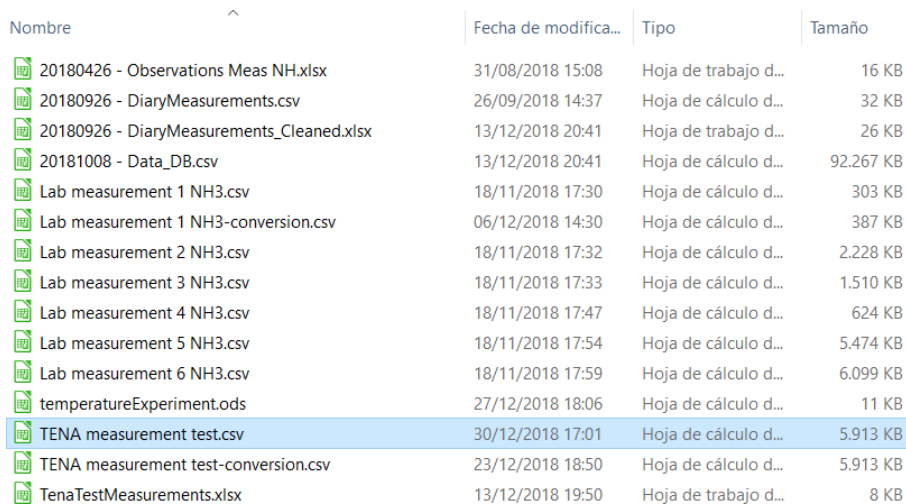
Chapter 4

Results

4.1. Structuring the data and plotting it

4.1.1. Structuring the data

Once the code of this section is executed, it is created a new Excel file, with the name corresponding to the remark.



Nombre	Fecha de modifica...	Tipo	Tamaño
20180426 - Observations Meas NH.xlsx	31/08/2018 15:08	Hoja de trabajo d...	16 KB
20180926 - DiaryMeasurements.csv	26/09/2018 14:37	Hoja de cálculo d...	32 KB
20180926 - DiaryMeasurements_Cleaned.xlsx	13/12/2018 20:41	Hoja de trabajo d...	26 KB
20181008 - Data_DB.csv	13/12/2018 20:41	Hoja de cálculo d...	92.267 KB
Lab measurement 1 NH3.csv	18/11/2018 17:30	Hoja de cálculo d...	303 KB
Lab measurement 1 NH3-conversion.csv	06/12/2018 14:30	Hoja de cálculo d...	387 KB
Lab measurement 2 NH3.csv	18/11/2018 17:32	Hoja de cálculo d...	2.228 KB
Lab measurement 3 NH3.csv	18/11/2018 17:33	Hoja de cálculo d...	1.510 KB
Lab measurement 4 NH3.csv	18/11/2018 17:47	Hoja de cálculo d...	624 KB
Lab measurement 5 NH3.csv	18/11/2018 17:54	Hoja de cálculo d...	5.474 KB
Lab measurement 6 NH3.csv	18/11/2018 17:59	Hoja de cálculo d...	6.099 KB
temperatureExperiment.ods	27/12/2018 18:06	Hoja de cálculo d...	11 KB
TENA measurement test.csv	30/12/2018 17:01	Hoja de cálculo d...	5.913 KB
TENA measurement test-conversion.csv	23/12/2018 18:50	Hoja de cálculo d...	5.913 KB
TenaTestMeasurements.xlsx	13/12/2018 19:50	Hoja de trabajo d...	8 KB

Figure 4.1: Local folder where all the measurements files are placed

If the file is opened, it can be observed that it includes all the measurements in the range of time selected by the group of measurements selected:

id	timestamp	resident_id	device_id	nr_of_measurements	MICSNH3	MICSNO2	MICSCO	SGXNH3	SGXH2S	SPH2SH2S	SPH2STEMP	HIHTEMP	HIHHUM	DHITEMP	DHTHUM
179137	2018-07-30 00:00:10		32002f000b51353432383931	3	4003			1142	5	2117	2490	0	0.33.4	26.1	
179138	2018-07-30 00:00:16.1.0		360059000f51353532343635	7	3770			28	2294	2043	2390	0	0.31.5	27.1	
179139	2018-07-30 00:01:55		32002f000b51353432383931	10	3999			1144	4	2135	2488	0	0.33.5	26.0	
179140	2018-07-30 00:02:00.1.0		360059000f51353532343635	10	3770			27	2765	2053	2389	0	0.31.5	27.2	
179141	2018-07-30 00:02:48		32002f000b51353432383931	5	4015			1143	5	2107	2488	0	0.33.5	25.9	
179142	2018-07-30 00:02:58.1.0		360059000f51353532343635	2	3737			26	2193	2042	2387	0	0.31.5	0.0	
179143	2018-07-30 00:03:40.1.0		360059000f51353532343635	4	3732			26	3549	2045	2387	0	0.31.5	27.2	
179144	2018-07-30 00:03:51.1.0		360059000f51353532343635	1	3731			27	2352	2057	2388	0	0.31.5	27.2	
179145	2018-07-30 00:04:32		32002f000b51353432383931	10	4000			1142	4	2143	2486	0	0.33.6	25.8	
179146	2018-07-30 00:04:49.1.0		360059000f51353532343635	5	3747			27	2898	2052	2388	0	0.31.5	27.2	
179147	2018-07-30 00:06:06		32002f000b51353432383931	9	3981			1143	4	2133	2487	0	0.33.6	25.8	
179148	2018-07-30 00:06:33.1.0		360059000f51353532343635	10	3748			27	2826	2044	2389	0	0.31.5	27.2	
179149	2018-07-30 00:07:41.1.0		360059000f51353532343635	6	3767			28	2470	2054	2389	0	0.31.5	0.0	
179150	2018-07-30 00:07:41		32002f000b51353432383931	9	3999			1143	5	2029	2486	0	0.33.6	25.8	
179151	2018-07-30 00:09:05.1.0		360059000f51353532343635	8	3751			26	2454	2046	2389	0	0.31.5	27.2	
179152	2018-07-30 00:09:05		32002f000b51353432383931	8	4016			1142	5	2080	2486	0	0.33.6	25.7	
179153	2018-07-30 00:10:51		32002f000b51353432383931	10	4012			1144	4	2017	2485	0	0.33.7	25.7	

Figure 4.2: First sensor values of the new Excel

200960	2018-08-02 23:55:29		32002f000b51353432383931	3	3862			1142	5	2026	2525	0	0.34.7	24.5	
200961	2018-08-02 23:55:35		3.40E+23	1	3859			3040	72	1992	1904	0	0.36.9	0.0	
200962	2018-08-02 23:56:27		3.40E+23	3	3845			3044	73	1994	1906	0	0.36.9	19.3	
200963	2018-08-02 23:56:38.1.0		360059000f51353532343635	10	3070			26	2389	2051	2449	0	0.32.7	0.0	
200964	2018-08-02 23:57:09		3.40E+23	4	3848			3027	73	1994	1889	0	0.36.9	19.3	
200965	2018-08-02 23:57:24		32002f000b51353432383931	10	3945			1143	4	2032	2524	0	0.34.7	24.5	
200966	2018-08-02 23:57:31		3.40E+23	2	3854			2996	72	1994	1913	0	0.36.9	19.3	
200967	2018-08-02 23:57:52		3.40E+23	2	3855			3015	73	1994	1855	0	0.36.9	19.3	
200968	2018-08-02 23:58:35		3.40E+23	4	3858			3038	72	1994	1917	0	0.36.9	19.3	
200969	2018-08-02 23:58:56		3.40E+23	2	3861			3062	73	1994	1919	0	0.36.9	19.3	
200970	2018-08-02 23:59:06		3.40E+23	1	3847			3027	72	1993	1855	0	0.36.9	19.3	
200971	2018-08-02 23:59:08		32002f000b51353432383931	10	3947			1143	5	2008	2524	0	0.34.7	24.5	
200972	2018-08-02 23:58:42.1.0		360059000f51353532343635	10	3073			27	2837	2054	2440	0	0.32.8	25.6	
200973	2018-08-02 23:59:17		3.40E+23	1	3849			3058	73	1995	1906	0	0.36.9	19.3	
200974	2018-08-02 23:59:38		3.40E+23	1	3858			3021	71	1993	1906	0	0.36.9	19.3	

Figure 4.3: Last sensor values of the new Excel

After the update, this test proves the time added to the data:

```

In [3]: generalPDF = pd.read_csv(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20181008 - Data_DB.csv')
generalPDF["timestamp"].iloc[0:5]

Out[3]: 0    2018-03-04 15:38:48
1    2018-03-04 15:38:50
2    2018-03-04 15:38:52
3    2018-03-04 15:38:54
4    2018-03-04 15:41:03
Name: timestamp, dtype: object

In [4]: rlist = generalPDF["timestamp"].to_csv(None, header=False,
index=False).split('\n')
rlist = rlist[:-1]

In [5]: clist = list()
i=1
for index, item in enumerate(rlist):
    clist.append(datetime.strptime(rlist[index], '%Y-%m-%d %H:%M:%S'))

In [6]: nlist = list()
for item in clist:
    nlist.append(pd.to_datetime(item.date()).strftime('%Y-%m-%d') + ' ' + item.time().strftime('%H:%M:%S'))

In [7]: nnlist = list()
for item in nlist:
    nnlist.append(item+timedelta(hours=2))

In [9]: generalPDF["timestamp"] = pd.DataFrame({'timestamp':nnlist})
generalPDF["timestamp"].iloc[0:5]

Out[9]: 0    2018-03-04 17:38:48
1    2018-03-04 17:38:50
2    2018-03-04 17:38:52
3    2018-03-04 17:38:54
4    2018-03-04 17:41:03
Name: timestamp, dtype: datetime64[ns]

```

Figure 4.4: Execution of the update

In this functional example it is shown the first five timestamps of the entire data. It can be observed that 2 hours were added. The internal procedure of that script can be consulted in Annex 3, and a flow chart that summarizes it in Annex 2.

4.1.2. Plotting the data

Once the code of this section is executed, new graphs are saved in the local folder:

TENA measurement test SPH2SH2S sensor plot.png	19/12/2018 12:39	Archivo PNG	54 KB
TENA measurement test SPH2STEMP sensor plot.png	19/12/2018 13:07	Archivo PNG	47 KB
TENA measurement test(John_Connor) general plot.png	30/12/2018 17:44	Archivo PNG	44 KB
TENA measurement test(Kate_Brewster) general plot.png	30/12/2018 17:44	Archivo PNG	44 KB
TENA measurement test(Kyle_Reese) general plot.png	30/12/2018 17:44	Archivo PNG	44 KB
Tena plot.png	13/12/2018 20:09	Archivo PNG	25 KB

Figure 4.5: Local folder where all the plot files are placed

And this is an example of the result of this function:

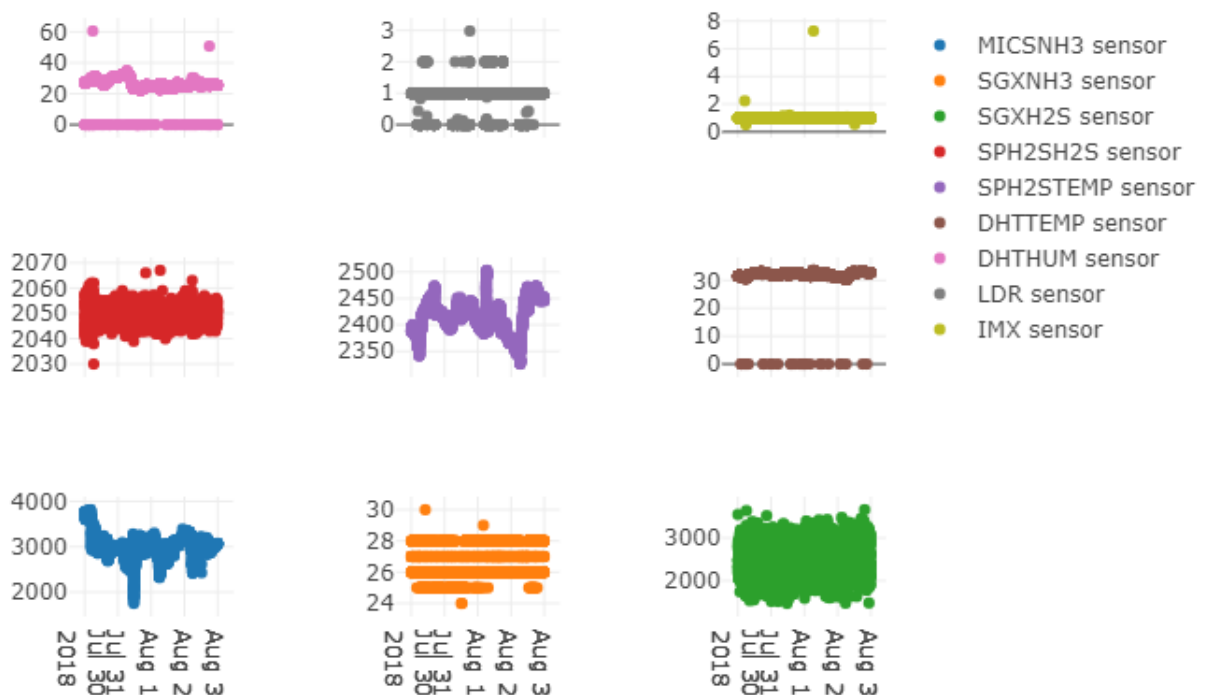


Figure 4.6: An example of the plotting results

This plots provide the user the visual information of the variation of the gas concentration and all the other parameters. A study of them will allow to find incontinence patterns.

Also it plots the selected sensor, for example the SGXNH3:

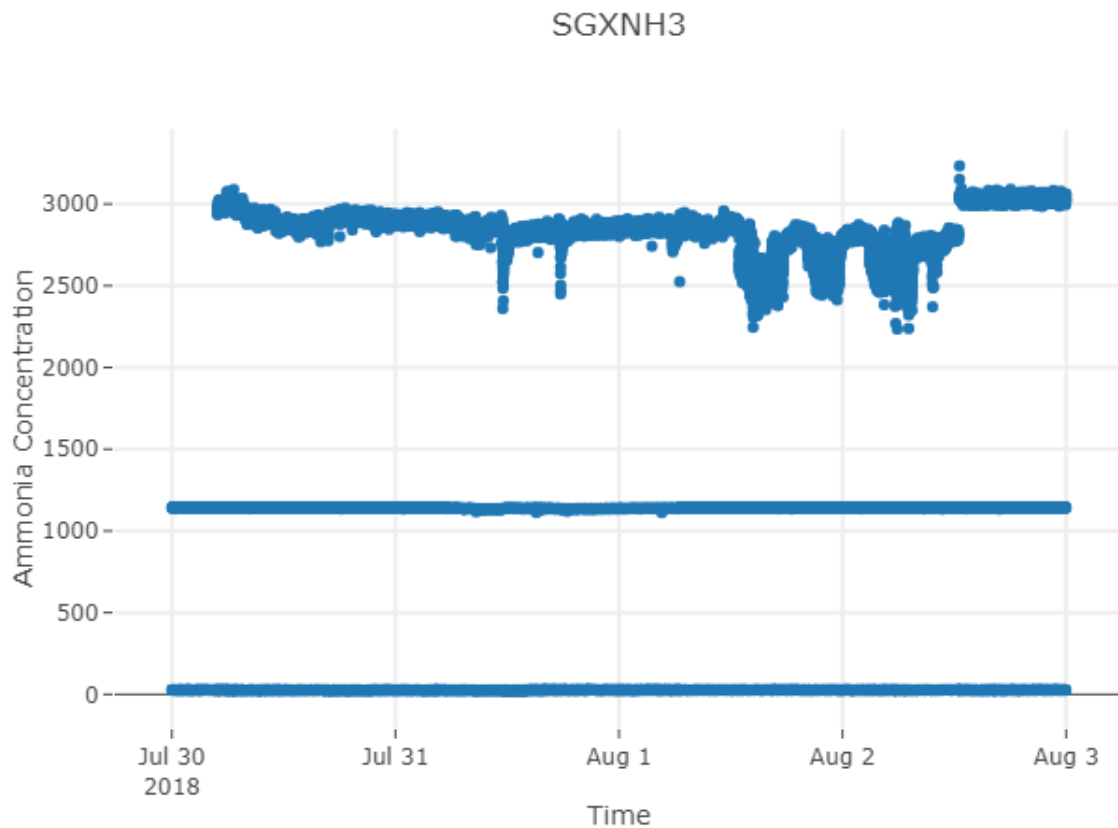


Figure 4.7: An example of the focused plot

4.2. Conversion rates

4.2.1. Experiment results

First of all, as the results that will be observed are not the expected, a direct comparison of the temperature effects on the sensor and the exact temperature of each measurement is not done, as is explained in chapter 3 with the equation 3.1 and 3.2. This will be further explained in the conclusion, in chapter 5.

First, the experiment measured the outputs of the microcontroller of both modules with 0 and 25 ppm of ammonia dilution placed in the incubator:

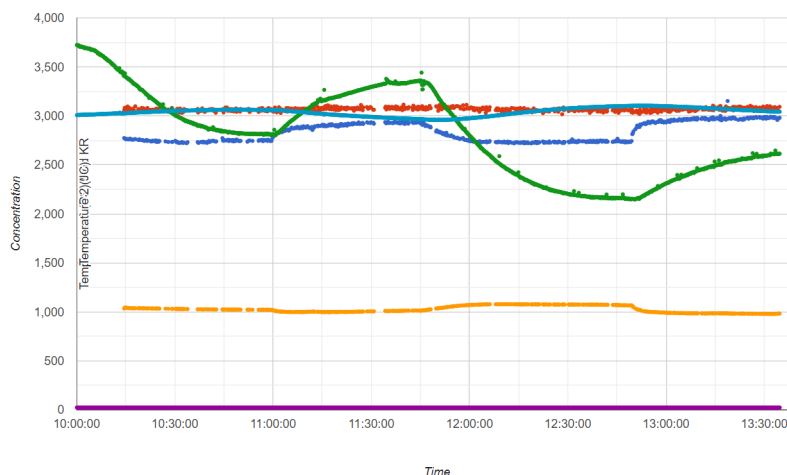


Figure 4.8: Experiment result with 0ppm and 25 ppm

In that result, it can be observed that the ammonia MiCS sensor (green data), the ammonia sensor SGX-4NH3 (purple data), that for the results might not work well or not work directly and the ammonia sensor MQ137 (light blue data) of the module placed inside the box, and the MiCS sensor (dark blue data), the SGX-4NH3 (red data) and the MQ137 (yellow data), of the module placed in the incubator but outside the box.

The MiCS and MQ137 ammonia sensors lower the output level as the gas concentration increases, and the SGX-4NH3 increases the output level as the gas concentration increases, so for this explanation can be detected the two patterns in that results. First of all, the sensors needed to stabilize to 0 ppm so they lowered the level until it was stable (So they don't use all the available range, because if not they will have 0 ppm at the highest digital value, 4095). After opening the door of the incubator and setting it to 18°C (at 11:00 am and 12:52 am), the sensor levels start to increase, as can be observed from 11:00 am to 12:50 am and from 12:52 am to 13:40 pm, so that means that the gas concentration is directly proportional with the temperature.

After the first test, once reached in the incubator 20°C, it is warmed until 30°C meanwhile the 25 ppm ammonia concentration was placed. After the stabilization, the same pattern is obtained. The SGX sensor of the module placed outside the box did ¹⁵ not suffer a big change in measurements with the change of the temperature as it will maintain the level in further test. This means that this sensor could not be working well or not be appropriate for this measurements.

Also notice that the both MQS sensor reacts to the change of ammonia with a very small change in his output. Another thing that might tell us a difference is the offset of both sensors. Placing this sensor very close to the solution and closed with the box, the sensor detects much less gas concentration (offset around 3000) than the sensor placed outside the box (approximately 1000). This is a very important effect that might show the dependence of the environment in this sensor. Further tests need to be done to prove that first impressions.

Now is important to take conclusions about the delay of the sensor measurements

¹⁵Remember that the SGX of the module placed in the box is not reacting as expected

In MiCS sensor placed output the box the delay is less than 30 min but in the MiCS sensor in the box is about 1 hour and 5 minutes. This is related with the spreading or not of the gas, that is related with the concentration that affects the time delay for sensor stabilizing. Another effect of the ammonia spread in the air of the room is that the levels of the MiCS placed outside the box, once stabilized, are very similar for any gas ammonia concentration.

This thesis is interested in the sensor outside the box because is similar to the environment of a room in a nursing home. But still 30 min to measure the ammonia concentration means that the resident has to wear the disposable once happened the voiding or let the bed humid. For this reason, this sensors seem not to be the adequate for that project because it is needed a faster response.

But the sensor starts changing its measurements once applied the gas, so this MiCS sensor actually is good for detecting if the resident had a voiding or not but not for relating the measurements with the quantity of voiding.

Let's see another test of the experiment:

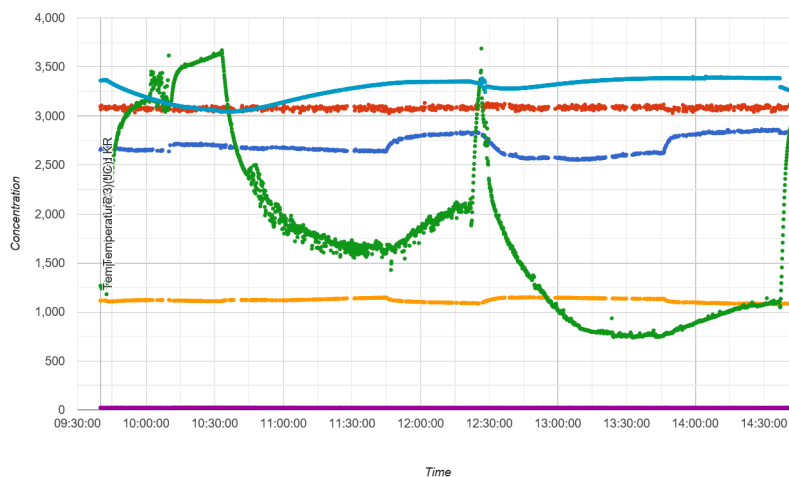


Figure 4.9: Experiment result with 50ppm and 75ppm ammonia dilutions

Also in this test can be observed a similar performance of the MiCS sensor ¹⁶. But the effect of the temperature in the MiCS sensor changes: the effect increases in the sensor placed outside the box and decreases in the other as the concentration applied into the incubator increases.

Also the changes of the ammonia measurements of the MiCS sensor placed outside the box for each gas concentration are not significant, so it proves the conclusions of the first test. The MiCS placed outside the box still stabilizes at similar levels from the last test.

For the MQS sensor can be observed a difference between both sensors. The sensor, placed outside the box, values continues to have small changes when increasing the gas concentration applied, but the MQS sensor has a bigger change. This could be explained because the ammonia dilution placed outside the box

¹⁶Suddenly the MiCS sensor placed inside the box decrease the measurement of ammonia after the test of 50 ppm and before starting the test of 75 ppm. This is caused by the opening of the box in order to change the gas concentration of the beaker

might be spreadin into the aire and loosing concentration.

The SGX sensor placed outside the box keeps the same perfomance but with an increase of the offset level, as an increase of the ammonia concentration inside the incubator.

Another test of the experiment will be done with higher ammonia concentrations, but once the ammonia concentration exceeds 100 ppm SGX sensor will not re-
sponse well, because its maximum measurement is 100 ppm.

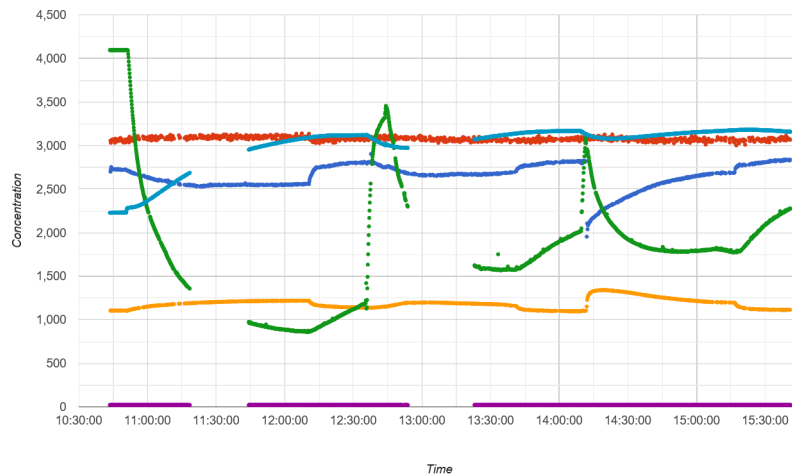


Figure 4.10: Experiment result with 100ppm, 150ppm and 200ppm ammonia dilutions

Here again it the gas sensor continues to have the same performance untill 100ppm ¹⁷. Also the effects of the gas concentration explained in the 50ppm and 75ppm test continues appearing.

But after, in the test for 150ppm and 200ppm, both MiCS starts measuring levels as if they from lower ammonia concentrations. An unexpected performance that needs to be tested in further sections.

Also can be observed that when the ammonia concentration is high, such as 200ppm, the MiCS sensor placed outside the box has a similar performance as the MiCS placed inside the box: between the end of the 150 ppm test and 200ppm test, where the beakers were extracted from the incubator to change the gas concentration it suddenly changes its perception of gas starting from a point that tells that the gas measured is much higher and after it stabilizes at a lower level, the opposite case as the MiCS placed inside the box.

Also it can be observed the strange performance of the MQS placed outside the box similar to the effect in the last test with the MQS sensor placed inside the box, whereas this one has the performance of the other in the last test. This seems to be the opposite performance that should be. So this result is very inconsistent to take some exact information about this sensor.

This test shows a performance undesired for our purposes, that will be reflected in the conclusions. For this reason, any characterization of the sensors could be

¹⁷Some data is missing due to the disconnection of the modules from the internet provided by the mobile, with any reason

programmed and applied.

In terms of temperature effects on sensor module, it can be observed in the curves of all the test taht for the MiCS sensor placed inside the box, the effect of the temperature in the measurements decrease in intermediate gas concentrations and increase when measuring the highest or the lowest ammonia concentrations.

4.3. Market application comparison

So, once executed the code of this section the following outputs are obtained:

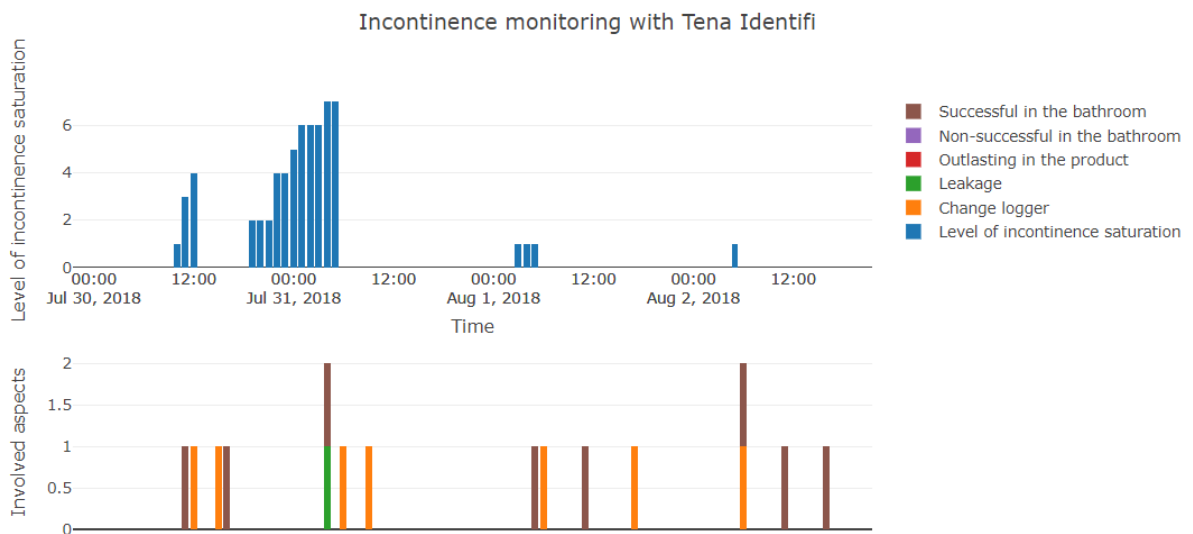
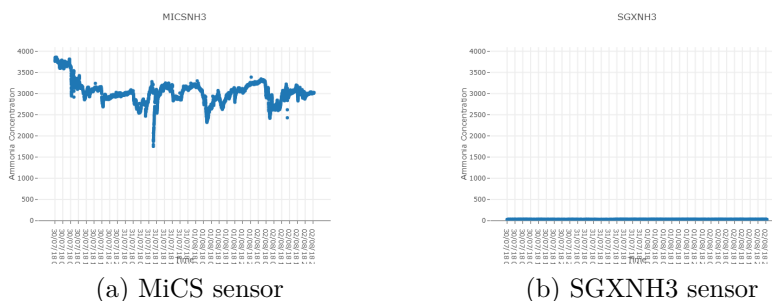


Figure 4.11: Tena Plot with all the values

4.3.1. Visual comparison

So for a first comparison, all the available sensors in the range of time are plotted by the script that plots data already explained:



(a) MiCS sensor

(b) SGXNH3 sensor

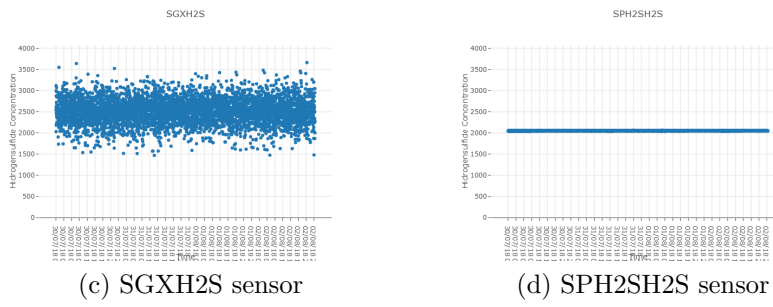


Figure 4.12: John Connor module measurement values

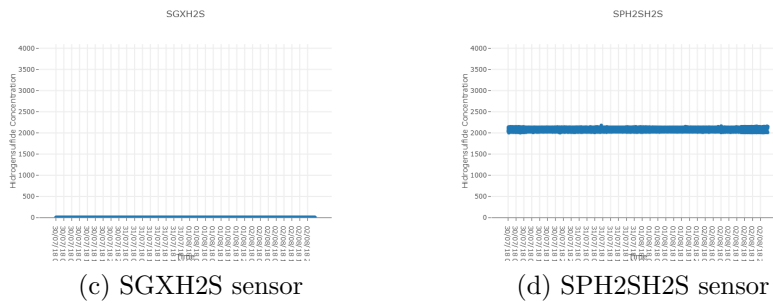
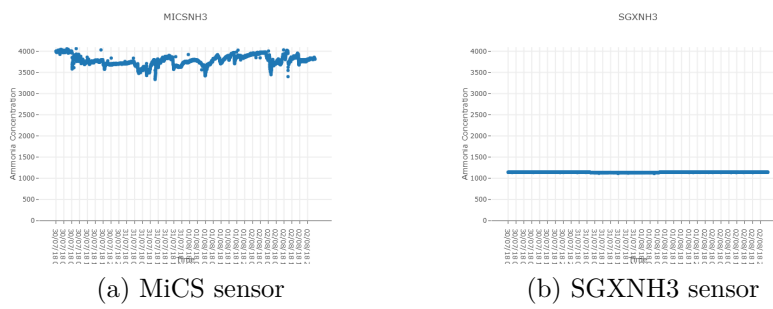
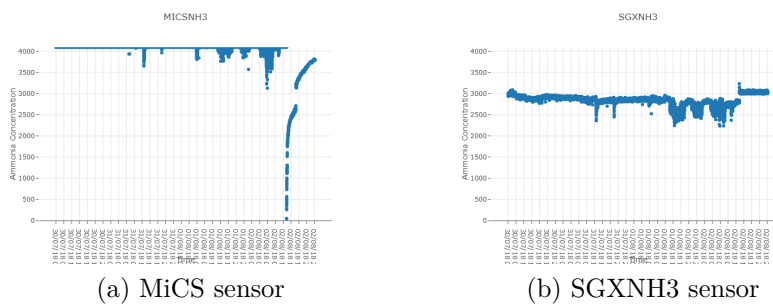


Figure 4.13: Kate Brewster module measurement values



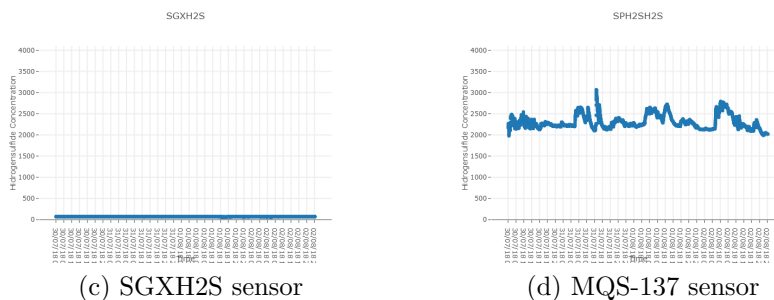


Figure 4.14: Kyle Reese module measurement values

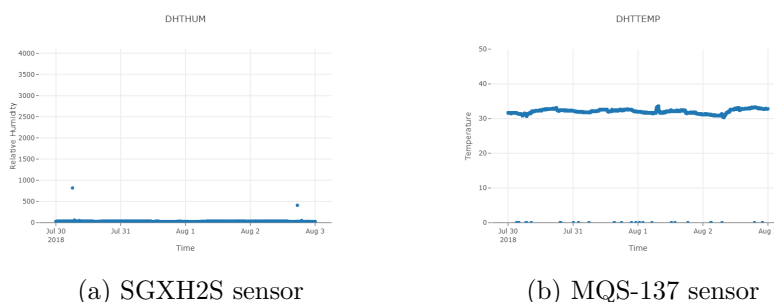


Figure 4.15: Ambient parameters measured

First of all, we observe that some of the sensors are not working well because of the outputs that they have. So for each sensor, the best performance will be chosen to compare. Also the plot of the humidity sensor is a combination of all the modules plots, but none of them seem to work. The relative humidity has an influence in the measurements but in that case any conclusion can be extracted. Also the effects of the temperature couldn't be observed, because it is stable. For the John Connor module, the MiCS sensor seems to have three peaks proportional to the peaks of the Tena Identifi incontinence measurements. The only problem is that the times where the pics happen are not the same. As the experiment shows, the delay time to stabilize the sensors is very long so for that reason might appear as well here this delay between both performances. The other sensors of this module seem not to work well, so for this module cannot be extracted more information.

Kate Brewster sensors seem not to be working as well.

Kyle Reese module adds interesting information to that process. MiCS sensor also has the similar peaks¹⁸ to the John Connor MiCS sensors. Also the SGX ammonia sensor is having similar peaks that confirm that our module and Tena Identifi data have a similar incontinence detection pattern. Als MQS sensor has a similar pattern to the others placing peaks.

The main difference observed is that MiCS and MQS sensor have difference between peaks, so the amount of gas concentration influences the measurements,

¹⁸The final peak might be because the module was reset.

whereas the SGX sensors have all the peaks at the same level, only obtaining if there is gas or not.

By this comparison the study of the performance of the sensors when measuring stool is incomplete. A case with only hydrogen sulfide peaks will give more information about that part of this thesis. But this was the data available for that comparison. With more data this section can be studied deeper and extract more information.

4.3.2. Mathematical comparison

Once executed the code of this section, this is the result obtained:

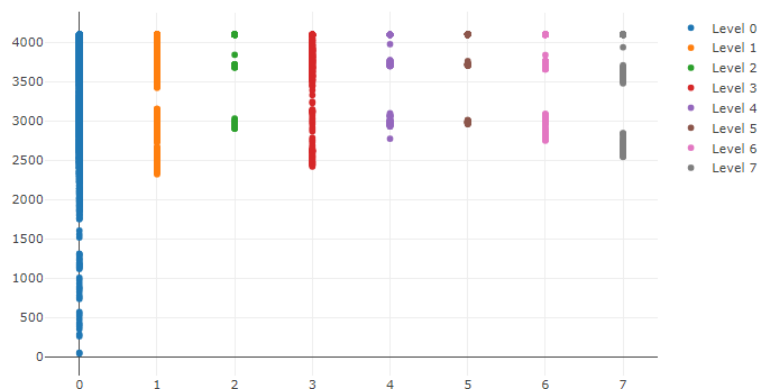


Figure 4.16: Result of the comparison between the ammonia sensor MICSNH3 and the tena levels

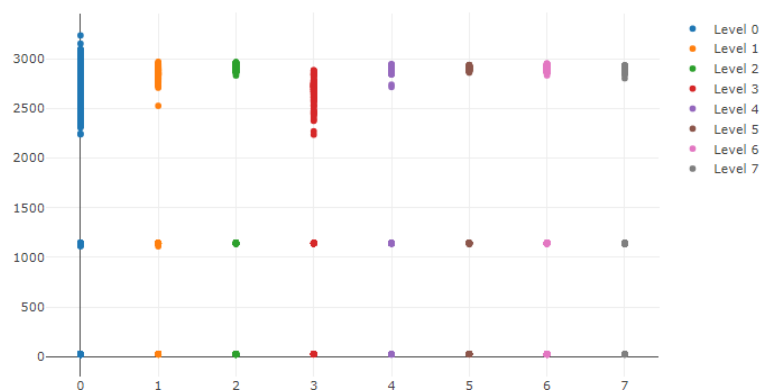


Figure 4.17: Result of the comparison between the ammonia sensor SGXNH3 and the tena levels

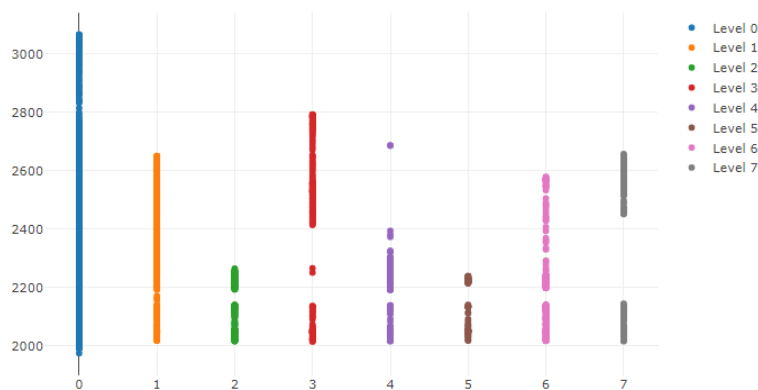


Figure 4.18: Result of the comparison between the ammonia sensor MQS-137 and the tena levels

Where the y axis of the plots are the digital value gas concentration obtained from the microcontroller, and the x axis are the different tena levels of incontinence. So some conclusions can be extracted here: first, this Tena Identifi data had zero measurements with level 8, that is why is not plotted in the graph; after, the sensor module measures the same gas concentration in difference incontinence levels. This could be because of the distance between the resident and the module placement, that the gas is diluted in the air of the room. Or also because the performance of this sensors is not adecuated for the purpouses of the thesis.

This results are also obtained in the laboratory experiment, where it shows that for different gas concentrations the measurement of the sensors was the same. And also proves the results obtained in the visual comparison.

Chapter 5

Conclusion

5.1. General Conclusion

Despite the results of the comparison and the experiment obtained, the objectives of this part of this thesis were achieved. The data can now be structured and plotted easily.

The laboratory experiment could be improved. Due to the results, any characterization could be done. A better incubator more exact when setting temperatures could also help. Also there is the necessity to digitalize that test with the measurements available in the database so the characterization could be easily done. Also the previous thesis explains that the ammonia sensors couldn't be tested because of the inconsistency. This explains the results obtained in the experiment and comparisons. This inconsistency could be caused by the distance of the sensors and the resident. More test should be done.

Still, the comparison program is already built and the characterization of the sensors experiment is detailed, so everything can be tested again once the module is improved.

Also the thesis writer focused more in the laboratory than in the comparison processes. Instead of that, a more deep comparison could have helped more than the laboratory experiment.

5.2. Future work

5.2.1. Improvements to the sensor module

As the results shown, some sensors of the modules aren't working, so they need to be changed or revised. Because the comparison and characterization of the sensors become difficult if they don't work.

Ammonia sensors need to be replaced due to their inconsistency. These sensors need to be tested to know which external factors affect the sensors output so can be chosen better ones.

As the results show, the measurement of the stool can't be tested with this data available. But, in the previous thesis they mention that the chosen H₂S sensors need to be replaced because they are electrochemical cells. Semiconductor hydro-

gen sulfide gas sensors should be used.

5.2.2. Another improvements

Once the sensors placed seem to be the right ones, the characterization of the sensors still need to be done so the conversion rates could be programmed, as explained before in the conclusion.

The experiment results were shown in a real time graphs, but the data had lack of information. With this data, available in the database, a script should plot the graphs in order to to add more information in the graph: axis titles, legend... Also in the mathematical comparison needs to be added this extra information in the plots.

The mathematical comparison can add more information if the humidity sensors and temperature sensors works well. So it needs to be tested again once revised theses sensors. Another useful test that could be done in the future is apply the mathematical comparison already programmed with different measurements of our sensor module placed at different distances of the resident. The distance might affect a lot in terms of sensor inconsistency. This could be easily tested with a distance sensor

Finally, there are some functional aspects of the scripts that needs to be improved: it needs to be done a script that, depending on the season that the measurement has been gathered, add 1 hour or 2 depending if its wintertime or summertime (now the script works well because all the measurements were done in summertime).

Also all the scripts run very slow because they aren't optimized. If this project needs to plot data in real time about incontinence levels, with this programs it will last a few seconds. They need to run faster.

Bibliography

- [E-media] E-MEDIA: E-media Research Lab. . – URL <https://iiw.kuleuven.be/onderzoek/emedi>
- [Heraeus 2019] HERAEUS: Heraeus Incubator. (2019). – URL https://www.geminibv.nl/labware/heraeus-b6-broedstoof-en?pts_language=en&set_language=en&pts_language=en
- [Identify] IDENTIFY, Tena: Tena Identify. . – URL <https://www.tena.nl/professionals/producten/tena-identifi/tena-identifi-sensor-wear/>
- [Latex] LATEX: Latex support forums. . – URL <https://tex.stackexchange.com/>
- [Lorant 2005] LORANT, Karoly: Demographic challenge in Europe. (2005). – URL <http://www.europarl.europa.eu/inddem/docs/papers/The%20demographic%20challenge%20in%20Europe.pdf>
- [Overleaf] OVERLEAF: Overleaf Latex support. . – URL <https://www.overleaf.com/>
- [Pandas] PANDAS: Pandas library. . – URL <https://pandas.pydata.org/>
- [Particle 2019] PARTICLE: Particle Photon. (2019). – URL <https://docs.particle.io/datasheets/wi-fi/photon-datasheet/>
- [Pewatron 2019] PEWATRON: *MiCS 6814 datasheet*. (2019). – URL <https://docs.particle.io/datasheets/wi-fi/photon-datasheet/>
- [Plotly] PLOTLY: Plotly library. . – URL <https://plot.ly/>
- [Putnam 1971] PUTNAM, David F.: Composition and concentrative properties of human urine. (1971). – URL <https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19710023044.pdf>
- [Python] PYTHON: Python support forums. . – URL <https://stackoverflow.com/>
- [SGX-Sensortech 2007a] SGX-SENSORTECH: *MiCS Application Note 2 Frequently Asked Questions for MiCS Gas Sensors*. (2007)
- [SGX-Sensortech 2007b] SGX-SENSORTECH: *MiCS Application Note How to set-up load resistance for MiCS sensors measurements*. (2007)

- [SGX-Sensortech 2015] SGX-SENSORTECH: *Electrochemical Sensors Application Note 2 Design of Electronics for Electrochemical Gas Sensors*. (2015)
- [SGX-Sensortech 2017] SGX-SENSORTECH: *Introduction to Electrochemical (EC) Gas Sensors*. (2017)
- [Simon Kupers 2019] SIMON KUPERS, Kristof V.: Design and development of an incontinence management system. (2019)
- [Thuiszorgwinkel 2019] THUISZORGWINKEL: Tena identifi Sensor Wear. (2019). – URL <https://www.thuiszorgwinkel.be/nl/shop/discrete-zekerheid/incontinentie-bij-mannen/volledig-verlies/bestellen/tena-identifi-sensor-wear>.
- [Wei u. a. 2019] WEI, Peng ; NING, Zhi ; YE, Sheng ; SUN, Li ; YANG, Fenhuan ; WONG, Ka C. ; WESTERDAHL, Dane ; LOUIE, Peter K.: Impact Analysis of Temperature and Humidity Conditions on Electrochemical Sensor Response in Ambient Air Quality Monitoring. In: *Sensors* 18 (2019), Nr. 2, S. 59. – URL <https://www.mdpi.com/1424-8220/18/2/59>

Annexes

A. Data Structure

A.1. Code to structure the data

```
In [ ]: %matplotlib inline
import pandas as pd

In [ ]: pdf_info = pd.read_excel(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20180926 - DiaryMeasurements_Cleaned.xlsx')
#pdf_info

In [ ]: cell_n = input("Introduce the cell you want to choose: ")
cell_n = int(cell_n)-1

In [ ]: selectedCell = pdf_info.loc[cell_n,:]
remarks = selectedCell['remarks']
remarks

In [ ]: selectedCells = pdf_info[pdf_info['remarks']==remarks]
selectedCells = selectedCells.reset_index(drop=True)
lastCell = selectedCells['end'].size-1

In [ ]: dateStart = str(selectedCells['start'][0])
dateStart

In [ ]: dateEnd = str(selectedCells['end'][lastCell])
dateEnd

In [ ]: generalPDF = pd.read_csv(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20181008 - Data_DB.csv')
#generalPDF

In [ ]: cellToStart = generalPDF.index.values[generalPDF['timestamp'] == dateStart]
cellToStart = int(cellToStart)
cellToStart

In [ ]: cellToEnd = generalPDF.index.values[generalPDF['timestamp'] == dateEnd]
cellToEnd = int(cellToEnd)
cellToEnd

In [ ]: rangeCells = generalPDF.iloc[cellToStart:cellToEnd+1]
rangeCells = rangeCells.reset_index(drop=True)
rangeCells

In [ ]: path = 'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\'
rangeCells.to_csv(path+remarks+".csv", index = False)
```

Figure 1: Code to create new Excel with the information provided in other Excels

A.2. Script flow diagram

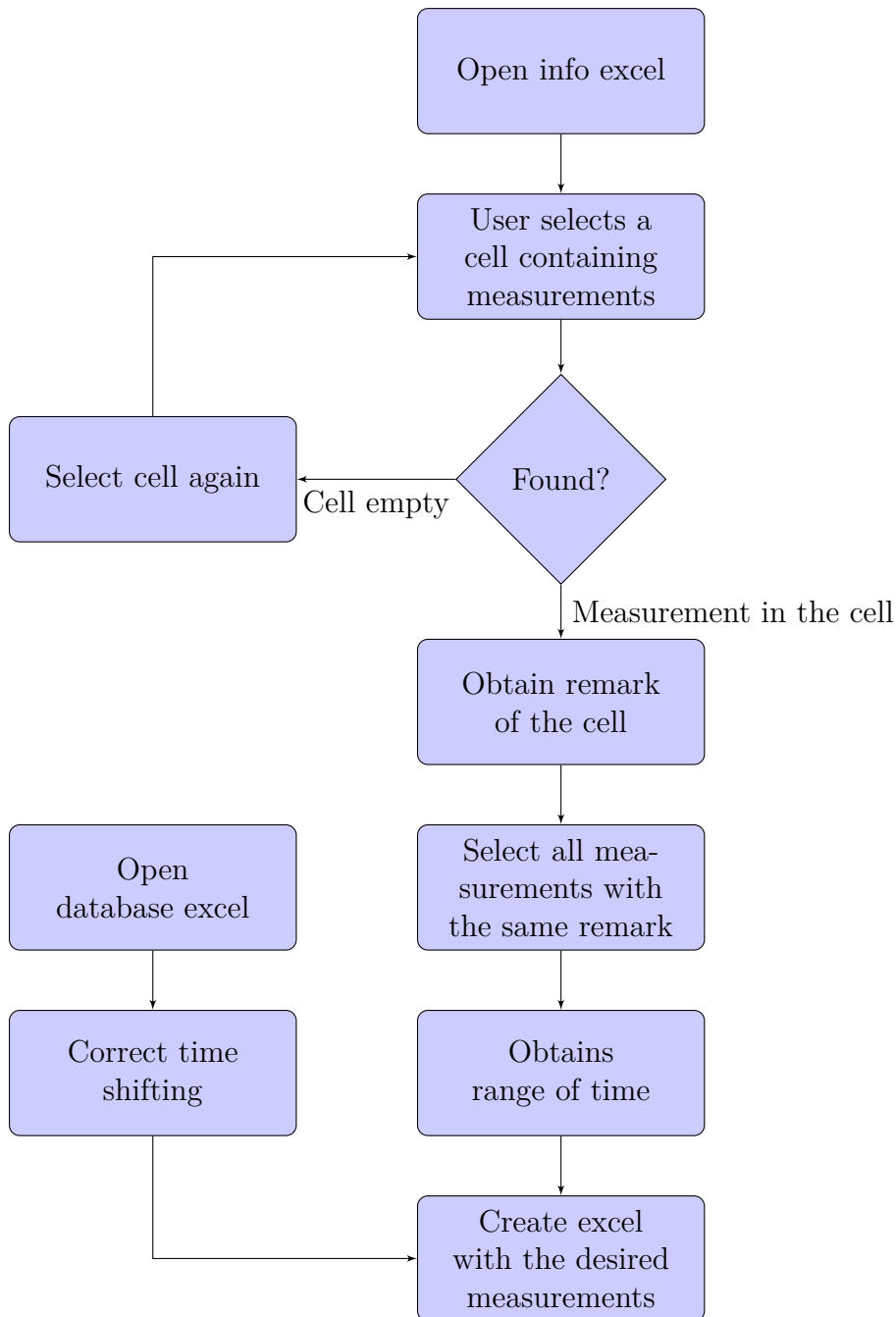


Figure 2: Flow diagram of the data structuration

A.3. Script results

```

In [1]: %matplotlib inline
import pandas as pd

In [2]: pdf_info = pd.read_excel(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20180926 - DiaryMeasurements_cleaned.xlsx')
#pdf_info

In [3]: cell_n = input("Introduce the cell you want to choose: ")
cell_n = int(cell_n)-1
Introduce the cell you want to choose: 317

In [4]: selectedCell = pdf_info.loc[cell_n,:]
remarks = selectedCell['remarks']
remarks
Out[4]: 'TENA measurement test'

In [5]: selectedCells = pdf_info[pdf_info['remarks']==remarks]
selectedCells = selectedCells.reset_index(drop=True)
lastCell = selectedCells['end'].size-1

In [6]: dateStart = str(selectedCells['start'][0])
dateStart
Out[6]: '2018-07-30 00:00:10'

In [7]: dateEnd = str(selectedCells['end'][lastCell])
dateEnd
Out[7]: '2018-08-02 23:59:38'

In [8]: generalPDF = pd.read_csv(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20181008 - Data_DB.csv')
#generalPDF

In [9]: cellToStart = generalPDF.index.values[generalPDF['timestamp'] == dateStart]
cellToStart = int(cellToStart)
cellToStart
Out[9]: 173644

In [10]: cellToEnd = generalPDF.index.values[generalPDF['timestamp'] == dateEnd]
cellToEnd = int(cellToEnd)
cellToEnd
Out[10]: 195481

In [11]: rangeCells = generalPDF.iloc[cellToStart:cellToEnd+1]
rangeCells = rangeCells.reset_index(drop=True)
rangeCells

```

6	179143	2018-07-30 00:03:40	1.0	360059000f51353532343635	4	3732	NaN	NaN	26	3549	2045
7	179144	2018-07-30 00:03:51	1.0	360059000f51353532343635	1	3731	NaN	NaN	27	2352	2057
8	179145	2018-07-30 00:04:32	NaN	32002f000b51353432383931	10	4000	NaN	NaN	1142	4	2143
9	179146	2018-07-30 00:04:49	1.0	360059000f51353532343635	5	3747	NaN	NaN	27	2898	2052
10	179147	2018-07-30 00:06:06	NaN	32002f000b51353432383931	9	3981	NaN	NaN	1143	4	2133
11	179148	2018-07-30 00:06:33	1.0	360059000f51353532343635	10	3748	NaN	NaN	27	2826	2044

```

In [12]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
rangeCells.to_csv(path+remarks+".csv", index = False)

```

Figure 3: Code to structure the data with the outputs once executed

B. Data graphs

B.1. Code to plot the data

```
In [ ]: %matplotlib inline

import pandas as pd
import plotly as py
import plotly.graph_objs as go
import matplotlib.pyplot as plot
import import_ipynb
import plotly.io as pio

py.offline.init_notebook_mode(connected=True)

In [ ]: cell_n = input("Introduce the cell you want to choose: ")
cell_n = int(cell_n)-1

In [ ]: pdf_info = pd.read_excel(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20180926 - DiaryMeasurements_Cleaned.xlsx')
selectedCell = pdf_info.loc[cell_n,:]
remarks = selectedCell['remarks']
remarks

In [ ]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
plotPDF = pd.read_csv(path+remarks+'.csv')
plotPDF

In [ ]: #import gasConversion

#plotPDF = gasConversion.gas_conversion(plotPDF)
#plotPDF

In [ ]: import IDauthenticator

plotPDF1,plotPDF2,plotPDF3,plotPDF4,plotPDF5,plotPDF6,plotPDF7 = IDauthenticator.change_id(plotPDF)

plotPDF1=plotPDF1.reset_index(drop=True)
plotPDF2=plotPDF2.reset_index(drop=True)
plotPDF3=plotPDF3.reset_index(drop=True)
plotPDF4=plotPDF4.reset_index(drop=True)
plotPDF5=plotPDF5.reset_index(drop=True)
plotPDF6=plotPDF6.reset_index(drop=True)
plotPDF7=plotPDF7.reset_index(drop=True)

In [ ]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
plotPDF.to_csv(path+remarks+"-conversion.csv", index = False)
```

```

In [ ]: import plot9

if(plotPDF1.empty==False):
    fig1 = plot9.plot_graphics(plotPDF1)
    nameimage1 = remarks+"("+plotPDF1["device_id"][0]+")"+" general plot'
    pio.write_image(fig1,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage1+'.png')

if(plotPDF2.empty==False):
    fig2 = plot9.plot_graphics(plotPDF2)
    nameimage2 = remarks+"("+plotPDF2["device_id"][0]+")"+" general plot'
    pio.write_image(fig2,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage2+'.png')

if(plotPDF3.empty==False):
    fig3 = plot9.plot_graphics(plotPDF3)
    nameimage3 = remarks+"("+plotPDF3["device_id"][0]+")"+" general plot'
    pio.write_image(fig3,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage3+'.png')

if(plotPDF4.empty==False):
    fig4 = plot9.plot_graphics(plotPDF4)
    nameimage4 = remarks+"("+plotPDF4["device_id"][0]+")"+" general plot'
    pio.write_image(fig4,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage4+'.png')

if(plotPDF5.empty==False):
    fig5 = plot9.plot_graphics(plotPDF5)
    nameimage5 = remarks+"("+plotPDF5["device_id"][0]+")"+" general plot'
    pio.write_image(fig5,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage5+'.png')

if(plotPDF6.empty==False):
    fig6 = plot9.plot_graphics(plotPDF6)
    nameimage6 = remarks+"("+plotPDF6["device_id"][0]+")"+" general plot'
    pio.write_image(fig6,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage6+'.png')

if(plotPDF7.empty==False):
    fig7 = plot9.plot_graphics(plotPDF7)
    nameimage7 = remarks+"("+plotPDF7["device_id"][0]+")"+" general plot'
    pio.write_image(fig7,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage7+'.png')

In [ ]: import FocusedPlot

sensor_n = int(input("Introduce the sensor do you want to view, from 1 to 9: "))
fig,sensor_name = FocusedPlot.focused_plot(plotPDF,sensor_n)
py.offline.iplot(fig)

nameimage = remarks+' '+sensor_name+' sensor plot'
pio.write_image(fig,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage+'.png')

```

Figure 4: Code to plot the data values

B.2. Id identification

```
In [1]: def change_id(plotPDF):  
    plotPDF1 = plotPDF[plotPDF["device_id"]=="2c003f000c47343438323536"]  
    plotPDF1.loc[:, "device_id"] = "Hannelore"  
  
    plotPDF2 = plotPDF[plotPDF["device_id"]=="32002f000b51353432383931"]  
    plotPDF2.loc[:, "device_id"] = "Kate_Brewster"  
  
    plotPDF3 = plotPDF[plotPDF["device_id"]=="360059000f51353532343635"]  
    plotPDF3.loc[:, "device_id"] = "Kyle_Reese"  
  
    plotPDF4 = plotPDF[plotPDF["device_id"]=="360059000f51353532343635"]  
    plotPDF4.loc[:, "device_id"] = "John_Connor"  
  
    plotPDF5 = plotPDF[plotPDF["device_id"]=="38005d000d51353532343635"]  
    plotPDF5.loc[:, "device_id"] = "Sarah_Connor"  
  
    plotPDF6 = plotPDF[plotPDF["device_id"]=="440020000d51353532343635"]  
    plotPDF6.loc[:, "device_id"] = "Darth_Vader"  
  
    plotPDF7 = plotPDF[plotPDF["device_id"]=="440060000d51353532343635"]  
    plotPDF7.loc[:, "device_id"] = "Peter_Silberman"  
  
    return (plotPDF1, plotPDF2, plotPDF3, plotPDF4, plotPDF5, plotPDF6, plotPDF7)
```

Figure 5: Code to separate all the data from the different sensor modules

B.3. Plot all the sensor module data

```
In [1]: def plot_graphics(plotPDF):  
import plotly as py  
import plotly.graph_objs as go  
  
trace1 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['MICSNH3'],  
    mode = 'markers',  
    name = 'MICSNH3 sensor'  
)  
  
trace2 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['SGXNH3'],  
    xaxis = 'x2',  
    yaxis = 'y2',  
    mode = 'markers',  
    name = 'SGXNH3 sensor'  
)  
  
trace3 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['SGXH2S'],  
    xaxis = 'x3',  
    yaxis = 'y3',  
    mode = 'markers',  
    name = 'SGXH2S sensor'  
)  
  
trace4 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['SPH2SH2S'],  
    xaxis = 'x4',  
    yaxis = 'y4',  
    mode = 'markers',  
    name = 'SPH2SH2S sensor'  
)  
  
trace5 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['SPH2STEMP'],  
    xaxis = 'x5',  
    yaxis = 'y5',  
    mode = 'markers',  
    name = 'SPH2STEMP sensor'  
)  
  
trace6 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['DHTEMP'],  
    xaxis = 'x6',  
    yaxis = 'y6',  
    mode = 'markers',  
    name = 'DHTEMP sensor'  
)  
  
trace7 = go.Scatter(  
    x = plotPDF['timestamp'],  
    y = plotPDF['DHTHUM'],  
    xaxis = 'x7',  
    yaxis = 'y7',  
    mode = 'markers',  
    name = 'DHTHUM sensor'  
)
```

```
trace8= go.Scatter(  
  x = plotPDF['timestamp'],  
  y = plotPDF['LDR'],  
  xaxis = 'x8',  
  yaxis = 'y8',  
  mode = 'markers',  
  name = 'LDR sensor'  
)  
  
trace9= go.Scatter(  
  x = plotPDF['timestamp'],  
  y = plotPDF['IMX'],  
  xaxis = 'x9',  
  yaxis = 'y9',  
  mode = 'markers',  
  name = 'IMX sensor'  
)  
  
data = [trace1, trace2, trace3, trace4, trace5, trace6, trace7, trace8, trace9]  
layout = go.Layout(  
  xaxis=dict(  
    domain=[0, 0.2]  
  ),  
  yaxis=dict(  
    domain=[0, 0.2]  
  ),  
  xaxis2=dict(  
    domain=[0.4, 0.6]  
  ),  
  yaxis2=dict(  
    domain=[0, 0.2],  
    anchor='x2'  
  ),  
  xaxis3=dict(  
    domain=[0.8, 1]  
  ),  
  xaxis4=dict(  
    domain=[0, 0.2]  
  ),  
  yaxis4=dict(  
    domain=[0.4, 0.6]  
  ),  
  xaxis5=dict(  
    domain=[0.4, 0.6]  
  ),  
  yaxis5=dict(  
    domain=[0.4, 0.6],  
    anchor='x5'  
  ),  
  xaxis6=dict(  
    domain=[0.8, 1]  
  ),  
  yaxis6=dict(  
    domain=[0.4, 0.6],  
    anchor='x6'  
  ),  
  xaxis7=dict(  
    domain=[0, 0.2]  
  ),  
  yaxis7=dict(  
    domain=[0.8, 1]  
  ),  
  xaxis8=dict(  
    domain=[0.4, 0.6]  
  ),  
  yaxis8=dict(  
    domain=[0.8, 1],  
    anchor='x8'  
  ),  
  xaxis9=dict(  
    domain=[0.8, 1]  
  ),  
  . . . . .  
)
```

```
yaxis=dict(  
    domain=[0.8, 1],  
    anchor='x9'  
)  
fig = go.Figure(data=data, layout=layout)  
return fig
```

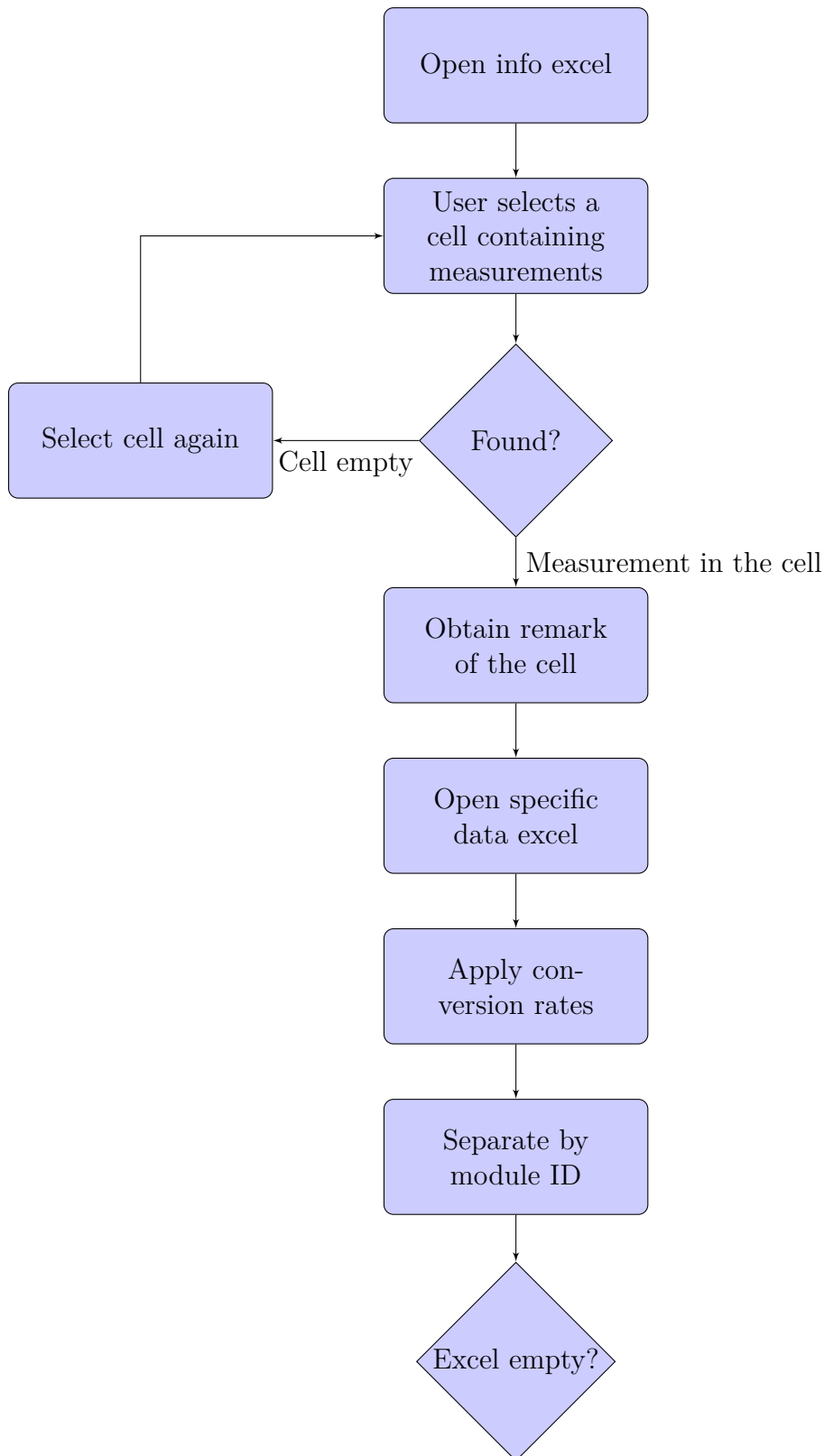
Figure 6: Code to plot the data values

B.4. Focused plot

```
In [1]: def focused_plot (plotPDF,sensor_n):  
  
    import plotly as py  
    import plotly.graph_objs as go  
  
    sensor_names = ["MICSNH3","SGXNH3","SGXH2S","SPH2SH2S","SPH2STEMP","DHTTEMP","DHTHUM","LDR","IMX"]  
    yaxis_names = ["Ammonia Concentration","Ammonia Concentration","Hidrogensulfide Concentration",  
                  ["Hidrogensulfide Concentration","Temperature","Temperature","Relative Humidity","Presence","Movement"]  
  
    trace = go.Scatter(  
        x = plotPDF['timestamp'],  
        y = plotPDF[sensor_names[sensor_n]],  
        mode = 'markers',  
        name = sensor_names[sensor_n]  
    )  
  
    layout= go.Layout(  
        title = sensor_names[sensor_n],  
        hovermode= 'closest',  
  
        xaxis= dict(  
            title= 'Time',  
            ticklen= 5,  
            zeroline= False,  
            gridwidth= 2,  
        ),  
  
        yaxis=dict(  
            title= yaxis_names[sensor_n],  
            ticklen= 5,  
            gridwidth= 2,  
        ),  
  
        showlegend= False  
    )  
  
    data = [trace]  
    fig = go.Figure(data=data,layout=layout)  
    return (fig, sensor_names[sensor_n])
```

Figure 7: Code to plot only the desired sensor values

B.5. Script flow diagram



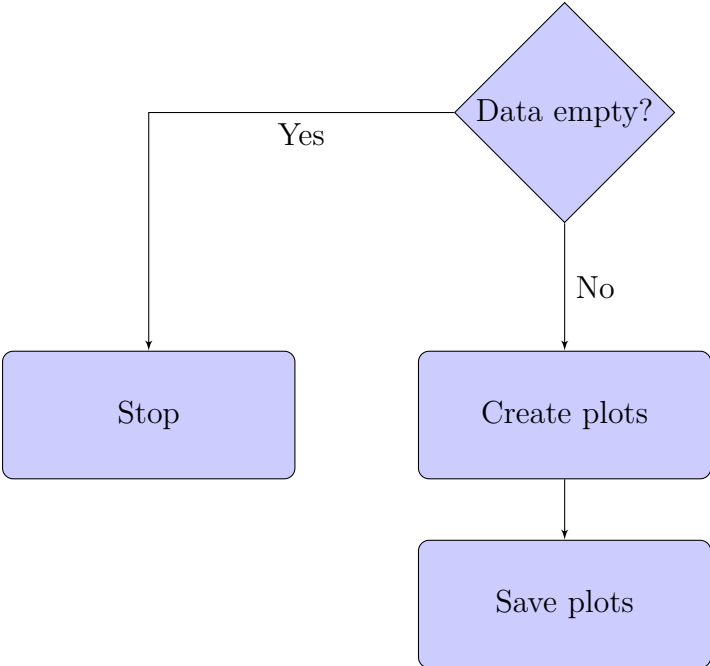


Figure 8: Flow diagram of the data plotting

B.6. Existing solutions

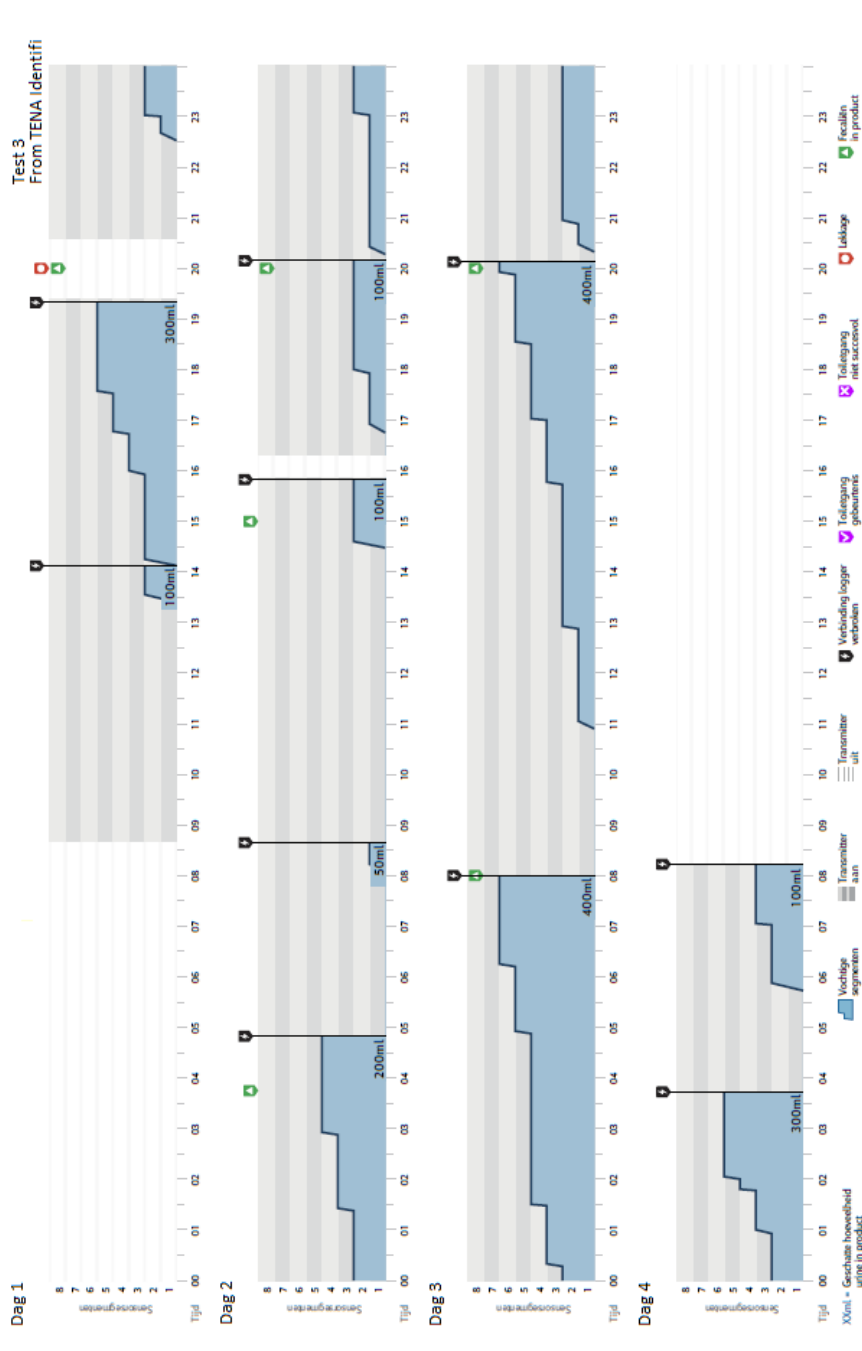



Figure 9: Tena measurement example



TOILETING REGISTRATIE - GRAAG INVULLEN GEDURENDE 72 UUR METING VANAF START LOGGER:

kamer nummer: 003 3017 Naam bewoner: _____ Nummer logger: _____ startdatum 72 uur meting: _____

START DATUM: 003 3017

ZET EEN X BIJ JUISTE TIJD	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Lekkage	X																							
Ontlasting in het product	X																							
Toilet -NIET succesvol	X																							
Toilet bezoek succesvol	X												X											
Medewerker (initialen) J.D.																								
Noteer andere observaties die het resultaat beïnvloeden																								
Dag 2 DATUM: <u>003 3017</u>																								
ZET EEN X BIJ JUISTE TIJD	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Lekkage	X							X																
Ontlasting in het product	X																							
Toilet -NIET succesvol	X																							
Toilet bezoek succesvol	X							X																
Medewerker (initialen) J.D.																								
Noteer andere observaties die het resultaat beïnvloeden																								
Dag 3 DATUM: <u>003 118</u>																								
ZET EEN X BIJ JUISTE TIJD	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Lekkage	X																							
Ontlasting in het product	X																							
Toilet -NIET succesvol	X																							
Toilet bezoek succesvol	X						X								X									
Medewerker (initialen) J.D.																								
Noteer andere observaties die het resultaat beïnvloeden																								
EIND DAG DATUM: <u>003 218</u>																								
ZET EEN X BIJ JUISTE TIJD	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Lekkage	X																							
Ontlasting in het product	X																							
Toilet -NIET succesvol	X																							
Toilet bezoek succesvol	X																							
Medewerker (initialen) J.D.																								
Noteer andere observaties die het resultaat beïnvloeden																								

Figure 10: Tena measurement example

B.7. Code to digitalize Tena Identifi results

```
In [ ]: from PIL import Image

import pandas as pd
import plotly as py
import plotly.graph_objs as go
import import_ipynb
import plotly.io as pio

py.offline.init_notebook_mode(connected=True)

In [ ]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
remarks = "TenaTestMeasurements"
tenaPDF = pd.read_excel(path+remarks+'.xlsx',header=None,names=["Date","Level","Change logger","Leakage","Outlasting in the produc
tenaPDF

In [ ]: import TenaPlot

fig = TenaPlot.tena_plot(tenaPDF)
py.offline.iplot(fig)

nameimage = "Tena plot"
pio.write_image(fig,'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Plots\\'+nameimage+'.png')
```

Figure 11: Code to digitalize the Tena Identifi values

B.8. Plotting the digital values of Tena identi

```

In [1]: def tena_plot (tenaPDF):

import plotly as py
import plotly.graph_objs as go

trace1 = go.Bar(
    x = tenaPDF['Date'],
    y = tenaPDF['Level'],
    xaxis = 'x1',
    yaxis = 'y1',
    name = "Level of incontinence saturation"
)

trace2 = go.Bar(
    x = tenaPDF['Date'],
    y = tenaPDF['Change logger'],
    xaxis = 'x2',
    yaxis = 'y2',
    name = "Change logger"
)

trace3 = go.Bar(
    x = tenaPDF['Date'],
    y = tenaPDF['Leakage'],
    xaxis = 'x2',
    yaxis = 'y2',
    name = "Leakage"
)

trace4 = go.Bar(
    x = tenaPDF['Date'],
    y = tenaPDF['Outlasting in the product'],
    xaxis = 'x2',
    yaxis = 'y2',
    name = "Outlasting in the product"
)

trace5 = go.Bar(
    x = tenaPDF['Date'],
    y = tenaPDF['Non-successful in the bathroom'],
    xaxis = 'x2',
    yaxis = 'y2',
    name = "Non-successful in the bathroom"
)

trace6 = go.Bar(
    x = tenaPDF['Date'],
    y = tenaPDF['Successful in the bathroom'],
    xaxis = 'x2',
    yaxis = 'y2',
    name = "Successful in the bathroom"
)

data = [trace1, trace2, trace3, trace4, trace5, trace6]
layout= go.Layout(
    title = "Incontinence monitoring with Tena Identifi",
    barmode= 'stack',

    xaxis = dict(
        domain=[0, 1],
        title = "Time"
    ),
    yaxis = dict(
        domain = [0.6, 1],
        title = "Level of incontinence saturation"
    ),
    xaxis2 = dict(
        domain=[0, 1],
    ),
    yaxis2 = dict(
        domain=[0, 0.4],
        anchor='x2',
        title = "Involved aspects"
    )
)

fig = go.Figure(data=data,layout=layout)
return (fig)

```

Figure 12: Code to plot the Tena Identifi values

C. Mathematical comparison

C.1. Comparison script

```
In [ ]: from PIL import Image

import pandas as pd
import plotly as py
import plotly.graph_objs as go
import import_ipynb
import plotly.io as pio
import time
from datetime import datetime

py.offline.init_notebook_mode(connected=True)

In [ ]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
remarks = "TenaTestMeasurements"
tenaPDF = pd.read_excel(path+remarks+'.xlsx',header=None,names=["Date","Level","Change logger","Leakage"])

In [ ]: tena_inclvl_0 = tenaPDF[tenaPDF["Level"]==0]
tena_inclvl_1 = tenaPDF[tenaPDF["Level"]==1]
tena_inclvl_2 = tenaPDF[tenaPDF["Level"]==2]
tena_inclvl_3 = tenaPDF[tenaPDF["Level"]==3]
tena_inclvl_4 = tenaPDF[tenaPDF["Level"]==4]
tena_inclvl_5 = tenaPDF[tenaPDF["Level"]==5]
tena_inclvl_6 = tenaPDF[tenaPDF["Level"]==6]
tena_inclvl_7 = tenaPDF[tenaPDF["Level"]==7]
tena_inclvl_8 = tenaPDF[tenaPDF["Level"]==8]

In [ ]: tena_lvls = list()

if (tena_inclvl_0.empty == False):
    tena_lvls.append(tena_inclvl_0["Level"].iloc[0])

if (tena_inclvl_1.empty == False):
    tena_lvls.append(tena_inclvl_1["Level"].iloc[0])

if (tena_inclvl_3.empty == False):
    tena_lvls.append(tena_inclvl_3["Level"].iloc[0])

if (tena_inclvl_4.empty == False):
    tena_lvls.append(tena_inclvl_4["Level"].iloc[0])

if (tena_inclvl_5.empty == False):
    tena_lvls.append(tena_inclvl_5["Level"].iloc[0])

if (tena_inclvl_6.empty == False):
    tena_lvls.append(tena_inclvl_6["Level"].iloc[0])

if (tena_inclvl_7.empty == False):
    tena_lvls.append(tena_inclvl_7["Level"].iloc[0])
```

```
if (tena_inclvl_8.empty == False):
    tena_lvls.append(tena_inclvl_8["Level"].iloc[0])
```

```
In [ ]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
dataPDF = pd.read_csv(path+"TENA measurement test.csv")
```

```
In [ ]: rlist = dataPDF["timestamp"].to_csv(None, header=False,
index=False).split('\n')
rlist = rlist[:-1]
```

```
In [ ]: clist = list()
i=1
for index, item in enumerate(rlist):
    clist.append(datetime.strptime(rlist[index], '%Y-%m-%d %H:%M:%S'))
```

```
In [ ]: nlist = list()
for item in clist:
    nlist.append(pd.to_datetime(item.date().strftime('%Y-%m-%d')+ ' ' + item.time().strftime('%H:%M:%S')))
```

```
In [ ]: import separateDates
import fnearestDate
import selectData

data_collection = {}

if (tena_inclvl_0.empty == False):
    dates_start_0 = list()
    dates_end_0 = list()
    start_cells_0 = list()
    end_cells_0 = list()
    data_0 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_0, dates_end_0 = separateDates.separate_dates(tena_inclvl_0, dates_start_0, dates_end_0)
    start_cells_0, end_cells_0 = fnearestDate.find_nearest_date(nlist, dates_start_0, dates_end_0, start_cells_0, end_cells_0)
    data_0 = selectData.select_data(dataPDF, start_cells_0, end_cells_0, data_0)
    data_0['Level'].iloc[0:data_0['timestamp'].size] = 0
    data_collection[tena_inclvl_0['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM35TEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_0['Level'].iloc[0]] = data_0
```

```
if (tena_inclvl_1.empty == False):
    dates_start_1 = list()
    dates_end_1 = list()
    start_cells_1 = list()
    end_cells_1 = list()
    data_1 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_1, dates_end_1 = separateDates.separate_dates(tena_inclvl_1, dates_start_1, dates_end_1)
    start_cells_1, end_cells_1 = fnearestDate.find_nearest_date(nlist, dates_start_1, dates_end_1, start_cells_1, end_cells_1)
    data_1 = selectData.select_data(dataPDF, start_cells_1, end_cells_1, data_1)
    data_1['Level'].iloc[0:data_1['timestamp'].size] = 1
    data_collection[tena_inclvl_1['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM35TEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_1['Level'].iloc[0]] = data_1
```

```
if (tena_inclvl_2.empty == False):
    dates_start_2 = list()
    dates_end_2 = list()
    start_cells_2 = list()
    end_cells_2 = list()
    data_2 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_2, dates_end_2 = separateDates.separate_dates(tena_inclvl_2, dates_start_2, dates_end_2)
    start_cells_2, end_cells_2 = fnearestDate.find_nearest_date(nlist, dates_start_2, dates_end_2, start_cells_2, end_cells_2)
    data_2 = selectData.select_data(dataPDF, start_cells_2, end_cells_2, data_2)
    data_2['Level'].iloc[0:data_2['timestamp'].size] = 2
    data_collection[tena_inclvl_2['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM35TEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_2['Level'].iloc[0]] = data_2
```

```
if (tena_inclvl_3.empty == False):
    dates_start_3 = list()
    dates_end_3 = list()
    start_cells_3 = list()
    end_cells_3 = list()
    data_3 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_3, dates_end_3 = separateDates.separate_dates(tena_inclvl_3, dates_start_3, dates_end_3)
    start_cells_3, end_cells_3 = fnearestDate.find_nearest_date(nlist, dates_start_3, dates_end_3, start_cells_3, end_cells_3)
    data_3 = selectData.select_data(dataPDF, start_cells_3, end_cells_3, data_3)
    data_3['Level'].iloc[0:data_3['timestamp'].size] = 3
    data_collection[tena_inclvl_3['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM35TEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_3['Level'].iloc[0]] = data_3
```

```

if (tena_inclvl_4.empty == False):
    dates_start_4 = list()
    dates_end_4 = list()
    start_cells_4 = list()
    end_cells_4 = list()
    data_4 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_4, dates_end_4 = separateDates.separate_dates(tena_inclvl_4, dates_start_4, dates_end_4)
    start_cells_4, end_cells_4 = fnearestDate.find_nearest_date(nlist, dates_start_4, dates_end_4, start_cells_4, end_cells_4)
    data_4 = selectData.select_data(dataPDF, start_cells_4, end_cells_4, data_4)
    data_4['Level'].iloc[0: data_4['timestamp'].size] = 4
    data_collection[tena_inclvl_4['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM3STEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_4['Level'].iloc[0]] = data_4

if (tena_inclvl_5.empty == False):
    dates_start_5 = list()
    dates_end_5 = list()
    start_cells_5 = list()
    end_cells_5 = list()
    data_5 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_5, dates_end_5 = separateDates.separate_dates(tena_inclvl_5, dates_start_5, dates_end_5)
    start_cells_5, end_cells_5 = fnearestDate.find_nearest_date(nlist, dates_start_5, dates_end_5, start_cells_5, end_cells_5)
    data_5 = selectData.select_data(dataPDF, start_cells_5, end_cells_5, data_5)
    data_5['Level'].iloc[0: data_5['timestamp'].size] = 5
    data_collection[tena_inclvl_5['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM3STEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_5['Level'].iloc[0]] = data_5

if (tena_inclvl_6.empty == False):
    dates_start_6 = list()
    dates_end_6 = list()
    start_cells_6 = list()
    end_cells_6 = list()
    data_6 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_6, dates_end_6 = separateDates.separate_dates(tena_inclvl_6, dates_start_6, dates_end_6)
    start_cells_6, end_cells_6 = fnearestDate.find_nearest_date(nlist, dates_start_6, dates_end_6, start_cells_6, end_cells_6)
    data_6 = selectData.select_data(dataPDF, start_cells_6, end_cells_6, data_6)
    data_6['Level'].iloc[0: data_6['timestamp'].size] = 6
    data_collection[tena_inclvl_6['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM3STEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_6['Level'].iloc[0]] = data_6

if (tena_inclvl_8.empty == False):
    dates_start_8 = list()
    dates_end_8 = list()
    start_cells_8 = list()
    end_cells_8 = list()
    data_8 = pd.DataFrame(columns=['timestamp', 'Level'])
    dates_start_8, dates_end_8 = separateDates.separate_dates(tena_inclvl_8, dates_start_8, dates_end_8)
    start_cells_8, end_cells_8 = fnearestDate.find_nearest_date(nlist, dates_start_8, dates_end_8, start_cells_8, end_cells_8)
    data_8 = selectData.select_data(dataPDF, start_cells_8, end_cells_8, data_8)
    data_8['Level'].iloc[0: data_8['timestamp'].size] = 8
    data_collection[tena_inclvl_8['Level'].iloc[0]] = pd.DataFrame(columns=['id', 'timestamp', 'resident_id', 'MICSNH3', 'MICSNO2', 'MICSNO', 'SGXNH3', 'SPH2SH2S', 'SPH2STEMP', 'HIHTEMP', 'HIHHUM', 'DHTTEMP', 'DHTHUM', 'LM3STEMP', 'LDR', 'IMX', 'data'])
    data_collection[tena_inclvl_8['Level'].iloc[0]] = data_8

In [ ]: import printComparison
data_plot = list()
data_plot = printComparison.print_comparison(data_collection, tena_lvls, data_plot)
py.offline.ipplot(data_plot)

```

Figure 13: Code to compare the applications

C.2. Script to select the data intervals

```
In [1]: def separate_dates(tenaPDF,dates_start,dates_end):
        actual_index = 0
        previous_index = 0

        dates_start.append(tenaPDF["Date"].iloc[0])

        for item in tenaPDF["Date"].index.values:
            previous_index = actual_index
            actual_index = item

            if(previous_index != 0):
                if(actual_index != previous_index+1):
                    dates_start.append(tenaPDF["Date"].iloc[actual_index])
                    dates_end.append(tenaPDF["Date"].iloc[previous_index])

        dates_end.append(tenaPDF["Date"].iloc[tenaPDF["Date"].size-1])

        return (dates_start, dates_end)
```

Figure 14: Code to select the data intervals

C.3. Script to find nearest data

```
In [4]: def find_nearest_date(nlist,dates_start,dates_end,start_cells,end_cells):
import pandas as pd

aux = list()
for x in dates_start:
    if(dates_start[dates_start.index(x)]==dates_end[dates_start.index(x)]):
        date_eq_n_list = [time for time in nlist if (time-x)<pd.to_timedelta(0)]
        for y in end_n_eq_list:

            newtime = y-x
            if (end_n_eq_list[0]==y):
                minimum_eq_end = newtime
                date_min_end = y

            elif (newtime > minimum_eq_end):
                minimum_eq_end = newtime
                date_min_eq = y

        start_cells.append(nlist.index(date_min_eq))
        end_cells.append(nlist.index(date_min_start)+1)

for x in dates_start:
    if(dates_start.index(x) not in aux):
        start_n_list = [time for time in nlist if (time-x)>pd.to_timedelta(0)]
        for y in start_n_list:

            newtime = y-x
            if (start_n_list[0]==y):
                minimum_start = newtime
                date_min_start = y

            elif (newtime < minimum_start):
                minimum_start = newtime
                date_min_start = y

        start_cells.append(nlist.index(date_min_start))

for x in dates_end:
    if(dates_end.index(x) not in aux):
        end_n_list = [time for time in nlist if (time-x)<pd.to_timedelta(0)]
        for y in end_n_list:

            newtime = y-x
            if (end_n_list[0]==y):
                minimum_end = newtime
                date_min_end = y

            elif (newtime > minimum_end):
                minimum_end = newtime
                date_min_end = y

        end_cells.append(nlist.index(date_min_end))

return start_cells,end_cells
```

Figure 15: Code to find the nearest data

C.4. Script to select the data in the range required

```

In [ ]: %matplotlib inline
import pandas as pd

In [ ]: pdf_info = pd.read_excel(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20180926 - DiaryMeasurements_Cleaned.xlsx')
#pdf_info

In [ ]: cell_n = input("Introduce the cell you want to choose: ")
cell_n = int(cell_n)-1

In [ ]: selectedCell = pdf_info.loc[cell_n,:]
remarks = selectedCell['remarks']

In [ ]: selectedCells = pdf_info[pdf_info['remarks']==remarks]
selectedCells = selectedCells.reset_index(drop=True)
lastCell = selectedCells['end'].size-1

In [ ]: dateStart = str(selectedCells['start'][0])
dateStart

In [ ]: dateEnd = str(selectedCells['end'][lastCell])
dateEnd

In [ ]: generalPDF = pd.read_csv(r'C:\Users\Andreu Ortega Blasi\Desktop\Uni\TFG\Measurements\20181008 - Data_DB.csv')
#generalPDF

In [ ]: cellToStart = generalPDF.index.values[generalPDF['timestamp'] == dateStart]
cellToStart = int(cellToStart)
cellToStart

In [ ]: cellToEnd = generalPDF.index.values[generalPDF['timestamp'] == dateEnd]
cellToEnd = int(cellToEnd)
cellToEnd

In [ ]: rangeCells = generalPDF.iloc[cellToStart:cellToEnd+1]
rangeCells = rangeCells.reset_index(drop=True)
rangeCells

In [ ]: path = 'C:\\Users\\Andreu Ortega Blasi\\Desktop\\Uni\\TFG\\Measurements\\'
rangeCells.to_csv(path+remarks+".csv", index = False)

```

Figure 16: Code select the data in the range required

C.5. Script to plot the comparison

```
In [2]: def print_comparison(data,tena_lvls,data_plot):

import plotly as py
import plotly.graph_objs as go

if(0 in tena_lvls):
    trace0 = go.Scatter(
        x = data[0]["Level"],
        y = data[0]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 0'
    )
    data_plot.append(trace0)

if(1 in tena_lvls):
    trace1 = go.Scatter(
        x = data[1]["Level"],
        y = data[1]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 1'
    )
    data_plot.append(trace1)

if(2 in tena_lvls):
    trace2 = go.Scatter(
        x = data[2]["Level"],
        y = data[2]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 2'
    )
    data_plot.append(trace2)

if(3 in tena_lvls):
    trace3 = go.Scatter(
        x = data[3]["Level"],
        y = data[3]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 3'
    )
    data_plot.append(trace3)

if(4 in tena_lvls):
    trace4 = go.Scatter(
        x = data[4]["Level"],
        y = data[4]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 4'
    )
    data_plot.append(trace4)

if(5 in tena_lvls):
    trace5 = go.Scatter(
        x = data[5]["Level"],
        y = data[5]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 5'
    )
    data_plot.append(trace5)

if(6 in tena_lvls):
    trace6 = go.Scatter(
        x = data[6]["Level"],
        y = data[6]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 6'
    )
    data_plot.append(trace6)

if(7 in tena_lvls):
    trace7 = go.Scatter(
        x = data[7]["Level"],
        y = data[7]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 7'
    )
    data_plot.append(trace7)

if(8 in tena_lvls):
    trace8 = go.Scatter(
        x = data[8]["Level"],
        y = data[8]["MICSNH3"], #"SGXNH3","SPH2SH2S"
        mode = 'markers',
        name = 'Level 8'
    )
    data_plot.append(trace8)

return data_plot
```

Figure 17: Code to obtain the traces of the desired plot

C.6. Comparison script flow diagram

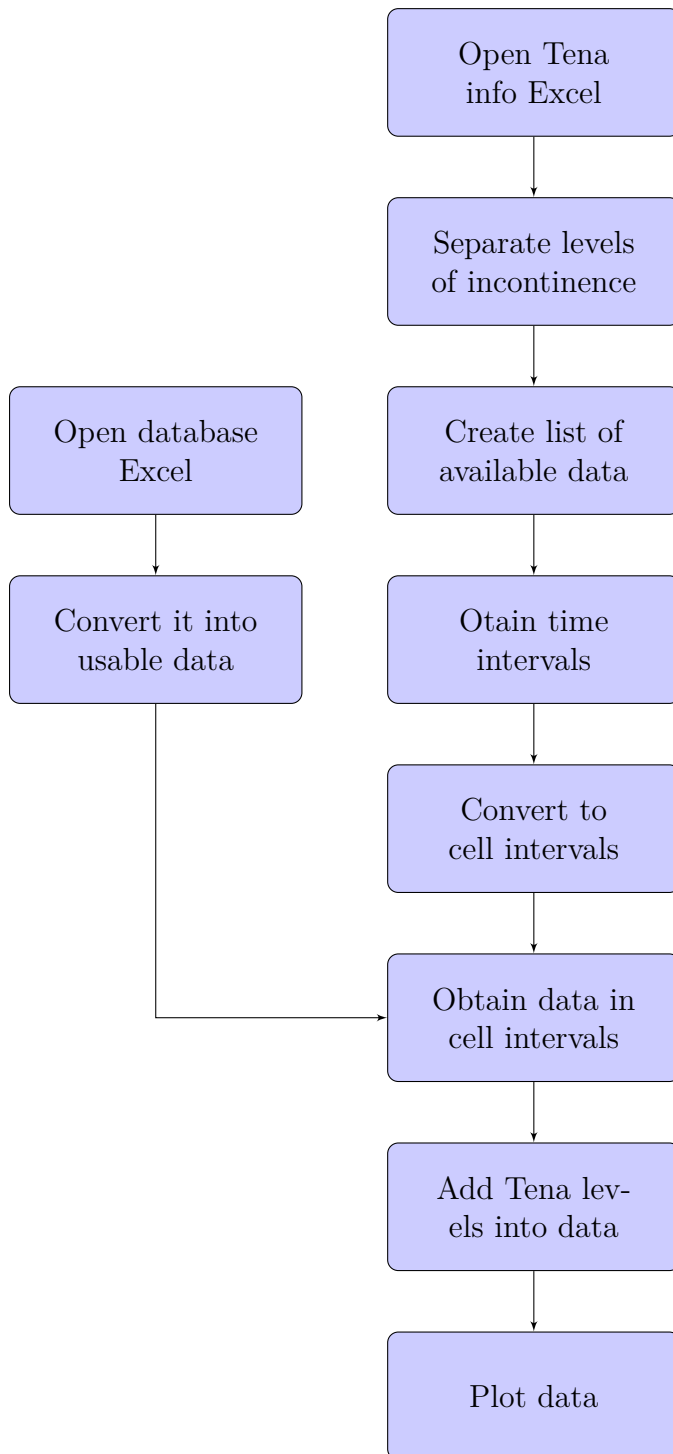


Figure 18: Flow diagram of the data comparison