Smart spaces: aware of users, preferences, behaviours and habits, in a non-invasive approach

Pedro Filipe Fernandes Oliveira

UMinho | 2023

Universidade do Minho
Escola de Engenharia

Pedro Filipe Fernandes Oliveira

Smart spaces: aware of users, preferences, behaviours and habits, in a non-invasive approach

Doctoral Program in Computer Science of the Universities of Minho, Aveiro and Porto

universidade de aveiro

Universidade do Minho

U.PORTO

March, 2023

Universidade do Minho
Escola de Engenharia

Pedro Filipe Fernandes Oliveira

Smart spaces: aware of users, preferences, behaviours and habits, in a non-invasive approach

Doctoral Program in Computer Science of the Universities of Minho, Aveiro and Porto

Universidade do Minho

Work developed under the supervision of:
Professor Doutor Paulo Jorge Freitas de Oliveira Novais
Professor Doutor Paulo Jorge Teixeira Matos

Acknowledgements

Obtaining a Ph.D. degree is normally seen as a solitary process, in which the candidate spends several years studying and specializing on some topic, and focusing on personal achievements. On this PhD not only personal achievements, but also on establishing connecting points and on sharing lessons learned with the peers. Many people had an influence on the work described in this thesis. Following I would like to mention those who had most impact.

First of all I must mention Paulo Novais and Paulo Matos, who more than supervisors became friends. They provided unconditional support giving me the opportunity to carry out my research thesis, for his guidance, many conversations and improvements.

Also, I would like to express my sincere appreciation to the Intelligent Systems research group at "Research Centre ALGORITMI" for the great working environment provided, the productive discussions and overall cooperation.

Last but not least, I want to extend my deepest gratitude to my friends for their friendship, support and patience. Finally, I would like to dedicate a few lines to my family, who has always been supportive and encouraging, giving me the conditions to go further, specially to my parents José and Justina.

This work is also yours,
with all my gratitude,
Pedro Oliveira

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the Universidade do Minho.

Braga, 10 de março de 2023

Assinado por: **PEDRO FILIPE FERNANDES OLIVEIRA**
Num. de Identificação: 13020963
Data: 2023.03.10 18:12:23 +0000

CHAVE MÓVEL

(Pedro Filipe Fernandes Oliveira)

"Stay Hungry.
Stay Foolish!"
(Steven Paul Jobs)

"Sic Parvis Magna." (Sir Francis Drake)

Resumo

Espaços inteligentes: conscientes dos utilizadores, preferências, comportamentos e hábitos, numa abordagem não invasiva

Numa nova era de desenvolvimento tecnológico, em que os utilizadores valorizam cada vez mais a qualidade de vida, seja na perspetiva de conforto, seja na perspetiva de não terem preocupações desnecessárias, impõe-se que as soluções tecnológicas caminhem nesse sentido, e vão de encontro à resolução dos problemas e dificuldades das pessoas.

Os espaços inteligentes têm um papel fundamental como solução para assegurar o conforto do ambiente e de utilização, sem ignorar aspetos igualmente relevantes, como a sustentabilidade ambiental. São a solução por excelência para promover conforto ao nível de cada espaço da habitação ou mesmo ao nível das preferências individuais dos seus utilizadores, de forma totalmente transparente e não invasiva, e considerando aspetos como tipo de espaço (doméstico, profissional ou de lazer), características térmicas do espaço e dos equipamentos de promoção de conforto, tempos de reação, condições externas (temperatura), entre outros.

Esta tese de doutoramento, realizada em contexto empresarial, visa demonstrar a viabilidade de definir uma framework, suportada por uma arquitetura multi-agente, para controlo de variáveis de conforto, como temperatura, luminosidade, e outras, que pode ser suportada por hardware low-cost e é aplicável ao parque edificado. Framework esta que se presta a diferentes implementações, consoante o tipo de espaço, personalização do conforto e privacidade desejada para os utilizadores e, eficiência na gestão dos recursos. A framework dá solução a aspetos como identificação dos utilizadores; gestão de conflitos resultantes de diferente preferências de conforto; segurança dos espaços, equipamentos e utilizadores; integração com serviços em cloud, incluindo partilha de preferências de conforto; entre outros.

A framework foi validada em dois cenários de utilização, relativamente ao conforto e eficiência energética, tendo em ambos casos sido apurados resultados bastante satisfatórios, sendo de realçar o conforto com elevado nível de satisfação por parte dos utilizadores.

Foi assim conseguida a completa especificação de uma framework que se espera que possa vir a ser um referencial para os fabricantes de soluções de conforto.

Palavras-chave: Espaços inteligentes, Gestão de conflitos, Inteligência Ambiental, Multi Agentes

Abstract

Smart spaces: aware of users, preferences, behaviours and habits, in a non-invasive approach

In a new era of technological development, in which users increasingly value quality of life, whether from the perspective of comfort or from the perspective of not having unnecessary concerns, it is imperative that technological solutions move in this direction, and can solve people's problems and difficulties.

Smart spaces play a fundamental role as a solution to ensure the comfort of the environment and its use, without ignoring equally relevant aspects, such as environmental sustainability. They are the excellence solution to promote comfort at the level of each space in the home or even at the level of the individual preferences of its users, in a totally transparent and non-invasive way, and considering aspects such as the type of space (domestic, professional or leisure), thermal characteristics of the space and comfort equipment's, reaction times, external conditions (temperature), among others.

This doctoral thesis, carried out in a business context, aims to demonstrate the feasibility of defining a framework, supported by a multi-agent architecture, to control comfort variables, such as temperature, luminosity, and others, which can be supported by low-cost hardware and is applicable to the existing buildings. This framework lends itself to different implementations, depending on the type of space, comfort customization and desired privacy for users and efficiency in resource management. The framework solves aspects such as user identification; conflict management resulting from different comfort preferences; safety of spaces, equipment and users; integration with cloud services, including sharing comfort preferences; between others.

The framework was validated in two use case scenarios, in terms of comfort and energy efficiency, and in both cases quite satisfactory results were achieved, with comfort being highlighted with a high level of satisfaction from the users.

Thus, the complete specification of a framework was achieved, which is expected to become a benchmark for manufacturers of comfort solutions.

Keywords: Ambient Intelligence, Conflicts-Management, Multi-Agents, Smart-spaces

Contents

List of Figures

List of Tables

List of Listings

Acronyms

6LoWPAN IPv6 over Low power Wireless Personal Area Networks *(p. 32)*

AES Advanced Encryption Standard *(p. 33)*
AI Artificial Intelligence *(pp. 2, 40, 41, 48, 71, 108, 110, 119)*
AmI Ambient Intelligence *(pp. 1–3, 5, 6, 11, 39–42, 45–47, 57, 71, 107, 108)*
API Application Programming Interface *(pp. 36, 75, 94, 95)*
AUPBH Aware of users, preferences, behaviours and habits *(p. 70)*

BDI Belief-Desire-Intention *(p. 71)*
BISITE Grupo de investigación en Bioinformática, Sistemas Informáticos Inteligentes y Tecnología Educativa *(p. 107)*
BLE Bluetooth Low Energy *(pp. 24–27, 29, 30, 32, 34–36, 57, 66, 68, 81, 85, 86)*

CAGR Compound Annual Growth Rate *(p. 17)*
CBR Case-based reasoning *(p. 75)*
CeDRI Research Centre in Digitalization and Intelligent Robotics *(p. 107)*
CHIP Connected Home over IP *(p. 36)*
CO Carbon monoxide *(pp. 43, 67, 109)*
CO2 Carbon dioxide *(pp. 43, 46, 67, 109)*
CORBA Common Object Request Broker Architecture *(pp. 44, 46)*
CS Computer Science *(p. 7)*
CSA Connectivity Standards Alliance *(p. 36)*

DSR Design Science Research *(p. 7)*

ECG Electrocardiogram *(pp. 21, 22)*

ECMA Standards organization for information and communication systems *(p. 27)*

ED End Device *(p. 32)*

EEG Electroencephalography *(pp. 21, 22)*

EMG Electromyography *(pp. 21, 22)*

GATT Generic Attributes *(p. 25)*

GDPR General Data Protection Regulation *(p. 79)*

GPS Global Positioning System *(pp. 24, 37, 66, 67, 82, 85)*

HCI Human Computer Interface *(p. 41)*

HTTPS Hyper Text Transfer Protocol Secure *(p. 85)*

HVAC Heating, Ventilating and Air Conditioning *(p. 4)*

IBSG Internet Business Solutions Group *(pp. 11, 12, 17)*

ICT Information and Communications Technology *(p. 38)*

IE Intelligent Environments *(pp. 1, 2, 53)*

IFTTT If This, Then That *(p. 33)*

iOS iPhone OS *(p. 24)*

IoT Internet of Things *(pp. 1, 3, 11–15, 17–19, 25, 26, 30, 31, 35, 36, 39, 45, 51, 80, 81, 83, 108)*

IPB Polytechnic Institute of Bragança *(pp. 107, 112)*

IPv6 Internet Protocol version 6 *(pp. 31, 32)*

IS Information Systems *(pp. 7, 13)*

ISL Incremental Synchronous Learning *(p. 43)*

Islab Intelligent Systems Labs *(p. 2)*

ISM Industrial, Scientific and Medical Radio Bands *(pp. 25, 30)*

ISTAG Information Society Technologies advisory group *(p. 40)*

JSP Java Server Pages *(pp. 44, 46)*

MAC Medium Access Control Layer *(p. 33)*

MAS Multi Agent System *(pp. 2, 7, 8, 51, 53, 59, 64, 70–76, 86, 87, 90, 96, 108, 109, 118)*

MIT Massachusetts Institute of Technology *(p. 11)*

NFC Near Field Communication *(pp. 24, 26–29, 66, 68, 81, 85, 86)*

OTA Over the Air *(p. 35)*

PDA Personal Digital Assistants *(pp. 20, 21, 43)*
PHY Physical Layer *(p. 33)*
PIN Personal Identification Number *(pp. 28, 29)*
POS Point of Sale *(p. 24)*
PPG Photoplethysmogram *(pp. 21, 22)*
PRS Procedural Reasoning System *(p. 71)*

RFID Radio-Frequency Identification *(pp. 11, 27, 37, 44, 46)*
ROI Return on Investment *(pp. 13, 14)*

SDK Software Development Kit *(p. 36)*
SHA Secure hash algorithm *(p. 85)*
SIG Special Interest Group *(pp. 25, 26)*
SLR Systematic Literature Review *(p. 8)*
SMS Short Message Service *(p. 37)*
SpO2 Peripheral Oxygen Saturation *(pp. 21, 22)*

TLS Transport Layer Security *(p. 85)*

UM University of Minho *(pp. 107, 112)*
UN United Nations *(p. 39)*
USAL University of Salamanca *(p. 107)*
UUID Universally Unique Identifier *(pp. 68, 81, 82, 84, 85, 109)*

VXML VoiceXML *(pp. 44, 46)*

Wi-Fi IEEE 802.11 *(pp. 24, 28–30, 32, 34, 36, 37, 68, 85, 86)*
WIFI Wireless *(pp. 24, 27, 28, 30, 35, 66, 81, 95)*
WPA2 Wi-Fi Protected Access 2 *(p. 29)*
WPAN Wireless Personal Area Network *(p. 31)*
WPS WIFI Protected Setup *(p. 29)*
WWDC Worldwide Developers Conference *(p. 35)*

1 Introduction

This chapter starts with a subject overview, identifies the context and motivation, defines the problem, formulates the research hypothesis, the goals to be achieved are enumerated, and concludes with a document organization brief.

1.1 Overview

We are on a new era of interaction between people and physical spaces. Users want that those spaces smartly adapt to their preferences in a transparent way. Considering this, managing comfort preference's conflicts of the different users and spaces on an IoT adaptive system is an actual problem, and with several research already done.

As shown in [3] there are new opportunities for research in the field of smart environments that should be explored. In particular the concepts of smart homes and home automation [1], currently in growing expansion in the scientific and research point of view, as the market demands for better solutions in this field. The aim is to take advantage of emerging technologies available in the market that support the denominated wearable devices [50] [54], and the non-invasive particularity of these to, in an autonomous way, adapt the environment to the comfort preferences of each user (e.g. thermal, acoustic, air quality, light, sun exposure). Provide comfort according to the preferences of each individual, is a challenge and an opportunity to create innovative solutions and new paradigms in the context of Intelligent Environments (IE) [13] [108].

User behaviour analysis in smart environments is performed mainly using data collected from the sensors dispersed throughout the environment [15] [17]. As concluded in [16], the perfect learning system for smart environments has not yet been found, and each new valuable contribution in this field puts us one step closer to the true concept of intelligent environments [38], more than ten years later, this statement is still current. It is also referred the need and the challenge of establishing a new effective paradigm for AmI, where the focus becomes the user and the ability to manage the complexity and richness of everyday human life [108] [16]. A recurring challenge in this field is the conflicts of interest management among the several users at the space [59] [116], that in this work was addressed through the use of MAS, as

well user information and real-time data acquisition from different sensors [32] [73]. It will be possible, making use of technologies and emerging wearable devices available on the market (e.g. smartwatches, smartphones, fitness trackers) [59], to focus the data collection process on the user, always considering that it will be a non-invasive process. This will significantly leverage/enrich the decision-making process and overtake the physical limits so far imposed by the need of sensors statically placed in a space [37] [20]. After analyzing the state of the art, it can be pointed out the scientific innovation and contribution that this thesis can bring to this field: conceive and design new solutions and paradigms that contribute to turn into a reality the concept of IE. Besides the scientific contributions, this work produced relevant results with industrial application since it has been carried out jointly with an industrial partner. Prior to this proposal, this partner conducted several market and search studies. Concluding that the international market does not offer solutions with such intelligence level, user-oriented or based on non-intrusive techniques of data acquisition. There are products with some features implemented, but which always need the programming and configuration of the product by the user, are oriented to the spaces and not to the individual or do not have the capability of self adjustment using Artificial Intelligence (AI) to enable predictive capabilities and improvements in the product effectiveness in the decision-making process.

Also at the industrial market, the big players like *Amazon*, *Google*, *Tado*, *Honeywell*, and others doesn't have the kind of features proposed by this thesis.

1.2 Context and Motivation

This Ph.D thesis has joint supervision of the Intelligent Systems Labs (Islab) , a research group integrated in the *ALGORITMI Research Centre*, coordinated by the supervisor Paulo Novais, and Polytechnic Institute of Bragança, represented by the supervisor Paulo Matos, and it is integrated in the applied research activities and technological development of *Techwelf, Lda.*, a company dedicated to Intelligent Environment solutions design, also represented by Paulo Matos.

Thus, it is intended that this work, in addition to the scientific component, also has practical and applicable results by the company.

There are few industrial applications in real context of this research, at the AmI field. It is intended that the resulting investigation of this thesis is implemented by the company involved, in the development of solutions for smart environments that it intends to place on the market.

Having this thesis an element of industrial application, is important to consider the economic relevance of this market type, particularly in terms of size, value and growth potential, as stated by the largest global consulting firms forecasts (Gartner [6], McKinsey [87] and Business Insider [94]).

As stated in section 2.2, a significant number of technological changes, as well new technologies introduction, has enabled the appearance and IoT evolution. This combined with the fact that such devices prices have come down drastically, placing sensors, processing power, bandwidth and more storage, available to more users and allowing a larger number of IoT appliances. These numbers are transversal to

several application fields, and the common factor is mainly its exponential growth expected in the coming years.

There is also the personal motivation to deal with the challenges identified in the AmI field referred in section 1.1, and presented in detail later at section 2.3.1. Particularly in what concerns to privacy, security, and management of multiple users in the same environment.

1.3 Problem definition

The problem/challenge, need or, on the economic perspective, opportunity, which aims to overcome, can be defined in general terms as a contribute to create intelligent environments capable of adapting to the users comfort needs/preferences, in an automatic, transparent and non-invasive way, whether these environments are for domestic, professional or public use. In addition, and although the focus is always on people, it is always important yo enhance the optimization of other aspects, such as the resources efficient management, the ecological footprint reduction, and others.

This challenge relatively to users, currently has as main difficulty the people mobility and the disparity of habits, schedules and every individual comfort preferences. The same is aggravated by physiological conditions, derived from a large number of factors (tiredness, mood, etc.), impacting the user preferences, in such way that current systems cannot measure.

Contextualize user preferences, is a process involving many variables and different dimensions, which makes this a high complexity problem.

In addition to the physiological conditions mentioned above, there are two critical and essential dimensions, these are the space (user location) and the time (day period). The spatial dimension introduces challenges associated with the user versus space relationship, whether in the generic concept of space or by type of space: personal, professional, public/social or other. Contextualize the user location is essential to optimize the comfort conditions and contribute to the performance and solution effectiveness.

The time dimension is equally critical, because the comfort preferences will change over the day course, as well the week or even the year. For example, the comfort preferences may be different between daytime and night-time, or between weekday and weekend/holidays. Namely it is know that the temperature in the human comfort context depends of different physiologically conditions, like mood, stress, anxiety, as well the physical conditions of each user.

In this dimension, it is also important to assess the changes over the year, that will have the seasons influence, which naturally also change user's comfort preferences, because it has to be considered not only the effective temperature, but also the thermal sensation, that naturally it will also have influence.

In this paradigm, which is intended to be the comfort superlative, there are at least these three dimensions: time, space and user comfort preferences.

The time and space dimensions are critical to contextualize the user's personal preferences, and provide the necessary information that will allow assess future preferences in a useful time.

It should be noted that Heating, Ventilating and Air Conditioning (HVAC) systems that promote comfort have different operation latencies (inertia). Early forecast of user presence and respective comfort preferences allows to surpass this latency, namely starting its functioning in due time, to achieve the desired comfort conditions. The same solution also allows to optimize energy savings associated with such equipment, since it allows to forecast when the systems may be disabled or placed to operate more efficiently. This savings can be defined in several aspects such as energy, maintenance, useful life period, etc.

And these are the main problem issues, of course there are accessory issues, such as the management of more than one user in the same space, among others that naturally arised and had to be addressed.

Figure 1 shows the scenario of an environment around which the present work was developed. Explaining this figure, it can be seen the user, that do the necessary input information to the system, namely the parameters that different actuators need, to do its function. Next, the local system must perform the synchronization and information share. Following, the system must perform the different environment components (climatization systems, security systems, other smart systems) management.

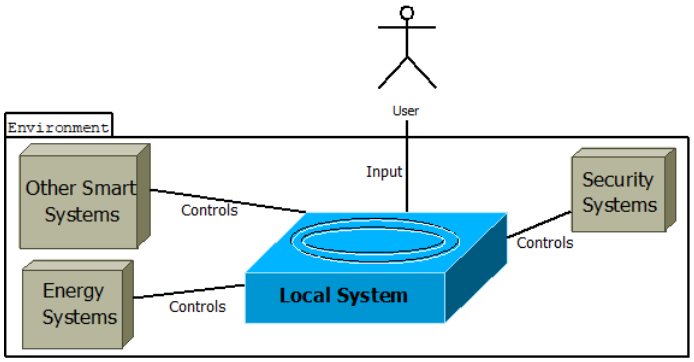


Figure 1: Overall scenario.

This scenario represents an environment at a given location and at a given point in time. As mentioned above, for a proper solution that fits all users, it is necessary to consider the user mobility. This implies considering the time and space dimensions [99].

As such, for a better understanding of the overall scenario in the user's daily life, figure 2 contextualizes the temporal and space dimensions present in this problem and already mentioned above. We can see that different user locations, combined with time context, naturally results in an environment with different characteristics. This kind of global scenario is also addressed in this thesis [100].

Like is represented at Figure 2, user moves between different spaces during the time. Also for a better understanding, of this problem aspect, at Figure 3 is represented the movement of two users, and we can see how different the schedules and habits of each one are.

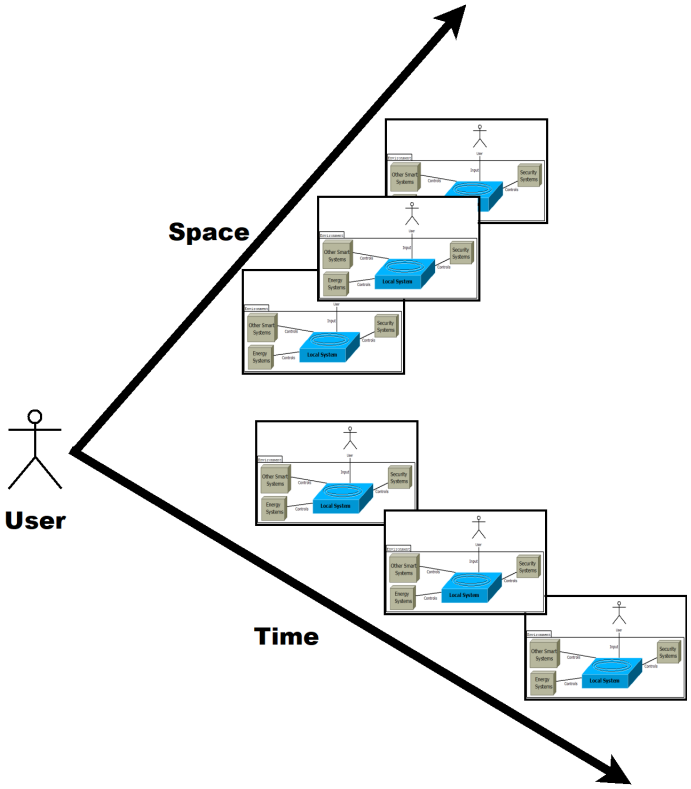


Figure 2: Contextualization of Time/Space Dimensions.

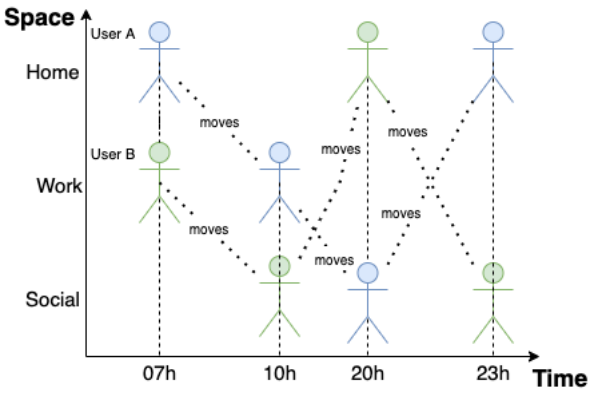


Figure 3: User movement - Time/Space Dimensions.

1.4 Research hypothesis

Considering the AmI field and this Ph.D. thesis scope, it is possible to evidence that there are different issues to be addressed. The comfort preference's management currently have different obstacles as well as a diverse number of specificities. Thus, the following questions elucidation it is intended to approach the research problem:

- How can we characterize an environment ?
- How human comfort is defined ?

- What are the human preference's set ?
- Can we automatically detect user preference's ?
- Is it possible to solve/minimize user comfort preference´s conflicts ?
- Can human comfort be measured ?

Considering all this questions, they have been materialized and define the proposed research hypothesis as: *A Smart space as a transparent, non-intrusive and safe solution to guarantee the satisfaction and comfort of users according to their preferences and routines.*

1.5 Objectives

Considering the problem definition, and the research hypothesis already detailed, the aim of this thesis is to create an architecture that takes advantage of emerging technologies that support wearable devices (e.g. smartwatches, smartphones, fitness trackers) and the non-invasive characteristic of these, for collecting data in an autonomous and transparent way without user intervention. And with that information, assist the comfort system decision-making processes to adapt the environment to suit each user's comfort preferences (e.g. thermal, acoustic, air quality, lighting, sun exposure).

Specifically this thesis aims to achieve the following goals:

- Characterize different types of environments (AmI);
- Characterize comfort in its different aspects and dimensions;
- Set a base architecture for a non-invasive system that takes advantage of different technologies (smartwatches, smartphones, fitness trackers), that facilitate user's interaction with existing systems [22];
- Represent different stakeholders, contexts and problem dimensions, who cooperate to achieve the optimal solution [21];
- Manage the possible conflicts of interest, namely between users at the same space [90];
- Develop solutions that allow ubiquity in users identification and their comfort preferences, in an automatic and transparent way, enhancing the integration between space, time and user;
- Apply the proposed architecture at a health facility and/or a higher education institution, taking advantage of the existing company's partnerships;
- Evaluate the architecture using real/simulated conflict management problems, between distinct users that share the same space.

1.6 Research methodology

Even with some literature works discussing if Computer Science (CS) is really a science or an engineering, according to Hassani [63], researches on this field can have theoretical or experimental nature or even both of them. Besides that, just like in other study fields, CS researches may use quantitative or qualitative methods.

To ensure that research is validated as reliable and relevant, both academically and in society, it must demonstrate rigorous development and be susceptible to discussion and confirmation. This is why a robust research method is imperative to the success of any study.

Design Science Research (DSR) is a theory that investigates the knowledge generation in the designing artifacts process, that is, how design methods can constitute research of a scientific nature [66].

It has its origins in engineering and the sciences of the artificial, is based on a problem-solving paradigm and aims to create innovations that define the ideas, practices, technical capabilities and products through which the analysis, design, implementation, management and use of information systems can be effectively and efficiently realized [124] [65] [104].

The DSR aims to raise research performance in Information Systems (IS) through a concise and conceptual framework, execute and evaluate the research.

This methodology was selected for adoption to solve this problem, as it was understood to be the most appropriate for the objectives to be achieved in order to solve the proposed problem, and to answer the research hypothesis indicated at section 1.4.

The DSR methodology will be applied in the development of this thesis, in order to develop artifacts to solve the proposed problem, namely the full architecture necessary to support the resolution, as well as the MAS development.

This artifacts will depend on the different objectives analysis identified at section 1.5. With this artifact development, it is understood that the proposed objectives are achieved and a solution is given to the presented problem.

Review Protocol

To start the review protocol development, the search string was defined through several iterations to find the most relevant articles related to the topic. Therefore, the search string used was as follows:

("Ambient Intelligence" OR "AmI" OR "smart-spaces") AND ("comfort" OR "comfort preferences") AND ("Multi-Agents" OR "MAS" OR "agents").

For applying the search string, the following data sources were defined: *IEEE Digital Library*, *ACM Digital Library*, *Google Scholar*, *Scopus* and *Springer Link*.

After the inclusion and exclusion criteria are defined, to filter the obtained papers set. The used criteria are show at table 1.

With the different criteria defined, the online tool *Parsif.al* [1] was then used to continue the Systematic Literature Review (SLR).

[1] https://parsif.al/

Inclusion	Exclusion
Scientific papers	Only after 2010 Partial documents Full technical No access Less than 5 citations

Table 1: Review Protocol criteria

Using the search string, 4880 articles were initially found, and after applying the different criteria, the results went down to 820 articles. After using the final criterion of more than 5 citations, it was reduced to **52 articles**.

The protocol review is defined as follow:

1. Search at Data Sources using the search string defined before;
2. Results filter using inclusion and exclusion criteria;
3. Read Abstracts and Conclusions;
4. Read Full Papers;
5. Definition of final papers set.

1.7 Thesis organization

This document is organized into five main chapters. In chapter 1, an introduction is performed, and further the problem definition, the objectives and is presented the research hypothesis and methodology that motivates this doctoral thesis development. Chapter 2 describes the state of the art at field, both in terms of scientific practices, such as emerging technologies, as well its economic viability. Chapter 3 describes the developed work and the approach proposed for the implementation, since the general requirements, architecture, detailed description, and other developed work such the user behaviour simulation, MAS and system data privacy.

At chapter 4 are defined and characterized the usage scenarios, the evaluation methodology, and a result analysis is also performed.

This thesis is concluded at chapter 5, with some summary and final considerations, the conclusions discussion, hypothesis validation, contribution of this work, the limitations and some future research directions are also pointed out.

2

Smart Spaces

This chapter start with a state of the art review, continues with the support concepts and technologies detail, do a extensive detail of AmI field and finalizes with the user in a smart space concept specification.

2.1 State of the Art

This section presents a literature survey in the areas related to the problem under study. It begins with the approach to the IoT topic, moving to the devices analysis, communication technologies and platforms, that are relevant for this thesis. Ending with the survey, description and review of the more relevant projects in the field of smart environments, and in this thesis context.

2.2 Support Concepts and Technologies

It's already been pointed out, as the next industrial revolution that will change the way businesses, governments and consumers interact with the physical world. This is the Internet of Things, commonly called IoT. There are already studies that analyze the growth and the IoT ecosystem, which allows entities (consumers, businesses and governments) connect and control their IoT devices in different scenarios.

The idea of IoT has emerged more than a decade in the Massachusetts Institute of Technology (MIT) laboratories [85]. Initially, the use of IoT was associated with Radio-Frequency Identification (RFID) technology and its application in tracking products in supply chains [54].

But the IoT has evolved from the approach based on RFID technology, for a vision that encompasses the IoT as a society where a diversity of different objects are connected ubiquitously [54]. The Internet Business Solutions Group (IBSG) at *Cisco*, reinforces this view by claiming that the IoT is the moment that more things than people are connected to the Internet. This connection is achieved by the incorporation of small sensors in a wide variety of objects, which enables that these objects with power processing and unique identifier may be connected to the Internet [120]. Going further, the IoT, not only lets things communicate with each other, but also creates new ways for people to communicate with things [20].

Basically, IoT consists in the override and interconnection between the physical and the virtual world [85], by creating wireless networks of objects, where things or objects have a unique ID, are connected to the Internet and have its own processing power that allows them to sense and respond to changes in their environment. According to [85], the IoT is leading the third wave of revolution in the industry of information technology and is already indicated by developed countries, as one of the most important strategic pillars for the promotion of economic development and technological innovation. The IBSG at *Cisco* argues that in 2010 there were already 12.5 billion things connected to the Internet, and at that time, this number had an expectation of grow to 50 billion by 2020 as shown in figure 4.

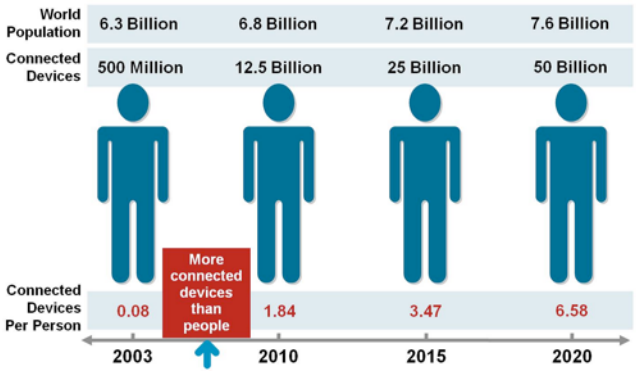


Figure 4: The IoT evolution [49].

And detailed by category at figure 5.

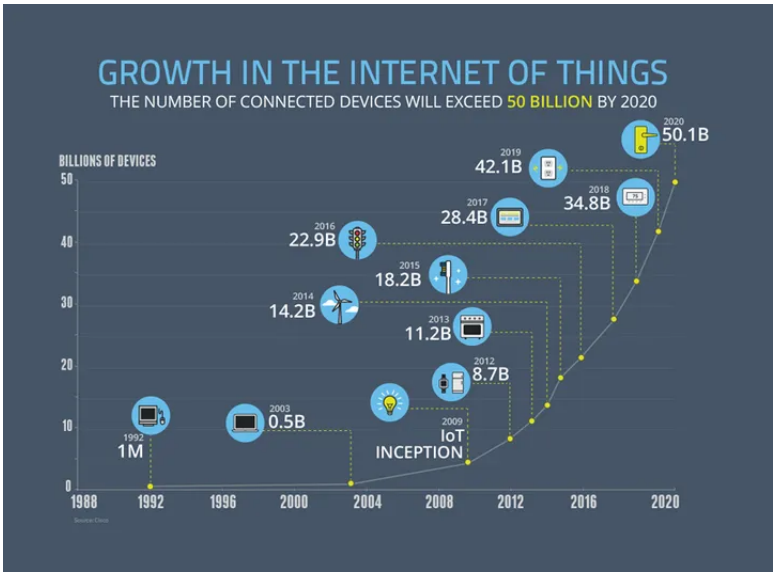


Figure 5: The IoT evolution by category [49].

In a network of IoT where objects communicate with other objects and people, about the changes in their environment, the amount of information generated is immense and available in real time [20], which means that the IoT has the potential to have an impact in most of the value chains [50], and even promoting the appearance of new business models [37].

The figure 6, shows the typical IoT ecosystem, the behavior of this can be seen as follows: the entity uses a remote peripheral (smartphone, tablet, etc.) to send a command or place an order to a IoT device via the network. Then the device performs the command and/or send information back through the network to be displayed on the remote device.

There are other scenarios where the IoT information generated by the device can be analyzed and stored in other locations, including the Cloud, a local database, on the mobile device, or even in the actual IoT device. There is also the gateway concept, that in this ecosystem has a role in bridging the gap between traditional network and IoT devices, thus allowing communication between IoT devices and traditional network or Internet.

According to the report from *Business Insider* [72], in 2020 were expected a total of 34 billion devices connected to the Internet, compared to the existing 10 billion in 2015. With a expectation to spent 6 trillion dollars in IoT solutions from 2015 to 2020.

Companies, to optimize their business, will be the largest user of IoT solutions. Are identified four main forms of the IoT improve business: reduced operating costs, increased productivity, expansion into new markets and development of new product lines.

On the other hand, governments put their focus on increased productivity, reduced costs and increased quality of life. This type of entity is identified as being the second largest in adherence to the IoT ecosystems [72].

Consumers are behind the companies and governments in support of this type of products. However they are also responsible for a massive amount of purchases.

The latest report from *McKinsey&Company* [87], referring to the IoT theme is also unequivocal about the exponential growth of the market, as well the diversity of new business opportunities that this entails. In this thesis several examples of applicability in different business areas are identified. Namely in retail stores of several fields, have the customer's location information in-store, allows customization of promotions in real time, based not only on location, but also in customer purchase history. Watch the movement and behaviour of customers, can also be used to optimize the content layout at stores. The inventory optimization process and even the payment will become completely transparent, being made automatically when the customer exits the store physical space.

For homes application, are indicated in the security field, the use of cameras and sensors, enabling the prediction and user alerts. As well applications and devices which make autonomous housing, learning the users habits, and identifying the best time to perform the tasks autonomously. Also at energy management level, by using autonomous sensors and thermostats taking advantage of user's habits knowledge, can allow a more efficient and self-adjusting solution.

IoT is currently seen as the revolution of this decade, in the IS field. Mainly by the numbers magnitude indicated by consultants, both in terms of installed devices, or in terms of predicted Return on Investment (ROI). The numbers presented here are the most current, regarding the forecasts for the time period (2015-2025).

At figure 7, provided by the *Business Insider* [72], we can identify the values for three levels of entities

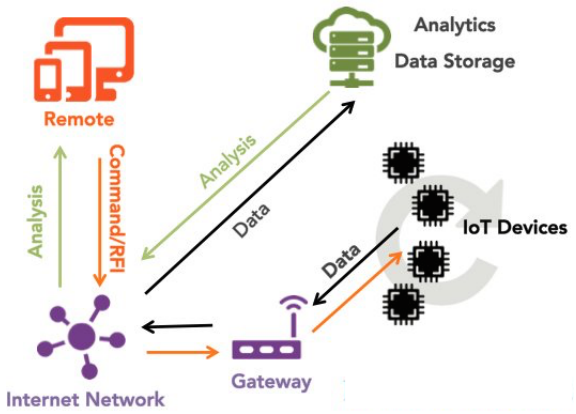


Figure 6: IoT Ecosystem [72].

(consumers, governments and business). And it can be seen as a curiosity, the projected total of 23.9 billion devices installed for 2020 and 12.7 trillion in ROI for the period (2015-2025).

Given this high value, the world's leading companies, have given emphasis to this growth, further fueling this market. Because this kind of technology has created an endless number of new business opportunities, as well improvement of existing businesses. And traditionally, large global companies seek to come first, to establish themselves as leaders in different sectors, as already happened in other technological revolutions.

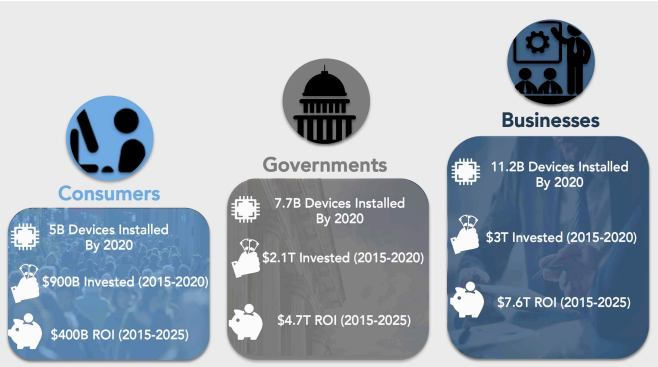


Figure 7: IoT number of devices installed by Entity [72].

2.2.1 IoT Appliance level

Forecasts continue to suggest that the wearables market, including smartwatches and fitness bands continue to grow. However, some questions remain about where and how these devices will be used and if they are viable at users daily life. And even if they will be mass adopted, and can have a real impact on consumers lives.

At one hand, there are the defenders that point its potential as comparable to smartphones and tablets, which opened a new computing era. On the other, there are skeptical that see much more limited opportunities in this field.

It is, however, the health sector, the most prominent field for wearable devices adoption [107]. Because there are already a number of emerging trends in consumer and healthcare at a professional level, which have been complemented by advances in technology in this field over the past few years. And where wearables are currently more used for fitness purposes, and thus show great potential for widespread adoption in this sector, for other purposes than fitness.

Technological giants, including *Apple*, *Google* and *Samsung*, have not neglected this market and are developing devices and platforms that will help bridge the gap between common user-fitness data storage and use of medical care in a real context, and so there are significant improvements to the user. This concept has also been revealed special interest to insurance companies.

Following are shown several charts with the projections previously mentioned. In addition to these numbers, which reflect market values really huge, which makes this a truly attractive market for any company.

Another type of business opportunity that IoT will achieve, are new business models, for instance, remote monitoring capability allows that a large number of activities can now be seen as service. As well, as existing business processes transformation, particularly in predictive maintenance and better assets use, which can increase productivity. Figure 8 illustrates the global forecast sales of such devices between 2010 and 2020 period.

Most of these numbers are estimates, because it takes some time to definitively close this numbers and also to be published by the different consultants.

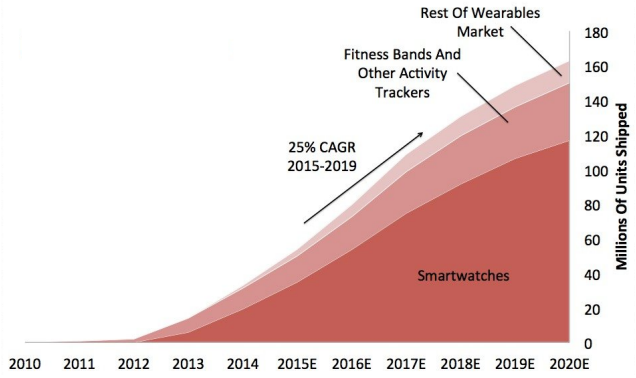


Figure 8: Global Wearable Device Unit Shipments Forecast [72].

Home Appliance

House connected devices include all smart appliances (washing machines and tumble dryers, fridges, TV's, etc.), security systems (Smart Home Security), (sensors connected to the Internet, monitors, cameras and alarm systems) and energy equipment (Smart Home Energy), (thermostats and smart lighting).

Nowadays, current smart home market state had been increased to the mass market since the stagnant period at 2015, this can be interpreted, after technology adoption curve analysis that can be seen in figure 9. This period it was the market transition between early adopter phase and the mass market

adoption. For this transition happen, manufacturers need to prove the necessity and importance of use for this type of devices. There are several identified barriers that are preventing the transition to mass adoption of smart homes market: the devices high price, as well long replacement cycles.

Typically, mass-market consumers, wait until his device is faulty to replace it. Then they will compare a connection-less product and a device with connection to see if benefits are worth the differential pricing. However, the biggest barrier is the technological fragmentation within smart homes ecosystem. Currently there are many types of networks, standards and devices for the smart home connection process. Creating obvious problems of interoperability and making it confusing for consumers to configure and control several smart devices solutions. Because they require multiple network devices (gateways) and applications for creating and maintaining his smart home.

Intelligent household devices are however becoming more common worldwide. As presented at [72], a smart home device can be set up, such as any single object that is in the house and is connected to the Internet, can be monitored or controlled remotely, and has a non-computational primary function. Several devices of this type inserted in the same house form the basis for a smart home ecosystem.

Until interoperability issue is completely resolved, users will have difficulty in choosing devices and systems to achieve a full smart home solution. The short-term solution, found to overcome this technological fragmentation, are the so called "closed ecosystem". These are usually composed of devices that are compatible with each other and can be controlled through a single point, this is normally achieved using only products from the same manufacturer that has the full range of products required for this purpose.

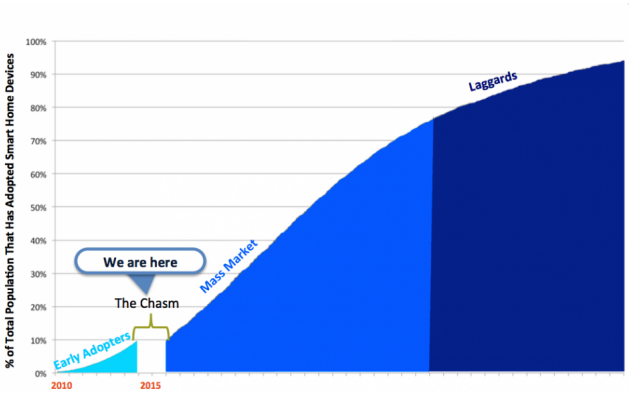


Figure 9: Smart Home adoption curve [72].

However, a large number of consumers still do not fully understand what devices connected to the home are and the operation of these. And this has led to the demand has not yet reached its full potential, there is already a high awareness and adoption, taking into account that this is a new category market. The houses around world, will become smarter and more connected during the coming years. It is expected that the devices become more present in the next two years, when it is expected that the growth reaches its peak for the analyzed period (2015-2025).

Indicators point to that the sales of devices connected to home will have a Compound Annual Growth Rate (CAGR) of 67% over 2015 to 2019, thus reaching a much higher growth than smartphones or tablets,

and thus reaches the value of 1.8 billion units sold in 2019, according to estimation from *Business Insider* consultant [72].

Figure 10, illustrates the global forecasting of sales by type of devices, related to smart homes between the period of (2012-2019).

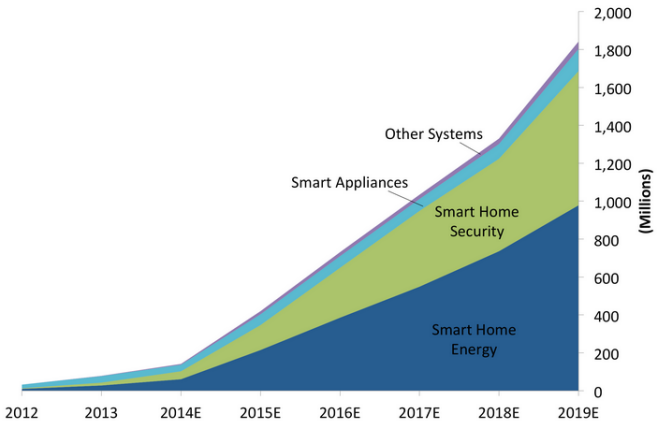


Figure 10: Global Connected-Home device shipments by device category [72].

This category of home connected devices represent about 25% of sales in IoT 2016 market. This representation gradually increase to about 27% in 2019, when the growth in other areas of IoT be intensified.

Translated into numbers, the sales revenue of home devices connected have exceed 61 billion dollars for 2015. And this number rises to a CAGR of 52% reaching the 490 billion dollars value in 2019.

The energy management equipment's (Smart Home Energy) and safety systems (Smart Home Security), including devices such as smart thermostats, smoke detectors and alarms will become popular initially, opening the way to a wider subsequent adoption of products by consumers. With the latest *Business Insider* report publication [94], which reinforces the huge growth already foreseen in previous reports.

Figure 11 shows the most important factors in an IoT platform. As can be seen, the most relevant factor is by far the security/safety and followed by the easy implementation and integration fact.

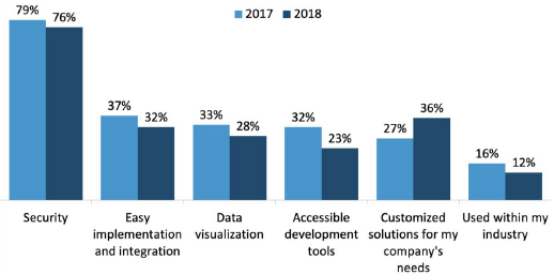


Figure 11: Most important factors and capabilities in an IoT platform [94].

Then in figures 12 we can see the number of installed devices, pointing to a little more than 20 billions in 2020, a quite below than the initial previsions of *Cisco* IBSG group, with the 50 billions installed devices

now estimated to the year 2024.

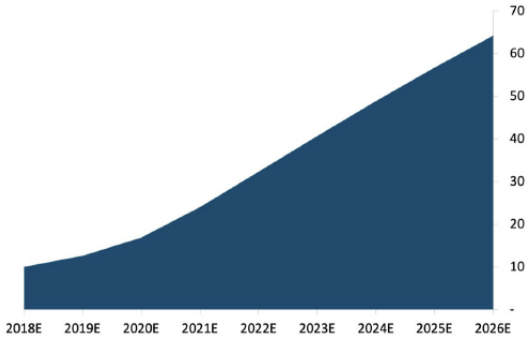


Figure 12: Forecast: Global IoT device installation base (Billions) [94].

Related to the installed devices annual increase, we can analyze the figure 13, and see that the peak is expected to be at 2022 and 2023 with 8 billions of IoT devices installation, and this will slowly decreasing to close to 7.5 billions of IoT devices at 2026.

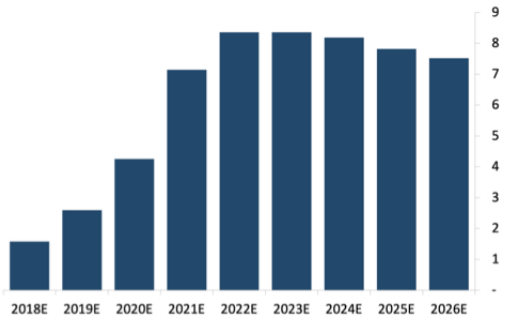


Figure 13: Forecast: Total IoT device annual installations (Billions) [94].

That's because the number of devices growth will be fueled by falling device costs on both an upfront and continuing basis.

Overall, the forecast at figure 14 points that, companies and consumers will spend almost 15 trillion of dollars on IoT devices, solutions, and supporting systems from 2018 through 2026, with the annual investment surpassing 1 trillion of dollars in 2021, and 1.5 trillion of dollars in 2022.

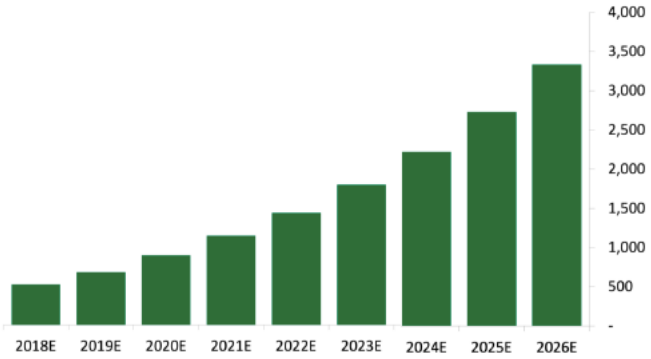


Figure 14: Forecast: Global IoT Investment (Billions ($)) [94].

Analyzing IoT solutions types that companies are using in figure 15, we highlight above all remote monitoring, which is utilized by 62% of respondent. Remote monitoring devices provide a wealth of data on assets and equipment that can be leveraged to follow utilization and trends, ensure proper procedures are being followed, and enable large-scale analysis to increase efficiency and engage in useful practices like predictive maintenance.

And similarly, 43% of respondents say their companies use IoT devices for asset tracking. Many firms use IoT devices to gain visibility throughout their supply chains or to enable better tracking of shipped goods. These simple IoT devices can provide critical data to streamline operations and identify potential issues or bottlenecks.

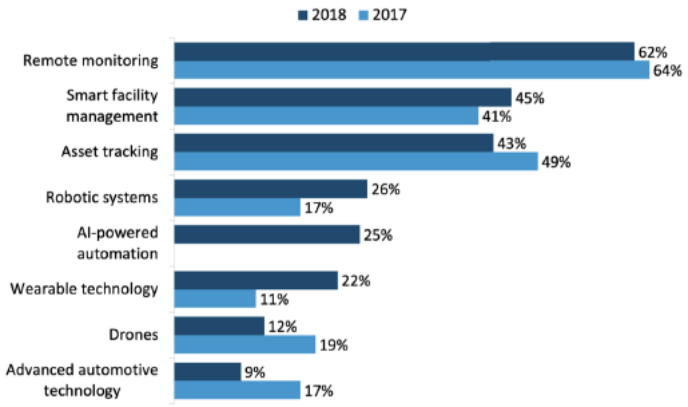


Figure 15: Types of IoT solutions companies are using [94].

Regarding the smart cities topic, in figures 16, investment in this field can be analyzed. In 2018, two primary types of smart city solution providers emerged telecoms and tech giants, and they promise to define the market moving forward.

The main types of companies involved in working with cities and public-private partnerships to implement smart city programs are telecoms, and technology giants. So far, telecoms have been far more successful in putting solutions into place, and generating results [94].

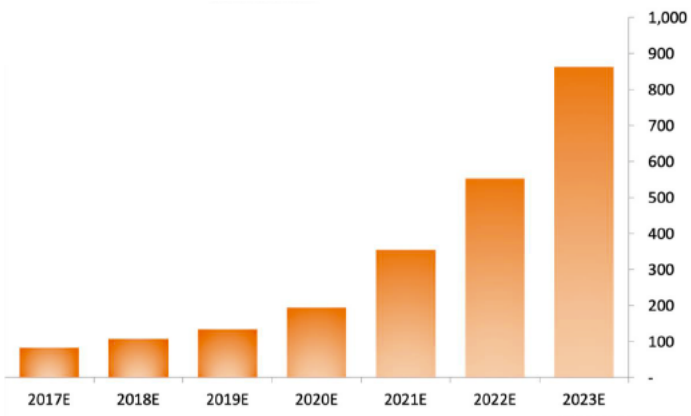


Figure 16: Annual smart city investment (Billions ($)) [94].

Also at figure 17 the information generated in billions of terabytes can be analyzed, with a estimation that smart city systems will generate nearly 180 billion terabytes of data each year by 2023.

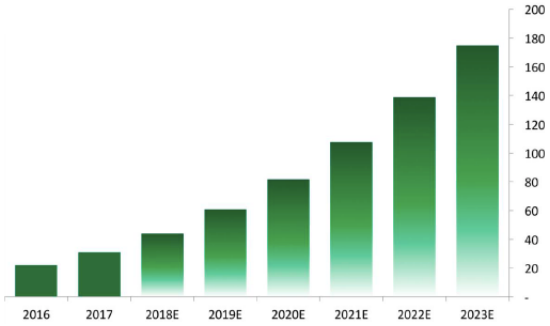


Figure 17: Annual smart city data generated (Billions of terabytes) [94].

2.2.2 Device Types

Following the different device types that can be accommodated by this thesis, are fully detailed, and explained his viability to the use on this thesis.

Smartphones

The ongoing technological revolution has also contributed to the mobile devices massification with more features, storage capacity and data processing. Currently these devices are called "smartphones", derived from the characteristics indicated above.

Despite its massification, it was not long ago that the technology arrived to the general public. With its advanced computing capabilities and other features, smartphones have quickly gained popularity. Before the smartphones invention, there were several devices that were used, including traditional mobile phones, and Personal Digital Assistants (PDA) devices.

Continuously, the technology was combined and the concept of smartphone was born. The first prototype comes in 1992 with the name of *Simon*, by *IBM* [110] having features such as PDA and fax, touch screen and other more advanced.

Ten years later, it was time for the *Kyocera* company launched on the market the *QCP 6035* model, the first to combine phone and computer functions. The operating system was the *Palm OS* and had internet connection [35].

Second part of nineteen decade, many mobile phone users have started using PDA. Initial PDA performed different systems such as *Blackberry OS* and *Palm OS*. Nokia launched a phone combined with a PDA in 1996. The device was called *Nokia 9000* [110].

This manufacturer, turned out to spread the smartphones use worldwide. However, the real revolution came in 2007, with the *Apple's* smartphone introduction and, later in 2008, the *Android* system smartphones introduction by *Google*. Currently these two systems represent 96.3% of the global smartphone market [71].

Related to that fact, this thesis is intended to be compatible with those two platforms, and thus achieve a high potential users market share. These devices have mostly one or more communication technologies, identified in the section 2.2.3, allowing full compatibility with the proposed system.

As shown in figure 1, the smartphone will be one of the possibilities which user can interact with the system as input device. Other kind of possibilities provided are demonstrated below.

Wearables

As noted above, wearables are still in mass adoption process, however, as a less invasive device that the phone, and assumed that always be in user possession. It is also the type of device most used in this thesis context.

This will allow the user mobility, without using invasive processes, such as using a specific device, or some type of manual querying to the user. Because, as it is known in many user's daily activities (sports, leisure, hygiene, others), they are carried out without the smartphone presence. However, it is assumed that wearable is most of the time in user possession, enabling a ubiquitous interaction with the system.

These devices are currently on the market in diverse ways, from watches, necklaces, bracelets, and even jewelry. What will covers almost all users and situation types.

Concerning that, sensors present in wearables, and data that they allow to acquire, has also been increasing. Since the sensors miniaturization and integration process in the wearables will allow it.

Currently, the most common data that these devices collect is listed in the table 2. In a brief table description, we find different sensor types, which collect several data, important and necessary for the systems optimization process.

Pedometer is currently the most common sensor. It is present in most equipment's, including newer smartphones, and it collects user steps number. Another sensor that has become common, from most practical data acquisition forms, is the heartbeat, and it is currently in several fitness bracelets and smartwatches.

We also have less common sensors, which some are already present in wearables available at domestic market, like Electrocardiogram (ECG). However some are already in industrial market, while others are still in development, and scheduled for market introduction in near future. Examples of such sensors, are Electroencephalography (EEG) for brain activity analysis, Electromyography (EMG) sensor for muscle activity and Photoplethysmogram (PPG) to blood volumetric measurement.

The Peripheral Oxygen Saturation (SpO2) sensor to analyze oxygen saturation, needle biosensor to analyze glucose level, body temperature sensor and sweat sensor [119] are already implemented in medical devices for private use, and some state of the art smartwatches. It is also expected that the continuous sensors miniaturization allow to introduce them in smaller and domestic use wearables.

Portable biosensors design and development, allow them to be integrated into wearables for different physiological indicators monitoring, and has caught the scientific community and industry attention in recent years [103]. The main reason for this fact, is health care costs rising as well recent technological advances, that allow to achieve device miniaturization including bio-sensors, and smart textiles development that integrate these devices in user clothes. With this kind of devices, users can do some health

Sensor	Data	Example	Units
Pedometer	Steps counting	1000	Integer
EMG	Muscle activity walking	90	ms
EEG	Brain activity signals	90	ms
ECG	Electrical activity of heart signals	90	ms
PPG	Volumetric blood measurement	15	mmHg
SpO2	Oxygen saturation	98	%
Bio-sensor needle	Blood Glucose level	5	mmol/L
Heart Rate	Number of hearth contractions	80	bpm
Temperature	Body temperature	36	°C or °F
Sweat	Skin water evaporation rate	25	g/min.m2

Table 2: Wearables - Common sensor's data [112].

care monitoring, at his own home, and with its own devices, without need to pay a much higher cost at health care units.

Also micro-electronics development, particularly in wireless communications, and systems based on wearable sensors, can transform health care, enabling proactive people management involved in health care and status of patients ubiquitous monitoring.

Such systems may include different physiological sensor types, transmission and processing modules, enabling the creation of mobile, discrete and cost effective solutions. Enabling continuously monitoring of different and relevant activity parameters for health, throughout the user's daily life and independent from where they are. Due to all these features, wearables will also be used for interaction with the proposed system.

Smart Speakers

Nowadays we find in the market several intelligent speakers options. Namely all major manufacturers, such as *Apple*, *Google* and *Amazon*, have developed products in this line. Taking advantage of this product line and his software for great innovations in home automation, especially voice control of all smart peripherals available at user's home.

Some of these devices also have integrated displays, thus allowing user feedback, as well viewing information from peripheral such as security devices.

Amazon has the *Echo* product line, which allows the *Amazon's* cloud-based voice service be available on tens of millions *Amazon* devices and third-party device manufacturers. With *Alexa*, we can create natural voice experiences that offer a more intuitive way to interact with technology that is used everyday. Namely with device control and automation like cameras, lights, TV's and others, as shown in figure 18 and 19.

Apple has the *Homepod* available, which in addition to being a speaker, allows integration with the so called *Apple Home Kit* detailed in subsection 2.2.4, thus allowing interconnection with the remaining interconnected devices, as well control with voice instructions.

In this sense too, these peripherals can be integrated into this thesis, as these devices usually exist in

Figure 18: Amazon Echo line [9].

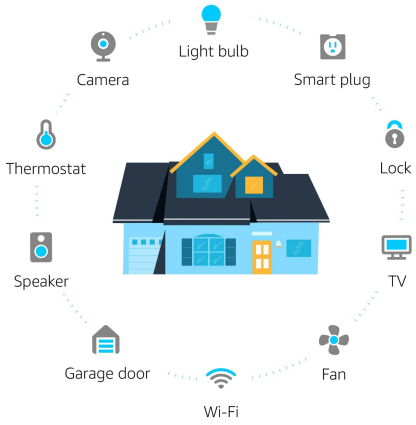


Figure 19: Alexa controlled device types [9].

Figure 20: Apple Homepod [11].

different house spaces, for musical reproduction. Thus, they can be used to user detection and interact with the different performance systems available in the home automation system.

Beacons

Beacons, also known as *iBeacons*, is a more friendly name for "indoor proximity system" technology. In practice, it allows to locate objects or people carrying these, with high accuracy in closed areas. Making a parallel, *Beacons* are for indoors, as Global Positioning System (GPS) is for outdoor environments.

Because of steel structures and other factors construction derived of enclosed spaces, it is natural that GPS signal is weak when users are inside this kind of spaces. In these cases, *Beacons* are a great solution: being relatively inexpensive hardware, and fairly small size to be installed on wall or a shop terminal. Because it is a technology very precise and applicable on a large scale (low cost), they have been used in Point of Sale (POS) processing, events, transit systems, corporate buildings, institutions,

schools, public spaces, as well private residences.

Beacon operation method, uses BLE to detect other devices proximity and transmit a unique identifier number which is then received by the device operating system with which it is communicating. After establish the communication, two actions can be taken by the system, namely:

- *Passive action*, is simply to save (in local memory or in database) the connection existence. For example, the smartphone X approached the sensor Y. In practice this means that a given space user just passed the location where the Beacon is installed.
- *Active action*, happens when communication starts some activity on the user's device. Sending notifications, system status change, or start an action within a specific application, are some of the possible activities. As an example, the case of receiving a notification for a particular sale within a retail space. All this is managed by the users smartphone.

It is important to note that smartphone never sends user data to the *Beacon*, unless the user explicitly allows, to avoid privacy and security problems.

Beacons do not have the ability to detect geographic locations. Only allowing that the smartphone can known the distance between it and the beacon. For other location is necessary to use complementary technologies, such as GPS, Geofencing, or Wireless (WIFI) networks. *Beacons* have interoperability between (iPhone OS (iOS), *Android*, *Windows*, etc.) platforms. Requiring only that devices have Bluetooth 4.0 or higher. *Beacons* do not have any kind of intelligence. Being in this case, all necessary intelligence in the application installed on the device that interacts with *Beacon*. Applications to interact with *Beacons* must be previously installed on the devices because *Beacons* cannot install any application on the device. In figure 21, are some *Beacons* examples, currently present on the market.

Figure 21: Beacons example's [125].

2.2.3 Communication level

Related to communication technologies, for this thesis are explored and detailed BLE, NFC, Wi-Fi Direct, Zigbee, Z-Wave, and also Thread, all of them are following detailed.

Bluetooth Low Energy

Bluetooth Low Energy (BLE) is an extension of *Bluetooth 4.0* standard. It was introduced by the Special Interest Group (SIG) in late 2009 and is optimized specifically for devices that use small batteries

and require a very small consumption [81]. Devices that support BLE communications are certified as *Bluetooth Smart Devices* at SIG. They operate in the same Industrial, Scientific and Medical Radio Bands (ISM) than traditional *Bluetooth* devices, which is divided into 40 channels: 3 for advertising process and 37 for data communication.

It will be a standard for new decade that will support the called IoT. A major advantage over traditional Bluetooth is a very reduced consumption, which is achieved by the device search simplification and connection procedures and activity window reduction, typically by sending only small data packets for a few seconds and entering standby mode rest of the time. At the connection beginning, the client device synchronizes its clock with server device and so it just need to wake up periodically to send data. The time between packet exchange sequence start is called connection interval.

Data packets are usually much smaller than traditional Bluetooth packets, the maximum size of a BLE packet is 2971 bits. They are transmitted at 1 Mbit/s through the air, allowing an active transmit window of only a few microseconds [47].

To optimize power consumption, BLE data transfer speed is of 0.26 Mbit/s compared to 0.7 and 2.1 Mbit/s of the traditional Bluetooth, which is not critical for most applications/users.

It is designed to send small data pieces (state exposure). Data can be made available by local events and be customer read at any time. The interface model is simple Generic Attributes (GATT). Another important limitation of this is coverage radius to be a maximum of 50 meters (with 10 meters limitation for good signal quality in normal working environments such as offices), half of the maximum allowed by the traditional Bluetooth 4.0 [25].

The device server is responsible for establishing the network connection, the client device on the other hand is constantly waiting at advertising mode in order to receive discovery requests. To establish a connection between two or more devices, server device must send a discovery message, which is broadcast sent to all devices within range, repeatedly going through all Bluetooth frequencies. Client devices with active discovery mode, listen to discover messages in the frequency with which they are assigned and, when they receive this kind of messages, send a response to the server device, containing its address and class, as well additional data that may be requested.

Device server can then automatically or by user's selection, send a connection request for the specified client device, if this one is configured to accept connection requests, it will reply with a response. Otherwise will refuse connection attempt [70]. After the server device receives a positive connection response, the connection is successfully established, on both sides.

Last version is the Bluetooth 5.3 specification, and it was presented by the Bluetooth SIG in July 2021 and aim, among other goals, to make the technology more suitable for applications based on the IoT, such as smart speakers or lamps, as well wearables, such as smart watches or bracelets.

To this end, Bluetooth 5.3 has a connection subtracting mode that allows a device to move from a high to low performance state and vice versa more quickly, contributing to energy savings.

This is useful, for instance, for a smartwatch that transmits data to a smartphone: once that data has been sent, the device can more quickly switch back to a reduced work mode to save battery life.

The new version also introduces a channel rating enhancement. To reduce the risk of interference, the Bluetooth frequency range is divided into several channels. Those that are congested or noisy are classified as "bad" to be avoided during communication.

In previous technology versions, this channel classification is established only by the main communication device. In Bluetooth 5.3, both the main device and the peripheral (the one with which the main communicates) participate in the channel classification process. Thus, the accuracy on the channels to be avoided increases, making the connection more stable.

Another novelty of Bluetooth 5.3 that contributes to energy savings and, in addition, can improve connection efficiency, is the introduction of the *AdvDataInfo* (ADI) parameter in advertising packages (remembering, the way the device transmits a signal to warn that is available for connection).

Figure 22, illustrates some of the most common application examples of BLE technology.

Figure 22: BLE example applications.

Near Field Communication

Near Field Communication (NFC), is a wireless communication technology, Japan is one of the NFC adoption pioneers, and the technology as taken shape in 2002 by *Philips* and *Sony* hands. From beginning, the idea was to use the technology on mobile devices, mobile phones, digital cameras, laptops, etc [136].

At the time, the two companies were determined to promote the NFC, which is why they presented its specification to Standards organization for information and communication systems (ECMA), an organization responsible for the standards of communication and information systems. In 2003, the technology was recognized by the standard *ISO/IEC 18092*.

However, the NFC only began to heat up in mid 2004, when the NFC Forum was created, an organization that gathers now about 150 companies interested in the development and use of applications based on NFC. Among them, are: *Google*, *PayPal*, *RIM*, *LG*, *American Express*, *Nokia*, *Samsung*, *Intel*, *NEC*, *Visa*, *Huawei* and *Qualcomm*.

Note that NFC is in some way based on RFID, a more consolidated technology that enables, as the name implies, radio frequency identification applications. But since there are so many options for this purpose, such as WIFI and BLE, the advantage of its adoption in relation to others, can be seen, not so much by what technology does, but mainly how it does.

In a few words, the NFC is a specification that allows wireless communication between two devices through a simple approach between them, without the need to enter passwords, click buttons, or take

any action to establish the communication. So long as the devices are close enough, the communication is automatically established and the corresponding action taken. These devices can be mobile phones, tablets, ID cards, electronic tickets and any other item capable of support the installation of a NFC chip [41].

The distance that the devices must be with each other to establish a connection, is really short (the maximum is about 10 centimeters), to make clear the intention of communication, without that it can happen accidentally.

A range so limited, should not be seen as a disadvantage, but a security requirement. The NFC technology is designed to allow communication between two devices and no more than that. The principle is simple: one plays the role of initiator, having the task of starting the communication and control the exchange of information. The other plays the *Target* role, and should respond to the *Initiator* requests. Communication is established using radio frequency, from 13.56 MHz, with data transmission speeds ranging between 106, 212 and 424 kb/s (kilobits per second).

Transmission can occur in two ways:

- *Passive*: in this mode, only one of the devices (usually the initiator) produces the connection RF signal. The second device is only powered by it. This makes it possible to put NFC tags on items that do not receive direct power supply, such as cards, packaging and posters.
- *Active*: in this mode, both devices produce the radio signal. It is for example, the mode used in payment systems involving a smartphone and a receiver in the stores payment terminal.

We must also consider that there are three operating modes, which together increase the usage possibilities:

- *Reading and writing*: based on the passive communication, allows reading or modifying data on a NFC device such as a receiver that discounts existing credits in a travel pass.
- *Peer-to-peer*: it is a method for bidirectional exchange of information between the two devices, each can send or receive data to the other. It may be useful, for example, for exchanging files between two mobile phones.
- *Card emulation*: in this mode, the NFC device behaves/works and is seen as a smart card.

NFC technology may be used in a wide number of applications, including most critical applications involving confidential user data. An example can be seen in the *Google Wallet* service [57], which enables the user to pay the bills using a smartphone with *Android* operating system (mobile payment) instead of the traditional credit card or cash. The process is quite simple: the user approaches his smartphone to a receiver, which may be at the cash register of the shopping center, and should both devices have an NFC chip. Once communication is established (about a few seconds), the terminal receives the process

information, as the total value of the purchase. Then just the user confirm the operation, by entering its Personal Identification Number (PIN) on the phone to confirm the payment [41].

Other expected applications for NFC can be:

- *Identification*: NFC can be inserted into an ID card, for example, to identify the arrival of an employee to the company or his access to a particular sector;
- *Virtual tour guide*: if the user is in a museum, he can approach the mobile phone at a nearby receiver to have on its device more information about the exhibition;
- *Advertising*: while waiting for the bus, the user can approach his mobile to an advertising poster and, by doing so, get for instance a discount to the advertiser's store;
- *Price*: to know the price of a product on the shelf or even more details about this, just approach the phone to the item, for additional information appears on the screen [136].

Figure 23, illustrates some of the most common application examples of NFC technology.

Figure 23: NFC example applications.

Wi-Fi Direct

The Wi-Fi Direct has emerged through the WIFI Alliance [7], the international association in charge of certifying WIFI. The Wi-Fi Direct main objective is that simple tasks only require simple connections.

For instance, printing from a computer or mobile phone in a wireless printer, sharing images with another person in the same room, or to stream video from the phone to TV, or other peripheral. In none of these situations is required a Internet connection, but only the concrete device connection (printer, TV, scanners, and others).

With Wi-Fi Direct, it becomes easier and also faster. Wi-Fi Direct devices can connect to each other without having to go through an access point, may establish ad-hoc networks when necessary, allowing to search the devices that are available and choose which it is intended to do the connection.

This type of technology is very similar to BLE, however is faster. At the security level, the Wi-Fi Direct technology uses WIFI Protected Setup (WPS) and Wi-Fi Protected Access 2 (WPA2) to prevent unauthorized connections and maintain the communications private.

There are two ways to establish a connection, using the physical buttons (pressing the physical button on both devices) or via PIN code. The Wi-Fi Direct technology includes two potentially useful things, direct Wi-Fi device discovery and Discovery Service. The device, in addition to know who are the devices in range, also knows what type of devices and the services they offer. For example, in case we want to proceed to the display of an image, it will only be displayed on devices to which we can send images. In the case of printing, also it will only be visible to printers, or devices that are connected to printers. Fundamentally that happens in the process previous to the connection, so there is no waste of time in trying to connect with devices that not allow to perform the intended tasks.

Figure 24, illustrates some of the most common application examples of Wi-Fi Direct technology.

Figure 24: Wi-Fi Direct example applications.

ZigBee

The ZigBee protocol is based on the *IEEE 802.15.4* standard for personal wireless networks, similar to the Wi-Fi 802.11 standard. The main design requirements can be formulated as: low latency, mid-range bandwidth (lower than smartphone, but higher than other IoT technologies) and the use of unlicensed industrial, scientific and medical radio frequency bands (ISM bands), which limits the spectrum load.

Less data intensive scenarios with low latency and protection requirements that are within the capabilities of the ISM bands is why creators should consider ZigBee. They are similar to what you know as WIFI, they emit in the 2.4 Ghz band, although there are also 2 specific bands, at 868 Mhz for Europe and 915 Mhz for the United States.

One of the aspects that sets ZigBee apart from similar technologies is the network protocol, ZigBee PRO. It was created to address specific challenges such as region-specific implementations, cross-band communications, and most importantly, built-in security. This makes it a robust network protocol for the intended applications.

One of the great advantages is the low consumption it has and the hardware needed to create a device is much smaller than what is needed to do the same with BLE or Wi-Fi, which reduces the cost in part.

Finally, another advantage is that, thanks to the way it works, which we will explain below, the devices, if a “node” falls, could be reorganized so as not to lose connectivity.

The ZigBee mesh topology allows data relay between ZigBee devices to transport data over long distances.

There are three types of nodes in a ZigBee mesh network: they are coordinators (controller), routers (router) and end devices. Each ZigBee network requires a coordinator, which is a device responsible for forming the network and forwarding traffic. After a network is formed, the coordinator adopts the ZigBee router capabilities, which acts as an intermediate node that transmits data from other devices. A router never goes into sleep mode. Routers can also be a ZigBee end device.

End devices can only communicate with the main nodes, that is, the coordinator or the routers. These devices are energy efficient devices and can go into sleep mode to save energy. Each parent node can serve up to 20 ZigBee end devices. A ZigBee network example can be seen at figure 25. And figure 26, illustrates some of the most common application examples of ZigBee technology.

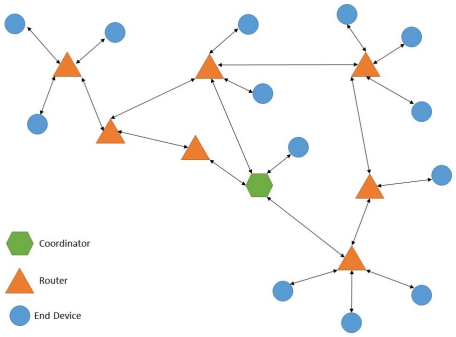


Figure 25: ZigBee network example [8].

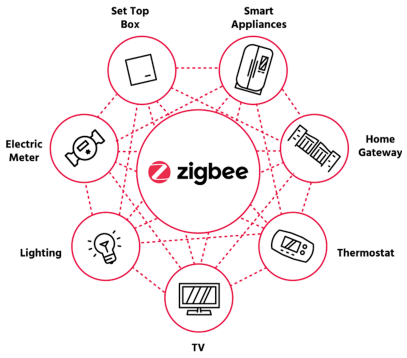


Figure 26: ZigBee example applications [8].

Z-Wave

The Z-Wave protocol is an inter-operable, wireless, RF-based communications technology designed specifically for control, monitoring, and status reading applications in residential and light commercial environments. Broadly deployed (with over 100 million products sold worldwide), Z-Wave is by far the world market leader in wireless control, bringing affordable, reliable and easy-to-use ‘smart’ products to many millions of people in every aspect of daily life.

Z-Wave encompasses a broad ecosystem of smart products and services that work seamlessly between brands and versions. This interoperability, which has been the hallmark of Z-Wave technology since 2002, is achieved and maintained through Z-Wave certification, a testing program administered by the *Z-Wave Alliance* consortium.

Z-Wave certification ensures that all Z-Wave products work together with each other regardless of brand, including backward-compatibility between versions. The certification process includes technical testing, programs for uniformity of marks, and enforcement of the certification standards.

While other technologies claim interoperability, only Z-Wave offers interoperability at the product level. This ensures manufacturers, integrators and end users that their products and services will work together with all certified Z-Wave products.

The Z-Wave ecosystem encompasses more than 4,000 interoperable products. These products work together through stringent enforcement of Z-Wave certification, performed at independent test labs, and overseen by the *Z-Wave Alliance*.

Figure 27, illustrates some of the most common application examples of Z-Wave technology.

Figure 27: Z-Wave example applications [77].

Thread

The *Thread Group* is a group of hundreds of companies that support a mesh network protocol called *Thread*, it is composed by the major players like *Amazon*, *Apple*, *Google*, *Nordic* or *Siemens*.

In the past years, all the attention has been focused on *Zigbee* and *Z-Wave*, the established standards of the smart home. But *Thread* could yet be the single most important wireless protocol for the smart home future, especially as it's at the forefront of the *Matter* smart home initiative.

Thread is an Internet Protocol version 6 (IPv6) based networking protocol designed for low-power IoT devices in an *IEEE 802.15.4-2006* wireless mesh network, commonly called Wireless Personal Area Network (WPAN). *Thread* is independent of other 802.15 mesh networking protocols, such a *ZigBee*, *Z-Wave*, and BLE.

As primary *Thread* features we can identify: simplicity, simple installation, start up, and operation;

security, all devices in a *Thread* network are authenticated and all communications are encrypted, reliability, self-healing mesh networking, with no single point of failure, and spread-spectrum techniques to provide immunity to interference, efficiency, low-power *Thread* devices can sleep and operate on battery power for years, and scalability, thread networks can scale up to hundreds of devices.

Like *Zigbee* and *Z-Wave*, *Thread* can connect all the devices together in a giant mesh, and unlike *Zigbee* and *Z-Wave*, *Thread* doesn't require a smart home hub to connect them, they just require a *Thread* border router, as can be seen at figure 29.

A *Thread* border router connects *Thread* devices to other *IP-based* networks, such as Wi-Fi or Ethernet. Some different devices nowadays include Thread border routers like the *Apple HomePod Mini*, or the latest-gen *Apple TV 4K*. From *Amazon*, we also find *Thread* border routers in the *4th-gen Echo* smart speaker and all Wi-Fi 6 and *Eero* mesh routers. *Google* has been building *Thread* radios into its devices for years and the *Nest Hub Max* smart display, *2nd-gen Nest Hub* and the *Google Nest Wifi* mesh router all now act as border routers.

In a *Thread* network, nodes are split into two roles, the Router and the End Device (ED). A Router is a node that: forwards packets for network devices, provides secure commissioning services for devices trying to join the network and keeps its transceiver enabled at all times.

An ED is a node that: communicates primarily with a single Router, does not forward packets for other network devices and can disable its transceiver to reduce power.

At figure 28 we can see a *Thread* network example.

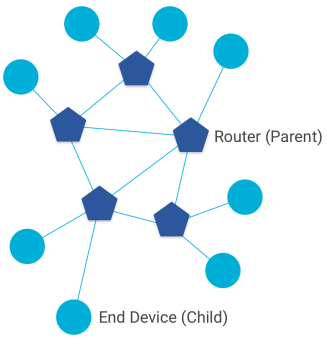


Figure 28: Thread network [76].

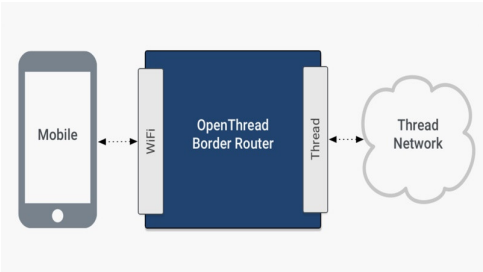


Figure 29: Thread border router [131].

The devices connect to each other without a single point of failure, which also means the network can "self-heal". So if one device goes down or a connection becomes spotty, the network can adjust and carry on without breaking.

Most importantly, *Thread* is inter-operable by design. It uses open standards like IPv6 and a IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) foundation. In other words, all the devices would be able to talk each other, no matter the manufacturer.

Nowadays, when we try to pool smart home devices together into routines, we use services like *HomeKit*, *If This, Then That (IFTTT)*, *Alexa* and *Google Assistant*. They essentially act as remotes that make it seem like the devices are working together seamlessly. Like when we say good night and your door locks and your lights turn off.

Thread, on the other hand, would let the devices directly talk to each other. So smart lights from *Philips* could talk to a *LG* refrigerator without a hub or service getting in the middle of it. It also uses Advanced Encryption Standard (AES) encryption, which *Thread Group* says closes holes that are present in other networking protocols.

A home *Thread* network can support over 250 devices with multiple hops. Also related to energy use, *Thread Group* says *Thread's* power efficiency means that devices running on AA batteries will last for years, as it runs on the power-efficient *IEEE 802.15.4* Medium Access Control Layer (MAC)/Physical Layer (PHY) and uses short messaging between devices to conserve power.

Retro compatibility is also possible, because *Thread* technology can be enabled in some older devices using firmware, as long this devices support the *802.15.4* protocol, they can be upgraded to support *Thread* [131].

2.2.4 IoT Platforms

As mentioned in section 2.2, the major players in the technological market (*Apple* and *Google*) are aware of the homes automation growing, and the need to centralize information and have control of it, so that users can make the management of all existing technology in housing, from a single point, regardless the installed devices manufacturer.

This section analyses the platforms presented by these manufacturers, their purpose, and potential gaps. And also details Matter, and some concepts about Geofencing and Smart Cities.

Apple HomeKit

The *HomeKit* is a framework for communication and control devices connected to the user home. Using this framework, users can discover existing devices and proceed with the configuration, as well as, create actions to control them. *Apple* has developed a framework called *HomeKit* to simplify the current status in terms of home automation.

With this framework, *Apple* created a common language that smart devices, regardless of manufacturer, can understand. Having the integration, supported by the well known voice assistant *Siri* [12] provided by the brand, thus allowing the capability of controlling intelligent devices using speech. So, it

is possible to have a number of intelligent devices (lamps, alarms, etc.) from multiple manufacturers, that are able to communicate and work together. This framework directly supports BLE and Wi-Fi as communication technologies. Other technologies such as *Z-Wave* or *ZigBee* are supported indirectly by using bridges [92].

One of the biggest advantages of this framework, is the possibility of use voice control, derived from the integration with *Siri* technology. Real automation, requires voice and gesture control, avoiding the use of smartphones and associated applications in the interaction with the devices. The interaction is much easier and natural using voice commands, for example, through a smart watch. Other features that distinguish *HomeKit* of competitor alternatives, will be described below.

Apple bet on two main technologies for communication with smart devices, these being the BLE and Wi-Fi. This position is justified by the attempt to create a consolidated standard of communication, and so dispense the use of hubs (bridges), that competition uses to support other communication protocols (*ZigBee*, *Z-Wave*) [134].

Additionally, this allows most of the communication processes to be transmitted locally and specifically to the devices, when the control is performed within the local network of housing. Rather, other hubs on the market, which typically commands passing first through the cloud and then sent to the local hub and then to the specific device. In this case, there is no dependence of the cloud, for operation and control of internal devices.

In short, this solution demonstrates how *Apple* using communication standards, enables the unification of devices using the existing smartphone, and the network connection present in the house. This avoids the necessity of additional costs and difficulties for the user with more devices. Having the role of central point, passed to the smartphone.

Please note, that in this case the cloud is also available, and can also make up a central point if the developers opt for it. The cloud is currently used to centralize information from all devices connected to the home, with all the mobile devices that the user own. Having this the role of synchronize all data between devices, keeping these always with the most current information.

Figure 30, illustrates the presentation of *HomeKit* by *Apple* during the conference Worldwide Developers Conference (WWDC) on 2014.

Figure 30: Apple Home Kit [123].

Android Things/Google Brillo

In early 2015, at the *Google I/O* conference organized by this multinational, was presented the *Brillo* and *Weave*, the proposed solution of this company for the interconnection of smart devices.

Brillo is an operating system derived from *Android* but adjusted for IoT devices. It supports WIFI and BLE. The *Brillo* system promotes the connectivity of the various devices so that they are able to communicate through a common language. Allowing the connection of devices through the user's phone, so that it can control devices such as thermostats, alarm, etc. *Brillo* also allows the management and storage of data collected from the sensors, and is designed to run on connected devices with few hardware resources (memory and low-frequency processors). According to *Google*, beyond WIFI and BLE technology, *Brillo* has minimal system requirements (32 MB RAM).

This emphasis on low power consumption ensures that the appliances and devices, such as door locks, can also connect to the system. *Brillo* brings simplicity and speed of software development for hardware, providing a lightweight operating system based on *Android*, central services, a development kit, and a developer console. Can be chosen a variety of hardware features and customization options, and quickly move from prototype to production and make a scale management with updates Over the Air (OTA), metrics, and reports. On the other hand, *Weave* is a multi-platform common language that will allows *Brillo* devices, smartphones, and the Internet to communicate with each others. Existing *Android* devices will be able to automatically detect *Brillo* devices. The *Weave* uses a common language that allows sensors and platform devices to communicate through a simplified process using the Cloud. This solves the fragmentation problem that currently exists in home automation, allowing connected devices to communicate in several ways, as well using different software.

In Figure 31, can be seen the operation envisaged between the different devices that can integrate this platform.

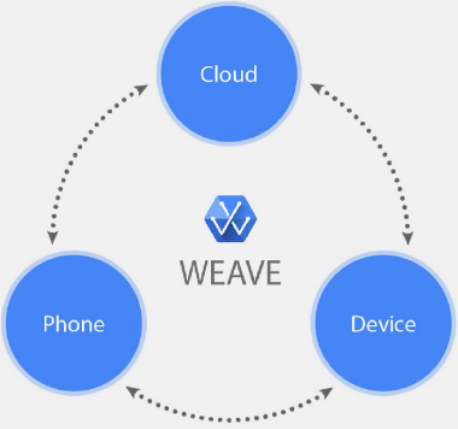


Figure 31: Google Brillo and Weave [123].

Recently, *Google* announced its new IoT initiative based on *Android*, the *Aptly* named *Android Things*. It was back in May 2015 that *Google* announced *Project Brillo* as its IoT operating system, having looked at the feedback it received from developers about *Brillo*, Google decided to ditch it and create *Android*

Things. By adding *Android* to the name, *Google* is emphasizing that developers are able to use familiar *Android* Software Development Kit (SDK), Application Programming Interface (API) and services including the *Google Cloud Platform*. *Android Things* is only a preview at the moment, however enough of the final operating system is working so that developers can start creating *Android Things* based projects.

Matter

Matter started life in 2019 as Connected Home over IP (CHIP) project, a collaboration between some of the biggest players in tech: *Apple*, *Google*, *Amazon*, *Samsung*, the *ZigBee Alliance*, and other tech brands, which aimed to create a unified smart home standard.

The idea was that this would make it easier for manufacturers to develop products that work both with all three major voice assistants and also each other. At May 2021, the name was changed to *Matter*, while the *ZigBee Alliance* was rebranded as the Connectivity Standards Alliance (CSA).

Rather than introducing any entirely new technology, which would slightly defeat the object of being a unifying force, *Matter* uses only existing standards: Ethernet, Wi-Fi, BLE for initial pairing, and *Thread*.

As well making things simpler and improving interoperability, *Matter* also wants to make smart home more reliable and secure, and the inclusion of *Thread* will be key to that.

The list of devices that *Matter* will cover is pretty exhaustive, lighting and electrics, heating and cooling, locks and security devices, windows and blinds, and TV's are all included. And the line-up of alliance members and participants includes more than 240.

In short, it should have the whole smart home covered. At figure 32 we can see a smart home network topology composed by different *Thread* devices and *Matter* controllers.

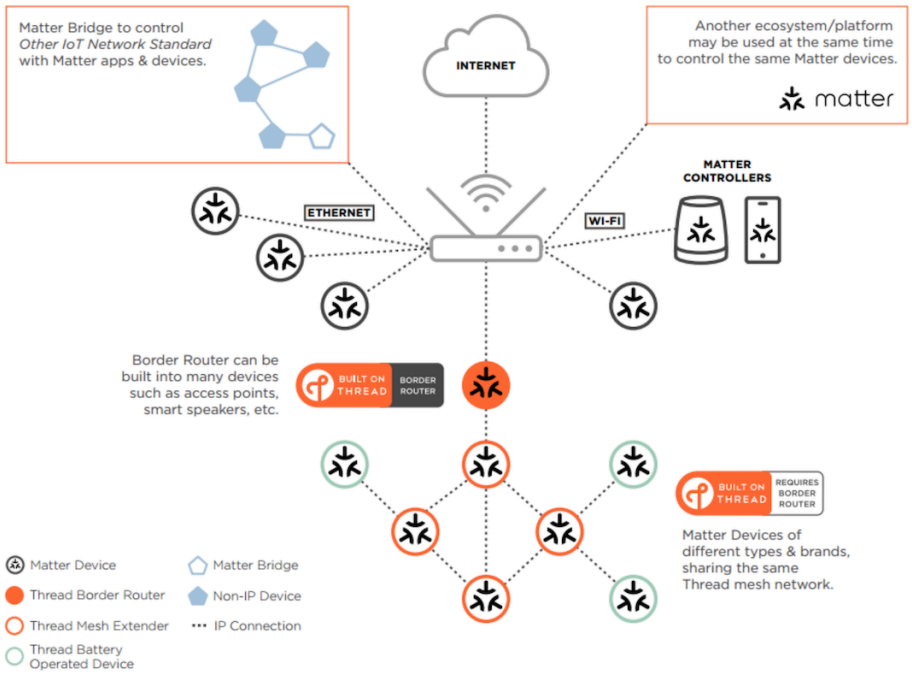


Figure 32: Smart home network topology [76].

Geofencing

Geofencing makes use of technologies such as RFID, GPS, cell phone antennas and even Wi-Fi signals to establish a virtual geographic perimeter. From this, it is possible to configure notifications triggering via push, Short Message Service (SMS), advertising on social networks or several other types of alerts or commands for devices that cross this perimeter, based on their geolocation.

At first it may seem simple, with limited functions, but the possibilities for using this technology are very wide. That is why it is so important to understand what geofencing is and how it can contribute to improving the functioning of companies.

Geofencing is a way to engage consumers based on hyper localization, and it can do a lot in terms of triggering immediate sales, as well understanding the buyer's mindset.

For example, a store could standing up a simple geofence in an area that surrounds its physical location. When users pass, they receive an alert or agreement triggered by location making them considerably more likely to stop and buy.

Alternatively, a car dealer, for example, could set up a geolocation designed to target individuals who are leaving a rival dealership after looking for a vehicle. Hitting them with a zero percent financing offer on a comparable car model at that time is more likely to make them come to the comparison shop, or at least consider an alternative option.

Finally, even if a geofenced offer or notification does not trigger an immediate visit or sale, it does allow a company to know exactly where a consumer went, and where they were when they received the message, which can help with refining to target efforts future based on which communications were most successful.

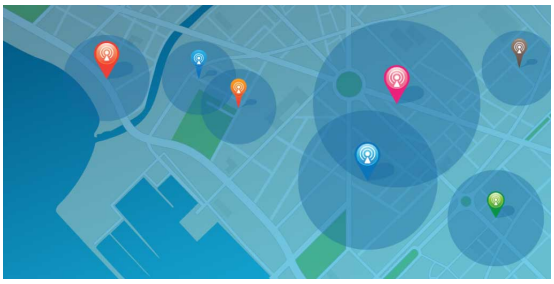


Figure 33: Geofencing.

Smart Cities

Currently, the world's main cities struggle to become more innovative spaces through Information and Communications Technology (ICT). Its use not only implies improvements in the provision of services, but also paves the way to transform our cities into true smart cities.

To be classified as a "smart city", it is not enough to be sustainable in just one area. The implementation of new technologies, the participation of citizens in public life, the search for efficiency in the management of available resources, increasing the citizens and visitors quality of life, are some of the characteristics that make a city smarter.

We can say that the smart cities evolution is on the rise, causing more and more cities to enter this plan. However, those that are not, already know what a smart city is and what benefits it can bring to both citizens and institutions.

The European Union defines Smart Cities, as a set of systems and people that interact intelligently, using energy, materials, services and resources in a sustainable way. We can say that it is a kind of fusion between people and systems that move in harmony, seeking to guarantee the urban spaces sustainable development. Basically, smart cities are those that invest in technology to improve municipal management and provide their citizens with a better quality of life and spaces sustainability.

When we talk about smart cities, we are referring to those cities where an optimized functioning of all administrations, whether public or private, in topics involving urban planning, sustainable and inclusive cities, the environment, and many others, has been incorporated, always with the objective of satisfying citizens and institutions.

When we talk about the smart cities characteristics, we can mention the following:

- Efficient urban planning;
- Environmental sustainability improvement;
- Technologies applied to education and health;
- E-commerce system;
- Shared data (open data).

Also, as main smart cities benefits, we can identify the following:

- IoT close relationship;
- Improves citizens relationship;
- Environment protection;
- Mobility improvement;
- More liveable cities.

Also in Portugal, more and more municipalities are attentive to technological innovations capable of building smart cities, promoting a healthy lifestyle, responsible use of public spaces, sustainable development and a growing quality of life.

The percentages give substance to this idea. Today, more than half of the world's population lives in large urban centers and the trend, according to the United Nations (UN), is to increase. By 2050, 70% of the population will live in large cities. All this brings a serious set of challenges to municipal management, namely with regard to pollution, car circulation, housing or access to services.

2.3 Ambient Intelligence

This section introduces the AmI definitions and concepts, details an extensive review of different projects in this field. And also identifies the challenges and trends that currently exists.

2.3.1 Concepts

The AmI field, currently appears as an emerging field of study of information systems, and with a perspective of potential future impact. Scrutinizing the term, it is defined by the *Merriam-Webster Dictionary* [91] as existing/present everywhere (ubiquitously).

But in the literature we have several other definitions/concepts like:

- A developing technology that will increasingly make our everyday environment sensitive and responsive to our presence [2].
- A potential future in which we will be surrounded by intelligent objects and in which the environment will recognize the presence of persons and will respond to it in an undetectable manner [48].
- "Ambient Intelligence" implies intelligence that is all around us [86].
- The presence of a digital environment that is sensitive, adaptive, and responsive to the presence of people [39].
- A vision of future daily life... contains the assumption that intelligent technology should disappear into our environment to bring humans an easy and entertaining life [40].
- A new research area for distributed, non-intrusive, and intelligent software systems [109].
- In an AmI environment people are surrounded with networks of embedded intelligent devices that can sense their state, anticipate, and perhaps adapt to their needs [132].
- A digital environment that supports people in their daily lives in a nonintrusive way [3].
- A digital environment that proactively, but sensibly, assists people in their daily lives [14].

Analyzing the current landscape, the reference company in the AmI field is undoubtedly the *Philips*. In fact, it was this company that in 1998 proposed the concept of AmI in a series of workshops, which were organized by this multinational [146]. At these workshops have been developed different scenarios that would lead to a consumer electronics industry, which was considered at the time as fragmented, until an expected reality in 2020, with friendly devices that make available support for information, communication and entertainment ubiquitously.

This development involves several departments of *Philips*, including research, design and global brand management. However, the first official publication mentioning the term AmI happens in 1999 in the

German magazine *IT Monitor* [4] and stresses the importance of *Mark Weiser* work, who worked for a long period of time in a new concept for mobile computing called ubiquitous computing [137].

This concept had enough influence and is considered the beginning of new developments, such as pervasive computing launched by *IBM*, and AmI by Philips.

On the other hand, ambient intelligence designation was defined by the advisory group for the information society and technology of the European community.

From its definition, we can see that AmI has a decisive relationship with many computer science areas. The contributing technologies are organized into five areas, as shown in figure 34.

A key factor in AmI research is the presence of intelligence, as the *Russell and Norvig* notion of an intelligent agent [118]. In this way, the AmI algorithm perceives the environment state, and users using sensors, reasons about the data using a variety of AI techniques, and acts upon the environment using controllers in such a way that the algorithm achieves its intended goal.

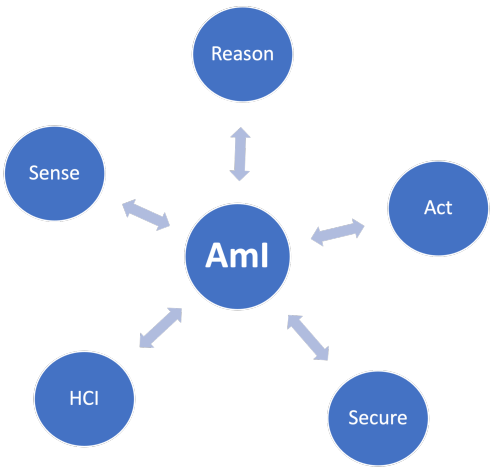


Figure 34: AmI and contributing technologies.

The process is illustrated at figure 35. Following, we focus on technologies that assist with sensing, reasoning, and acting.

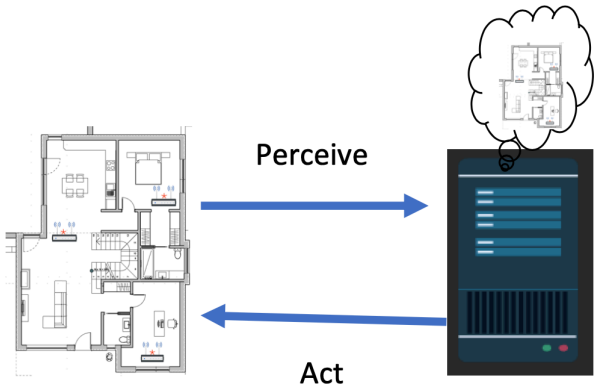


Figure 35: AmI agent interaction with the environment.

The agent perceives the environment state and residents using sensors. The agent models and reasons about this information, ultimately using it to make a decision. After, the environment state is changed through the present actuators.

On the other hand, while AmI draws from the AI field, it should not be considered AI synonymous. The Information Society Technologies advisory group (ISTAG) identifies five key technologies that are required to see AmI as a reality [48]. Two of these technologies fall outside the typical AI research scope, namely Human Computer Interface (HCI) interfaces and secure systems and devices.

Among others, as we can see at figure 36 the three major characteristics, that characterize AmI are: intelligence, ubiquity and transparency. Because each AmI concept element, implies supporting the lives of users in different contexts. The combination of intelligence and technology should help people's lives during the daily life, and evolve to the user state adapt. Also in these cases, the emotional part is important and will affect the way user's live, individually and independently upon its quotidian.

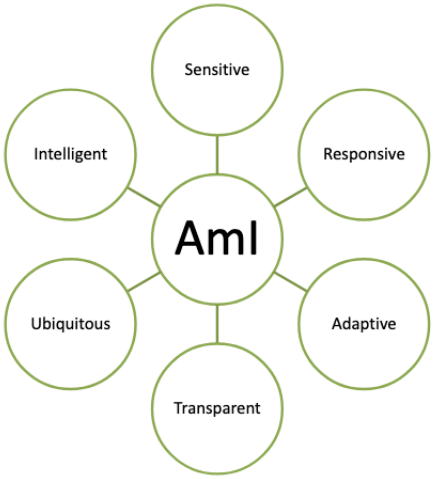


Figure 36: AmI characteristics.

2.3.2 Projects

Smart environments are already under study [98] [115] for some time, and during the state of the art survey in this field, were identified the reference projects worldwide, that are described below.

The *MavPad* project involves the replication of a small apartment with traditional divisions such as bedroom, living room, bathroom and kitchen. This apartment has been fitted with different types of information-gathering sensors, concerning the context and users, trying with that, give intelligence to the environment [145].

The *iDorm* project uses motion sensors, temperature, humidity and light, and already have automation mechanisms for monitoring the doors, heat, and blinds. The same is characterized mainly by the use of techniques based on fuzzy logic, for the process of learning the preferences of users within an environment. It was developed at the *University of Essex* (United Kingdom).

It was created a dorm, which served as an experimental environment. It is an inhabited environment that enables various activities such as sleep, work and leisure activities. To do so the environment has bed, desk, wardrobe and entertainment system, also containing various sensors for detecting temperature and occupancy (user seated, user lying), humidity and light levels. The present actuators allow opening and closing doors, and adjusting heaters and curtains. The computational components incorporated in this project include: an integrated agent that receives the time of day, as well as, the sensor readings within the environment (ambient light and temperature) and outside the room (ambient light and temperature), the state of windows (open or closed) and user activity (sitting, sleeping, using the entertainment system or using the computer). This agent contains the user's learned behaviour and, taking advantage of this and the information gathered from the sensors using a fuzzy logic technique, determines which actions to take. These actions can go through adjustment of heating/cooling, lights or blinds. This project also includes a robot equipped with obstacle avoidance, controlled by the *iDorm* agent. Allowing the food transportation, medicines, or other necessities. The agent has robot real-time location, which allows to send instructions for this move to a destination depending on the items it carries.

Two types of rules were created, static and user-independent, they have the kind of reaction in case of emergency and the light and temperature settings when the room is not occupied. And the rules that are being learned regarding the user's personal preferences.

For individual identification of users is used an ID. The process consists of a period of monitoring and activity analysis each time a new user enters the environment. These examples are then used in the learning phase. Since this is based on negative reinforcement, it is assumed that users will make changes in the environment always they feel uncomfortable. The technique used in this process was the Incremental Synchronous Learning (ISL) based on fuzzy logic.

After these processes (monitoring, learning), the *iDorm* agent initiates the environmental monitoring process. When there are changes in user preferences, the learning process checks whether to change any of the rules is needed. The authors state that is also stored a set of information (limited to 450 rules), containing the rules of the previous occupants. This information is used after the initial monitoring phase, and the system tries to find the best match with the stored data. So, the learning starts from an initial rule.

In the experimental results of this project, is reported environmental occupation by a user (5.5 days/132 hours). This had the ability to interact with the environment using a PDA. After the experimental period, the number of rules learned over time was analyzed. It was based on the assumption that if the system had a good performance in learning preferences, then they would be few settings interventions by the user, and these interventions would also later be learned by the system. It was found that within 24 hours, a large number of rules is learned, and after that the number decreased significantly up to 60 hours. In the period between 60 and 72 hours there was an increase in the learned rules, and after 72 hours has not learned any other rule.

The change in the period between 60 and 72 hours can be explained due to the introduction of new activities by the user in their daily routine during this period. It also concluded that the embedded

agents dramatically reduce the need for user intervention and that the information stored, also reduces the learning time that the ISL needs. As future work is aimed the implementation of this project in environments with multiple users, and in fully functional apartments.

In the case of *Sensor 9K*, this is a middleware layer that allows to promote the creation of applications for intelligent environments [44].

A relatively more recent project than the above is the *I3A*, it covers a building using a network of sensors that allow to have different data from each of the sensors in the network, including: temperature, Carbon dioxide (CO2), Carbon monoxide (CO), humidity , state of appliances, etc. This project allows prototyping solutions for smart environments, taking advantage of the ability of such sensors can be programmed individually [45].

The *Saves* project includes an intelligent environment, designed for use in construction and user occupancy profiles to maintain and regulate the temperature of a building [79].

The *MavHome* project (*Managing an Intelligent Versatile Home*) was developed at the *University of Texas*. Aiming to create a room that can act as a rational agent, which has sensors and actuators, to get information about users and so can provide comfort and efficiency. In this case, comfort is referred to as the atmosphere of the environment, including temperature, ventilation, lighting, etc. As efficiency being assessed by the gas and electricity costs to achieve the desired comfort.

Being main objectives of the project, reduce user intervention in environmental monitoring and reducing energy consumption. The architecture of this project, uses a rational hierarchy of agents. Each agent has four layers: decision, information, communication and physical. The sensors collect environmental information, transmit the data to the agents, the information is stored in the database, and new information can be passed to the decision layer, where it is decided whether any action should be taken. In this case, the decisions are passed to the appropriate actuators of the physical layer for implementing the actions. The communication process uses the Common Object Request Broker Architecture (CORBA) model, is point to point and publish-subscribe.

This project combines several algorithms for learning the habits of users, it is assumed that only one user is in the home each time. The agents try to predict their next move, and perform action policies to changing environmental conditions.

The learning of user habits is achieved by mining the data collected from user interaction with the environment, and of these extract patterns. The components of this project have been implemented and tested in a work environment and in an apartment of a student. Information gathered from sensors including light, temperature, humidity, movement and status of doors and windows. And the results obtained show a reduction about 72 to 76% in manual interaction by the user in both tested environments.

The *GENIO* project come from the effort of two companies in Spain. *Fagor* and *Ikerlan*, who want to coat the homes of ambient intelligence. The name of this project comes from the word in the Spanish language concerning "genius", that is, ensuring their wishes. In this project, the various appliances (washing machine, oven, refrigerator, sensors and alarms) are networked and are managed by a central controller. This controller allows to keep a conversation and respond to user instructions in their native

language (Spanish).

This controller is called *Maior-Domo* and is represented by an avatar. Some of the functions of the devices present in this project are: the oven has a recipe database, a recipe is chosen and the oven fits for their preparation. The refrigerator using RFID technology allows to have knowledge of all products stored in this, by reading the tags present in products.

The computer *Maior-Domo*, communicates with the different electronic devices, having this voice recognition developed using Java, VoiceXML (VXML), Java Server Pages (JSP), and JavaBeans. And also a component to enable text-to-speech and a microphone for the user to communicate wireless with the *Maior-Domo* from anywhere in the room.

Some scenarios presented for this project, go through voice instructions in order to read the user's emails, turn on the washing machine, check the fridge contents and prepare the necessary shopping list, prepare a recipe, as well at the multimedia level, playing music by gender or author chosen.

The main contributions of this project, are the several possible settings, control of multiple devices, and voice processing. Furthermore, the identified future work includes identifying the user who give commands to the *Maior-Domo*, to ensure that users who aren't assigned to control certain functions will not succeed in doing so, for example to ensure that children do not control the security functions.

Nowadays the most famous and recognized testing environment is, however, the *HomeLab* of the multinational *Philips* [69]. This is probably the more complete project in this area, and that involves more valences at study. Because on this project beyond the traditional collections of information, are also collected images and sound, using hidden cameras for this purpose.

Currently, the state of the art relating to the detection of persons in interior spaces, as well as their movement between different parts of space, is made mainly by static sensors, and the detection of the position of the person (sitting, lying down, standing) and their movements are measured through computer vision, using shape recognition. However, this process is time consuming, expensive, as well as, subject to prior preparation of space. What prevents their use from the perspective of the average user, independently of the space where it is. The *Philips HomeLab* [69], has been using such techniques to study and automate environmental characteristics to suit user's preferences.

This project also permits brightness control, not only the intensity level, but also in terms of its hue. Are then collected and evaluated the different reactions of the users, to the changes introduced in the environment.

These environments are used to test and validate the environmental intelligence theory, which is essentially the behavioral analysis, stress assessment and user's routines. This information, currently concentrates the world research focus in this area.

Environments, such as the simulated by *Philips HomeLab* [69] try to gain knowledge about the user experience, earn user behaviour knowledge and take advantage of this by inserting it in the new innovation cycles. The study of AmI in these environments, intend to find new methods to acquire this knowledge in different situations. Environments that allow a rich user experience, will have to be created, because only then the AmI will bring tangible benefits, to be adopted by the users.

As such, the AmI need a joint effort between users and researchers, and of course through experimentation. And *HomeLab* can be defined as a way to achieve that future. Perhaps in the prospect of set guidelines, to guide the global research in this sector. There are however isolated instances of innovation, often catapulted by technological development, which has been felt extensively in recent decades.

Table 3 presented below, summarizes the characteristics and main features of the previously presented projects.

Project	**Environment/ Objectives**	**Technology/ Key Features**
MavPad	Replica of a small apartment; Provide the environment with intelligence Information-gathering sensors;	bpm
iDorm	Dorm (bed, desk, entertainment system);Assess the learning of rules after the experimental period;	Sensors (motion, temperature, humidity); Automation mechanisms; Integrated agent; Door control; Adjust of heaters; Robot to transport necessities; Incremental Synchronous Learning (ISL); Fuzzy Logic;
Sensor 9K	Middleware Layer; Promote the creation of applications for intelligent environments;	Middleware Layer;
I3A	Prototyping solutions for smart environments; Sensors network (temperature, CO2, humidity);	Individual programmed sensors;
Saves	Intelligent environment; Designed for use in construction and user occupancy profiles	Maintain and regulate a building temperature;
MavHome	Room; Reduce user intervention; Reduce energy consumption;	Sensors; Actuators; Rational hierarchy of agents; CORBA; Four Layers Agents; 72-76% interaction reduction; Room acting as rational agent;
GENIO	Provide homes with ambient intelligence;	Networked appliances (washing machine, oven, refrigerator, sensors, alarms); RFID; Java, VXML, JSP, JavaBeans; Central controller; Voice instructions;
HomeLab	Lab; Test and validate the environmental intelligence theory; Behavioral analysis; Stress assessment;	Computer vision; Shape recognition; Hidden cameras; Collect of images and sound; Luminosity control;

Table 3: AmI Projects review.

2.3.3 Challenges and Trends

As identified at [39], there are several challenges that have somehow stagnant the development of AmI solutions to the pace that is originally expected [97]. The challenges outlined are the following:

- **Privacy, identity and security**, this challenge despite transversal to several areas, currently remains on the agenda with respect to AmI solutions, mainly derived from IoT devices massification. Several authors [51] have analyzed the different projects developed in this area, given its validation in terms of privacy and security issues of the projects. This challenge has been rather disregarded and is not given its due importance, but currently the panorama is changing. And since there are projects like *SWAMI* (Safeguards in a World of Ambient Intelligence) [143] pointing the various scenarios where things can go wrong, and the safety barriers necessary to prevent this happening.

- **Physical restrictions and type of hardware**, as noted at [43] it still be a challenge to develop solutions with higher computational and communication performance, enabling support to the increasing complexity of services. And in parallel, the miniaturization of devices making them smaller, lighter and more efficient.

- **Devices battery life**, in this field there is still much work to do. However, several technologies that have recently emerged have allowed significant savings in this area. What allowed that currently, certain types of devices may not have hours or days of autonomy, but instead years of autonomy. However beyond this fact, increasing the capacity and performance of batteries continues to be investigated, to get a new generation of such components that allow to radically change all other areas having a direct dependency of this type of energy storage.

- **Management of multiple users simultaneously in the same space**, AmI will not exist, if the question of multiple users is not solved, because the concept of intelligence should allow the management of as many users as necessary. A society that is in constant interaction and mobility, and has different preferences, implies a large heterogeneity. A AmI scenario will have to be able to overcome all the constraints that come of these factors.

- **Dissemination of researchers efforts**, as in other types of research, also in the field of AmI, many researchers have done similar research that will be difficult to further use, due to different technologies and methodologies used by each one. Achieve a greater interoperability between researchers, will enable greater effectiveness in leveraging research, contributing to better results.

2.4 A User in a Smart Space

At this section is defined the role of a user in a smart space, the concepts of: habit, behaviour and context-aware.

The role of user in a smart space

Much has been written about the user role in any smart space. And how the smart space should adapt to different users, as well what improvements these spaces introduce in the quality of life of the users who frequent it.

Thus, it is understood that this system proposes a smart space that puts the user at the center. It is thus understood that the space must be fully and constantly adapted to the users who frequent it. And so the user comfort needs must be fully ensured. Because this system is understood to be the superlative of comfort, and there should not be any priority above this, for example, energy savings or other factors are understood to be always secondary. This is the only way to see a smart space, which brings sufficiently significant added value to the present users.

In this way, the user role, in this type of space, is simply to exist, not having to worry about adjustments. Thus, he will only need to move in his daily life and live his life at the different aspects (domestic, professional, leisure). The space management is completely automated, without any intervention or parameterization need.

Also considering user safety, this system will bring peace of mind to its users, who will trust that the system will also take the best decision and control at this level, alerting to any eventuality that may arise.

Habit

It is understood as an abstract and socially defined concept, with no “correct” or “incorrect” definition of habit [139]. The different definitions must be analyzed according to their usefulness in predicting, explaining and changing behavior. The traditional definition of “habit” as frequent, regular or persistent behavior is unsatisfactory because it does not offer any explanatory mechanism for these characteristics.

Next, explicit definitions of habit cited in different literature reviews are presented. These definitions agree on the description of elements of a process by which behavior is contextually stimulated, without conscious thought.

Five definitions portray ”habit” as the behavior generated by this process [55] [56] [95] [114]. One sees the habit as a tendency to engage in behavior [102], and two as the responses automaticity [133] [141].

It is always difficult to gauge the different users habits, and it is even more difficult knowing that they are constantly on the move. With this solution, and the inherent information collection, namely the different places that users frequent, as well the hours at which they attend.

Thus, despite the different users mobility with a relatively large history of information, and the application of an AI model, we can start to have prediction/detection of habits, namely prediction of the users presence in different places.

Behaviour

Behaviour is the central word in many fields, from life sciences, social sciences or psychology. Despite its importance and long-standing research in this field, many works and reference books refrain from defining their central research object [67], [130].

Among the rarity of definitions provided, most are surprisingly imprecise or apply only to certain types of behaviors or species (eg, “everything an animal does and how it does it”; [28]). The lack of a consensual definition is a topic of discussion and has been attributed to the fact that because behavior is so pervasive and intrinsic to everyday life, researchers often end up relying more on their intuitive understanding than on scientific definitions [24] [53] [82].

The lack of a well-established scientific definition of behavior has also contributed to the diversity of methods used to study it. In the field of biology, one relies heavily on observations and technology-based methods to measure and track (mostly animal) behavior. Psychological and social science disciplines, on the other hand, rely heavily on assessment and self-report methods to study human behavior (questionnaires and interviews), while observations are used less frequently.

In the literature, different definitions for behaviour can be found. Including:

- “the internally coordinated responses (actions or inactions) of whole living organisms (individuals or groups) to internal and/or external stimuli, excluding responses more easily understood as developmental changes” (p. 108) [82].

- “the organized entirety of the relationships of the living being and its environment (in the wider sense considering all relationships of whatever nature they may be; in the narrower sense considering only sensory-motor relationships)” (p. 117) [106].

- “those ongoing events of an organism or emanating from an organism that can be externally perceived” [138].

- Behaviour = Identity of the person, Want (motivational parameter), Know (cognitive parameter), Know-How (skill or competency parameter), Performance (procedural aspects such as bodily postures, movements), Achievement (outcome parameter), Personal Characteristics (individual difference parameter), Significance (“what the person is doing by doing the concrete thing he or she is doing”; p. 148) [24].

- “verbal utterances (excluding verbal reports in psychological assessment contexts) or movements that are potentially available to careful observers using normal sensory processes” (p. 372) [53].

Context-Aware

Context-aware computing is a paradigm, where applications and services can use user’s environmental data, like the location and current activity, daytime, and also other users and intelligent devices. As example, this can be seen when the system uses different kind of sensors to user’s activities monitoring [31]. Identifying this aspects (location, objects status and devices) allows the computational infrastructure adaptation to the system and in this way assist the user and show any alert/notification that can be relevant.

Enabling devices and applications that automatically sense and adapt to the changes in near physical and operational environments can enhance the user experience. A context for the interaction between users and devices is created, using environmental data [93].

On the literature, context awareness is defined through two perspectives [93], [84]:

- *Active context awareness*: the applications behaviour is adapted according to the sensed context, as an example when the system detects that the temperature is out of some predefined range;
- *Passive context awareness*: new/updated context information is presented to the users, or the context persists and it will be retrieved later, as an example when the system asks the user for data input to confirm a context.

Intelligent systems are supposed to use external data sources to adapt their behaviour according to this data. Considering specifically context-aware systems, they had to read the data and adapt themselves to any change identified on it, even if the data as some in-correction (incomplete, outdated or ambiguous) [27].

2.5 Synthesis

In this way, several good practices presented in the literature will be replicated, such as the use of sensors, context aware, users detection using non-invasive techniques, among others. The different preferences and the type of data that define them will also be considered, to the proposed architecture.

For this work, behavior is understood as the set of habits that a user has, namely their natural order or sequence or combination of these.

Behaviors can also be understood as the actions that a user has, considering certain conditions that arise such as temporary discomfort, illness, physical exercise (gym, swimming, etc.). It is well known that this type of situation leads to a natural change of behavior on the user part, namely with regard to their comfort preferences.

So understand these types of situations, and how each user reacts to them, adjusting their behavior when they happen. This is, therefore, an entirely relevant knowledge to reach a level of excellence, regarding this thesis topic. For example, knowing that a user, when he was previously in a place of leisure to perform physical exercise (gym, swimming pools, etc.), in a certain period of time after performing the exercise will tend to feel warmer, and therefore the spaces he frequents next must adjust to this context, to avoid manual adjustment by the user. As in illness situations, there may be periods of excessive cold or heat due to fevers, or similar situations, which will lead to a constant adjustment by the user in all spaces that he frequents.

Reaching this user behaviour knowledge level, allows the solution to be at a excellence level, compared to any product on the market, or reviewed in the state of the art.

3

Characterization and Methods

At this chapter the general requirements are explained, the overall architecture and the user behaviour simulation is detailed. Following all the MAS aspects are depicted, as well several points that must be considered, since the data and relevant information, as well the security and privacy issues that these data imply. Finishing with all the important results achieved.

3.1 General Requirements

In the following subsections are detailed the general requirements, namely: the required and relevant data, the user identification process, the detection and characterization of the user at the space and at the environment, and do the distinction between predictive and reactive comfort.

3.1.1 Required and Relevant Data

Any intelligent environment necessarily implies a process of data collection/acquisition. It will be using this information that the environment/space will optimize their decisions in a autonomous and intelligently way.

The existing literature already points out several ways of obtaining data in this kind of spaces. Imperatively the most traditional way on the market are sensors, these allow to collect different kinds of information (temperature, humidity, luminosity, presence, etc.) [16]. These sensors can be isolated, dispersed throughout the environment or integrated into other devices, that users use and carry with them in their daily routine, in a non-invasive way. These sensors are currently in devices such as smart watches, fitness bracelets, smartphones, etc. This IoT ubiquity will enable a constant information reading needed, for use by these intelligent environments. Especially with low power and miniaturized sensors, and technology evolution that allows communication between them and the Internet.

There are currently several research projects that aim to create economically viable sensors for physiological analysis [112], as indicated in table 2 that could be most valuable for a prediction and medical examination of each individual.

This kind of control, was previously only possible in appropriate environments, and using expensive large size devices, which is not allowed in any way, that each person could perform their own control independently and regardless of their technical knowledge.

The capacity of wearable devices to acquire physiological information, is of great relevance. Currently the wearables that are already on the market, collect certain indicators, as previously indicated in table 2, that using health models, already well validated scientifically, make possible to determine user comfort satisfaction. This information is relevant to change comfort conditions of a particular environment. Its use only become possible and practicable recently, since only now technology has enabled miniaturization of this sensor type, and so these are integrated into wearable devices with a commercial value that is accessible to most users.

In addition to these factors, using other fields of study, and in particular concerning indoor location, can be used the dispersion of terrestrial magnetic field, to have very high accuracies in indoors location [129], [127]. This is very important, to get distance between users and actuators that allow comfort conditions (heating, cooling, air quality). With this is possible to optimize operation of these actuators. This information is even more important on large size environments, containing a large number of comfort actuators, where they may be adjusted according to users environment dispersion.

Thus, for this type of systems it is necessary, as well extremely relevant, to identify the necessary information. And that in some way is to be highlighted or even essential for the system functioning.

According to the literature, the preferences that are understood as comfort in the daily life of the human being are well defined [52] [83].

Clearly as essential, temperature and relative humidity are defined. These are the ones that are most present in homes, as these are also the actuators that are most present (heat pump, air conditioning, boiler, etc.).

Other preferences, despite being considered and intervening in human comfort and well-being, are not usually so relevant and considered in the implementation of smart homes currently carried out, perhaps mainly because actuators that automatically control this type of preferences are not so common nowadays.

In particular, the possibility of luminance and brightness are currently beginning to be common in LED applications installed in homes. On the other hand, other preferences defined as multimedia, from the sound volume, to the type of music and favorite playlists. They are currently not so common, due to the fact that actuators allowing this type of control are not yet widespread.

Thus, in section 3.3.1, the preferences card is defined, which includes the different preferences identified in this section, as being important for the user comfort as well for the user and space safety.

3.1.2 Detailed Description

This thesis, particularly this subsection refers several times the space and environment concepts. As such, it is necessary to contextualize them, prior to use.

By *space* it means the *global context where the user is located*, namely the building, housing or public space.

On the other hand, *environment* refers to the *specific location within a given space*, namely the internal division, as the hall, room, or office, being the space naturally wider, and the environment inserted in the space.

The process and learning model proposed to the system is intended to be scientifically innovative, taking advantage of the latest research in this field and combining multiple factors and technologies described below:

- *Context awareness*, as described above since the context is entirely relevant in such systems. In the literature, context sensitive systems are described as members of the ubiquitous computing environments. These systems consider different information related to space, environment, resources, users and the relationship between each of them. They intend to get personalized decisions depending on contextual factors. In this way, we can have different decisions, to the same situation in different contexts [122]. The contextualization elements can be obtained in several ways as described in [19].

- *Use of sensors information* combined with machine learning techniques, including *Sequence Discovery*, *Fuzzy Logic*, *Genetic Programming*, *Multi-Layer Perceptron*, as described in [135], get habits information of the users present at the environment.

- *Use of logical sensors*, there are three types of sensors used to assess the context in such systems: physical, virtual and logical. Physical sensors capture the context of the information, and are dispersed throughout the environment. Virtual sensors are specified and configured to collect contextual information using as sources applications or services. Logical sensors, through the combination of physical and virtual sensors, intended to determine logical values for the attributes to be collected. For a smart environment contextualization, the attributes are typically collected using the three types of sensors depending on the contextual nature within the smart environments.

- *Use dynamically scaled priorities rules*, which must have the information considered essential for the correct functioning of the system, including the limits for the different parameters, like system reliability or user safety.

- *Use of MAS*, representing the different entities involved in the negotiation process, allowing an efficient outcome under different situations.

That said, the practical applicability of this thesis is the specification of an IE, namely to ensure a solution that covers the following features:

- User identification;

- Detection and characterization of the user at the space;
- Detection and characterization of the user in the environment;
- Preferences conflict management in the environment;

All these points can be seen in detail at the following subsections.

3.1.3 User identification

In this thesis, the user identification is one of the essential tasks and it will be analyzed carefully later. However in a first approach, there are two situations, explained below:

- *User ID sharing:* in this situation, when the user enters in the environment, the devices that are with him (smartphone, wearables, etc.) pass the user ID to the system that controls the environment. The system validates the ID in the Cloud, and from this it will get user's preference card, this concept is detailed at subsection 3.3.3. The system will then use the received card information, to adapt the environmental comfort conditions, using the automation available in the environment. In this case the system must be permanently connected to the Internet, so that is allowed access to the cloud.
- *User preferences card sharing:* In this case the user enter the environment, and share directly with the system its preferences card, using the card available in his compatible device (smartphone, wearable, etc.).

  The system collects data from these preferences and adapts, as in the previous case, the environment comfort conditions, using for that purpose the automated systems available in the environment. In this case, the system does not require an Internet connection, and all the process may be performed offline.

Both situations assume that the user has no part in the process, and is completely transparent to him. The use case diagram present in figure 37, illustrate the operating modes provided for the implementation of user detection process, and sharing of his preferences card with the environment system.

Initially are defined two use cases, depending on system network connection status. For systems located in environments where there is no cloud connection, or where system is offline, the user device share the preferences card directly with the system. Continuously the system proceeds to the environment adjustment according to existing needs and different actuators on the environment. In case of missing actuators, the respective shared preference will be discarded.

The other case implies a permanent system connection to the cloud. In this situation, the user does not directly share his preferences card, but his user ID, and this ID is validated in the cloud.

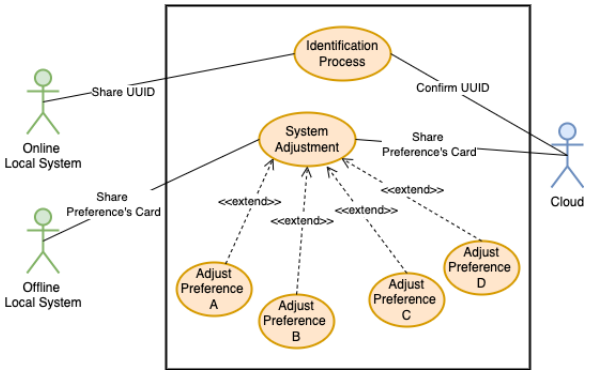


Figure 37: AmI System - Use Case diagram.

Upon successful validation carried out by the system, the user preferences card will be returned from the cloud. After a correct reception by the system, it will ensure environment management in the same way mentioned above.

In figure 38, an example of an environment is illustrated to demonstrate the use cases described above. Note that the communication processes, represented in this figure on arrow format, are expected to be user transparent and completely independent of its intervention.

After the user ID cloud validation, the respective preferences card is downloaded into the local system, and the control is made automatically by the local system, adjusting all the preferences existing in the environment.

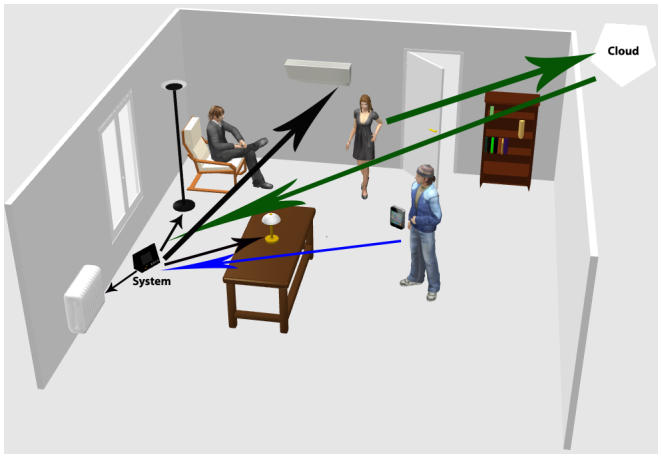


Figure 38: AmI System - Communication process.

It is important to highlight that it is necessary to do this user identification process at a periodic basis, not only to identify new users but also to detect when users leave the environment. Also, this periodic time can't be too low (few minutes), to avoid the user detection, when it only enters and leaves the environment after few minutes, because in this kind of situations it isn't expected to actuate.

Also the periodic time can't be too high, to detect all the users that stay some time on the environment and leave after. It was decided to use a *30 minutes* period, with this, the system must do a full users

detection, *48 times* per day. This number seems to be a good balance to surpass the identified situations described above.

3.1.4 Detection and characterization of the user at the space

In their daily lives, user moves through different spaces and situations where although there are routines, it becomes difficult to identify accurately and efficiently which are their movements and location. In this sense, having real-time user's location within a particular space, is an extreme valuable information regarding environment optimization using forecast methods, as explained in detail at subsection 3.1.7.

In the past, this information was just captured using presence sensors disposed statically, and had several gaps, including the existence of many points where they did not obtain information (blind spots). Its fixed nature, constant maintenance, usually restricted to specific areas where user identification was a critical need, is no longer sufficient for the current people mobility reality and the need for more ubiquitous solutions [16].

The limitations of motion sensor technology together with the understanding that to actually be smart, a smart home needs to be able to detect who is present, lead us to the inevitable conclusion. For intelligent smart home automation, a different presence detection solution is needed.

In figure 39, is illustrated an example of a context as described above. This figure shows a space divided in different environments. It can be seen, that naturally users move between environments, and so there may be environments with one or more users, as well empty environments.

Collect and evaluate this type of information is relevant, to optimize space conditions in general and of the environment in particular.

Figure 39: Detection and characterization of the user at the space.

3.1.5 Detection and characterization of the user in the environment

In addition to user's location in space, described above. It may also be extremely important to get specific user location within the environment. Because this distance will have influence, especially in the specification of their comfort conditions, allowing to know the distance among the heating/cooling sources, as well other equipment's available in the environment that interfere with comfort conditions.

With this information, response and configuration of these devices can be adjusted in real time to increase effectiveness performance, both in terms of service quality, such as energy efficiency, which will result in an obvious and desired energy costs reduction for the final consumer.

This type of indoors proximity information was previously technologically unfeasible, especially due to the costs of implementing a solution like that.

Currently existing sensors type and technologies, particularly in terms of BLE [25], are fully market implemented and at reduced cost to the user, which enables the optimization in a relevant way, and overcome the type of problems identified above.

In figure 40, is illustrated a context example as described above. This describes an environment example, where there are two users, and the proposed system should identify them.

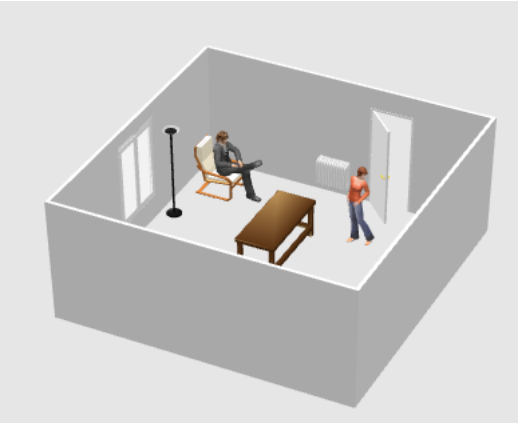


Figure 40: Detection and characterization of the user in the environment.

3.1.6 Preferences conflict management in the environment

Another problem that stays current in research is the conflict preferences management, which is also applicable in the AmI field.

It is known that each person is unique in its individuality, and this naturally translates in his comfort preferences, that also vary with the physiological and physical conditions of each one. Namely there may be incompatibilities between the comfort preferences of different users that are in the same environment.

To overcome incompatibility, it must exist obvious compromises between the different stakeholders preferences. This process should also be completely independent and transparent to users.

In the conflict management field, there are currently several work undergoing, most of it through the use of agents that carry out negotiations for potential conflicts resolution [90].

In figure 41, is illustrated an example of context as described above. Can be verified an environment where three users are present, which will invariably have different preferences. What will lead the system to act in order to overcome the best of these differences, as explained in detail at section 3.4.4.

Figure 41: Preferences conflict management in the environment.

3.1.7 Comfort - Predictive vs. Reactive

Especially regarding space temperature and humidity, there are different factors that influence the actuators performance to achieve the expected comfort conditions. Such as the interior and exterior space temperature, area, coating materials, among others, there is a period of inertia dependent on all these factors. So all this is considered when planning the space to achieve the desired comfort, as all of them will be a fundamental factor to consider when defining the inertia period, that is, the time period since a temperature is requested until it is effectively reached in space.

In addition to these factors, actuators also have different inertia times associated with their operation mode, it is well defined that, for example, underfloor heating systems have a significantly higher inertia value compared to air conditioning or fan coil systems. Because in this type of systems, in a normal space area, for the ascent of 3 to 4 degrees, we may need time periods in the order of 10 to 20 hours. In other words, it is clear how central this issue of inertia becomes, when we are talking about providing the user with immediate comfort.

So to overcome this period of time that goes from the temperature request until the space actually reaches that temperature, there are thus two different methodologies to achieve comfort: reactive and predictive.

- *Reactive*: that is, the one that only reacts after the user is present in the space, and only then the different actuators begin the process of adjusting to the user's preferences. So in this case, it

is natural that depending on the actuators present and the different factors previously identified, there is a significant latency. Which can lead to the fact that in certain cases the space does not even reach the desired conditions during the period of time that the user remains in it. In other words, this is a very ineffective process, as the user ends up not feeling comfortable during the most or the entire time he remains in the space.

- *Predictive*: this methodology emerged to overcome the inefficiency of the previous method, especially with the use of more passive actuators. The operation method of these involves prior knowledge of the times and temperatures expected for these times, and thus start to act for the necessary time before the presence of users, and so at the time of their presence the desired conditions are already achieved. This type of methodology identifies the reaction time necessary to reach a temperature differential between the current and the predicted one, and thus starts its operation with the number of hours necessary to reach the predicted temperature.

In this thesis, the reactive methodology is explored, however with the solution and model implemented, the way is open for an easy evolution to the predictive methodology, with all the advantages that it has. With the times and places information at which the users are present, it will be possible to evolve to the space predictive adjustment. This can be seen as a future work to be developed, as detailed at section 5.7.

3.2 User Behaviour Simulation

In this field some work was already done [30], [147], [68], [144]. This work it was been evaluated and improved, developing a more focused solution.

Pursuing this effort, several information of hundreds users is needed to test a MAS that simulates these users behaviour, like in most research projects carried out, data are usually needed to simulate behaviour and efficiency of the proposed solutions, since it is necessary to use them to validate and test research carried out at the most diverse levels.

The development of this simulation also arises from need to gather information on multiple users comfort preferences (temperature, humidity, musical playlist, musical genre, etc.). And besides that, also have information of how each user adapts their preferences to the place where he is [122]. This information, in addition to being necessary on a large scale (hundreds of users), would also be necessary in a very broad time-frame, always longer than one year. Because is known that comfort preferences normally vary according to the seasons.

Getting data with this dimension and involving so many users is a difficult task and, in addition to users collaboration, would require a high cost, regarding the equipment needed to collect this information [135].

Thus an algorithm was created, which simulates not only different users preferences variation, but also their daily life, considering the different places that user frequents (home, work, leisure places). In addition, relationship between users is also established, introducing the family and co-workers concept.

Using this kind of solution, also all security and privacy problems are overcome, without data collection from real users is needed [18].

3.2.1 Simulation Algorithm

The developed simulation algorithm has several predefined assumptions that allow the simulation to be as close to reality as possible. These assumptions are defined in the code as input variables and customized according to the simulation needs.

Types of schedule are defined, considering the different possibilities, in this case four, and depending on selected type of schedule, values for minimum and maximum delay are also defined.

Table 4 defines the four delays to different schedules, in this table we have minimum and maximum delay time, that the algorithm uses to generate random between these limits, and to introduce the generated value in each situation.

Parameter (Delay)	**Minimum (minutes)**	**Maximum (minutes)**
Enter Work	-10	10
Enter Home	-30	30
Exit Home	-40	40
Leisure Hours	-90	90

Table 4: Parameters and different delays.

The diagram presented at figure 42 illustrates more clearly the different processes that the algorithm executes.

The process is started by choosing the number of local systems associated with the user, between 1 and 5, depending on this choice the user is associated with the local systems defined. Consequently when selected the local system associated to the home, users are generated between (0-3) that define the concept of family. Regarding the local system associated with the workplace, co-workers are generated between (0-3) associated with the same workplace. Then, for each generated user, the corresponding time type is defined, so when the introduction of user history for each day is started, it is coherently associated with a type of schedule.

Then the process of entering history information for each user, it is started and is carried out consecutively for each day. Thus during the daily input process, the corresponding time of introduction and delay is generated, considering the limits previously defined for this delay, presented at table 4. For each daily period the introduction is performed, and new delay random intervals are defined relative to the time set as standard, in order to get close as possible to users daily reality.

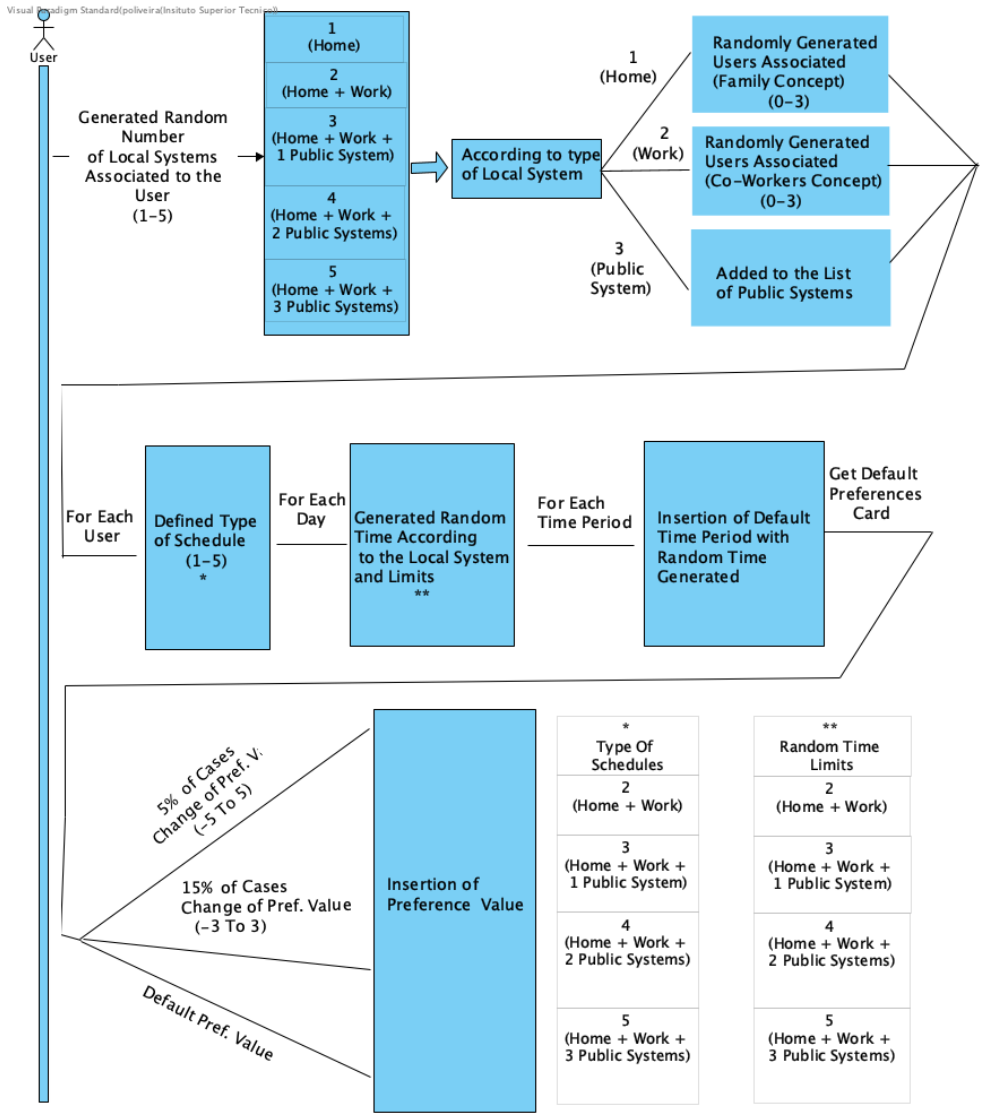


Figure 42: Diagram - Users behaviour simulation workflow.

In addition to the time, and for each period, are also entered the preferences for the correspondent user. In this case at table 5, the following premises were defined:

- In 5% of the situations preference value is changed, introducing in this case the randomness factor with the limits (-5; 5).
- For 15% of the situations, preference value is changed introducing the randomness factor with the limits (-3; 3).
- In all other situations, the preference value is entered without any change.

In this way, information required for this thesis is obtained using this simulation, with many users as necessary, as well for the necessary time period, in this case a *720 days* time window was defined, with *1000 users* in the system, which originates more than *three millions historical records* in the database.

% of Cases	Randomness Factor	
•	Minimum	Maximum
75%	•	•
15%	-3	3
5%	-5	5

Table 5: Randomness factor.

This process takes few hours, namely for the values defined before, for instance in an average computer it takes three to four hours to generate and insert at the database.

Next, at table 6, the defined parameters for the four schedules are presented, these schedules are as real as possible, and with that are obtained the most regular work shifts, like starting work at 08h or 09h of the morning, start at 16h in the afternoon, or work by night that starts at 0h. The most normal period of work (eight hours) are set, and with that is possible to define correspondent hours of exit home to go work, enter home hour after work, and also exit home to do some leisure activities at some social space, that is related to the period that user is not working.

Parameter (Hour)/ Schedule	A	B	C	D
Enter Work	08h00	09h00	16h00	00h00
Enter Home	17h10	18h10	00h10	08h10
Exit Home	07h50	08h50	15h50	23h50
Enter Lazer	20h00	21h00	10h00	15h00

Table 6: Parameters/Schedules.

Listing 3.1 demonstrates a small algorithm part, in the case where the system type is selected. At appendix B all the code is presented.

Listing 3.1: User Behaviour Simulation - System type selection.

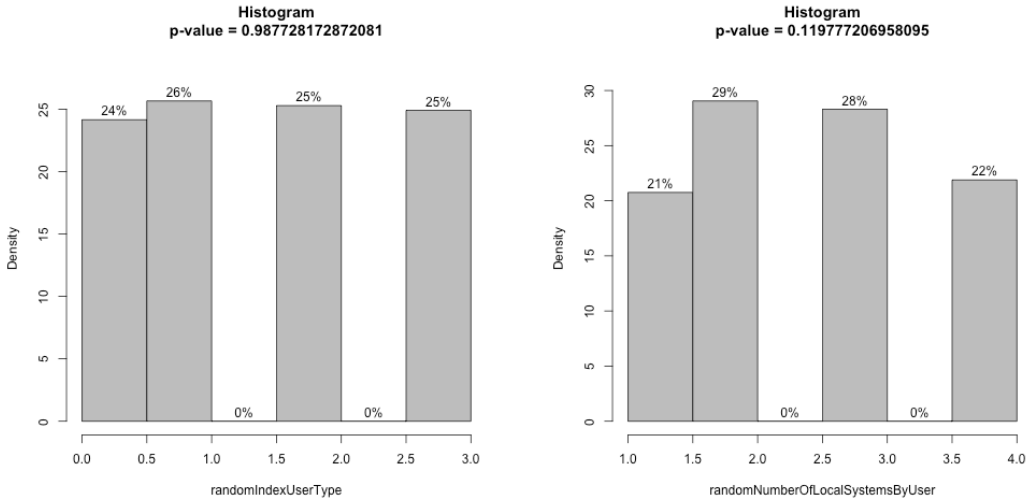
```
1  switch (randomNumberOfLocalSystemsByUser) {
2  case 1:
3  TypeOfSystem = 1;
4  System.out.println("Home");
5  idLocalSystem++;
6  addMoreUsersToSystem(idMasterUser, TypeOfSystem);
7  DescLocalSystem = ("Home: "+idMasterUser+"+"+idsUsersAtSystem);
8  genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
9  genenerateUserSimul.insertHistoryRecord(idMasterUser,idsUsersAtSystem,
10 idLocalSystem,TypeOfSystem);
11 break;
12 case 2:
13 ...
```

3.2.2 Validation and Statistical Analysis

The statistical analysis to support the work presented at section 3.2.1 is detailed in this section. For the simulation results validation, the frequency distribution analysis is used, using histograms, that is the dataset graphical representation in columns/bars previously tabulated and divided into classes [78].

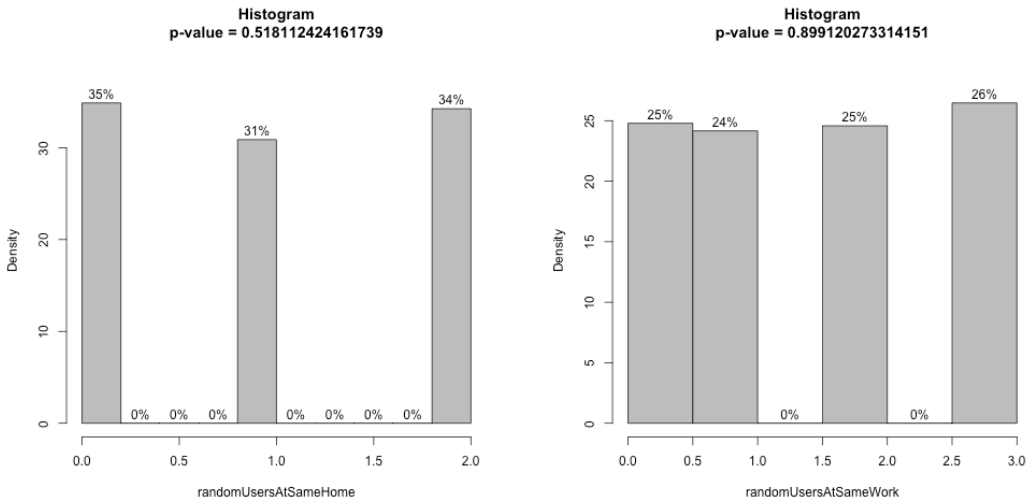
In this case for the different classes, each density is shown, as well the percentage, which it represents within each distribution.

As can be verified in the histogram analysis in figure 43 and 44, it can be concluded that the different classes are evenly distributed.



(a) User Type Selection

(b) Nr. of Local Systems by User



(c) Nr. of Users at same Home

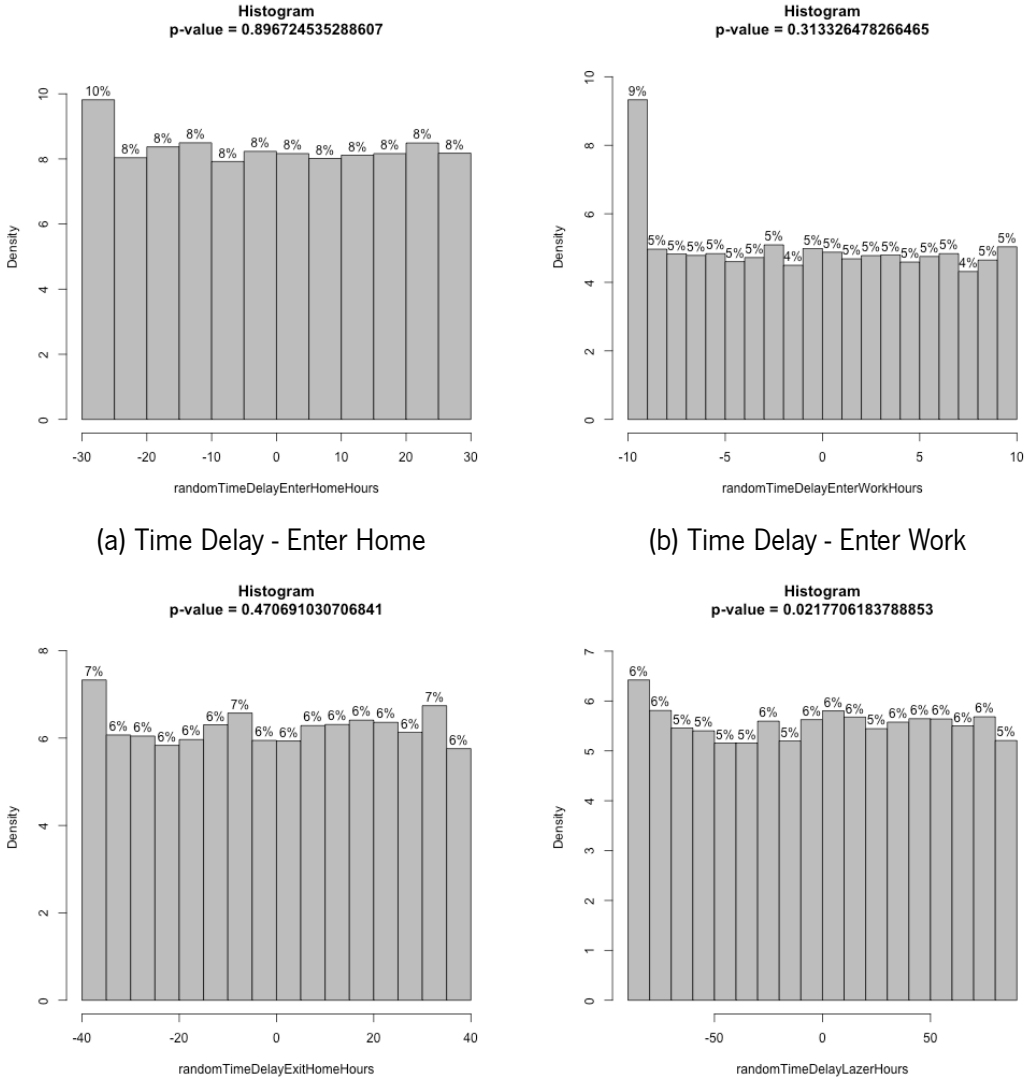
(d) Nr. of Users at same Work

Figure 43: Histogram - User Parameters.

Describing figure 43, in (a) is show the type of user (0-3), at (b) the number of local systems for each user (0-4), in (c) the number of users at the same home (0-2) (family concept) and at (d) the number of users at the same work (0-3) (co-workers concept).

In all histograms for each distribution, hypothetical frequency called *P-value* is calculated, also known as "observed significance level" [34], and also validate that all percentages density are very close, what confirm a evenly distribution.

Describing figure 44, in (a) is show the type of user (0-3), at (b) the number of local systems for each user (0-4), in (c) the number of users at the same home (0-2) (family concept) and at (d) the number of users at the same work (0-3) (co-workers concept).



(a) Time Delay - Enter Home

(b) Time Delay - Enter Work

(c) Time Delay - Exit Home

(d) Time Delay - Lazer Hours

Figure 44: Histogram - Time Delays.

With this work, a fully operational solution can be used to simulate how many users are needed, in any time period and with the comfort preferences needed for the problem.

In this case the information generated is used to test and improve the MAS, it depends from user information, namely behaviour and comfort preferences but also can be adapted to simulate other type of information.

This simulation algorithm can be generalized, as much to accommodate many cases as possible, and with that, the user only needs to define and configure the initial parameters for each case, and after retrieves how many data as he needs for the specific case defined.

Listing 3.2 demonstrates a small part of the code used to validate this simulation. At appendix C all the code is presented.

Listing 3.2: Code - Validation and Statistical Analysis .

```
histPercent <- function(dataTable, labelPvalue, colLabelName) {
  IntDataTable <- as.numeric(dataTable)
  #breaks=min(dataTable):max(dataTable)
  H <- hist(IntDataTable, plot=FALSE)
  H\$density <- with(H, 100 * density* diff(breaks)[1])
  labs <- paste(round(H\$density), "\%", sep="")
  labTitle<- paste("Histogram\n", labelPvalue, sep="")
  plot(H,freq = FALSE, col=colors, labels = labs, main=labTitle, xlab=colLabelName,
      ↪ ylim=c(0, 1.08*max(H\$density)))
}
  ...
```

3.3 Overall System Scenario

In the following subsections is detailed the overall system scenario namely: the preferences card, the system information and constraints and the system architecture.

3.3.1 Preference's Card

Each person has their comfort preferences and, of course, these vary with a number of physiological and other user intrinsic factors, as well the time of day, mood, etc. Also each person as an individual, has different comfort preferences depending on situations. In this way we have a number of unpredictable situations.

This requires that each person, whenever interacts with a new environment, has to perform a manual configuration of it, which is the current form that user has to adjust environment to his needs.

This being the automation era, also in this context there is a need to automate decision processes, given that this type of situation happens in people's daily lives. And this improvement, in addition to comfort increase, bring a number of added value, such as people time savings, economic savings as a result of energy efficiency increase, as well savings in equipment maintenance.

Operationalize this situation requires the definition of a parameters set, which are deemed essential to personal comfort. This set will be called user preference's card. This will be populated with relevant information, which is expected that can be updated at any time if the user wants to change his preferences.

This card includes a comprehensive number of preferences, but not all of them are applicable in all environments, because lack of compatible equipment's on the environment, or even because the environment configuration is not applicable. In this case, preferences not supported by the environment system are discarded.

As shown at subsection 2.2.3, several technologies support this implementation and allow the communication process between user devices and the environment system. These technologies are in a relatively stable state, but still in an evolutionary process to obtain better performances at different aspects (data transmission rates, range, consumption).

It was analyzed the feasibility of using NFC [136], BLE [25] and WIFI Direct [29] technologies. Not discarding the use of several simultaneously, if they complement each other in different situations. It is clear that different equipment manufacturers choose different technologies, and managing to cover a greater number of technologies compatible with the system, means reaching a larger number of possible users.

The preferences card will initially have the characteristics defined at table 7, since such are currently the most common in user's daily life. It is however a dynamic and scalable card, according to the required needs.

All this preferences can at any time be adjusted by the user, using the smartphone application, this preferences are stored locally, and also at cloud. Naturally, it was expected that the preferences card it will be very stable, and any adjustment by the user, it will be punctual.

Preference	Example	Units
Temperature	20	°C/°F
Luminance	30	Lux (lx)
Brightness	80	Watts/cm2
Relative Humidity	45	%
Sound	15	dBm
Musical Genre	Pop	-
Musical Playlist	(Adele, U2)	-

Table 7: Preference's Card example.

3.3.2 System Information and Constrains

On the other hand, as mentioned in section 1.3, the system will also have to provide a range of information, including for security issues, which enable to alert wherever the reference security values taken as safe are exceeded. At table 8 are represented the reference values for the most common gases that would jeopardize the users safety inside closed environments.

Also, to have the temporal and space dimension context represented in figure 2, it is also stored the date, time, indoor location, and physical local GPS coordinates. All this information, obviously refers to

the location where the system is installed. Table 8 exemplifies the information that will be available in the system. This information is scalable and adaptable to specific requirements.

Characteristic	Example	Units
Name	Local_Bedroom_GF	-
Time	20:00	Hours
Date	20/11/2015	Date
Indoor location	Bedroom_GF	-
GPS coordinates	39,399872, -8,224454	-
CO2	500	ppmv
CO	2	ppmv

Table 8: System Information example.

Different presence, temperature, luminosity and humidity sensors were inserted in this system. Some of them include all features, like some sensors that use *ZigBee* [58] [61] communication technology as shown in Figure 46. In this way, all sensors collected information is passed through *ZigBee* to a receiver that is connected to a *Raspberry* representing the local system, as shown in Figure 45. Then all information is stored on a database, and then can be used by different agents present in the system.

It is well known that security problems have relevant importance, particularly regarding intelligent environments, and all what can interact with user security. Also in this field, which deals with configuration parameters for user welfare, regarding different valences. It should be noted the importance to maintain these systems safety, since it is known that temperature and humidity values can impact well-being and even users health. Thus, it is necessary to define all actuators parameters, regarding security issues, especially maximum and minimum constraints values.

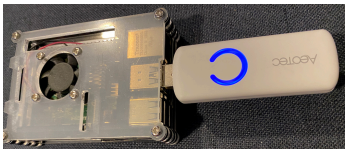


Figure 45: Raspberry.

Figure 46: ZigBee Sensor.

All these values are configured on local systems, and then used by agents in the decision-making process. These values have a high priority, and will work as restrictions, to ensure actuators equipment safety, but mainly concerned to users safety.

Table 9 identifies defined preference constraints, these are always necessary for a correct private or public environment balance. A maximum value is defined for each preference, and the increment/decrement change range that can be performed. This table will be customized according to each location,

and in the case of public places, it may have different restrictions, resulting from the specific environment of each space. In this way it is guaranteed the space and equipment's safety. These validations are guaranteed in the logical layer, depending of each negotiation agent results, before the result sent to actuators.

Preference/ Constrain	Minimum Value	Maximum Value	Change Range	Units
Temperature	15	28	+- 0.5	°C/°F
Luminance	0	40	+- 2	Lux (lx)
Brightness	0	100	+- 2	Watts/cm2
Relative Humidity	20	80	+- 5	%
Sound	0	30	+- 1	dBm

Table 9: System Preferences constraints example.

3.3.3 System Architecture

Figure 47, shows the developed architecture. Explaining this figure, it can be seen the user, that through of its different devices (smartphone, wearable, or other compatible) communicates with the system, and for that different technologies can be used (Wi-Fi Direct, NFC, BLE).

Next, the system performs the information synchronization/share with the Cloud, to validate the information. And then the system will perform the different components management in the environment (climatization systems, security systems, other smart systems).

Figure 48 shows an environment scenario around which the present work was developed. Explaining this architecture, first it can be seen one user who enters at the environment with his smartphone, and a *Local System* is present at the environment. In an autonomous way the smartphone communicates with the *Local System*, for this process different technologies can be used, like: Wi-Fi, NFC or BLE, and transmits the Universally Unique Identifier (UUID) to the *Local System* and this validates the UUID in the cloud, and after a correct match the respective preference's card is retrieved by the *Local System*, also at figure 37 the use case diagram details all the architecture.

With that information Local System start the negotiation process, as is in the schema at figure 52, when the negotiation is concluded, Local System has the final parameters to apply in the different actuators that are present at the environment like energy, security, multimedia, and other smart systems.

Continuing figure 48 explanation, a second user enters the environment, when the 30 minutes period analysis process occurs, the same process of user validation and retrieve of preferences card and the negotiation process, and possible conflict management is done at Local System. Finalized the negotiation process, new result parameters will be achieved and in the same way the parameters will be sent to the different actuators present.

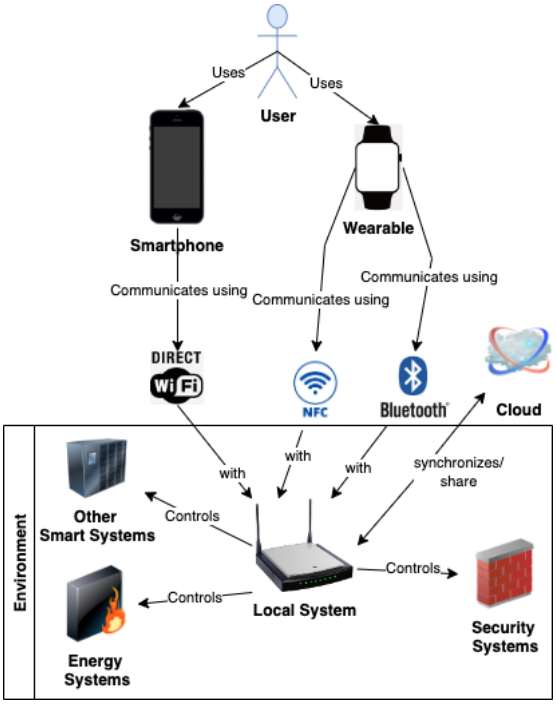


Figure 47: Overall System Architecture.

This analysis process will *automatic occurs every 30 minutes period*, and will detect all the new users that arrive at the environment, and in that way the environment always have the best comfort conditions to match all users preferences that are on the environment at a determined time.

The entire framework/architecture was developed in order to be easily scalable, dynamically, and with no need to change. In this sense, at the users level, the increase of these is carried out in a transparent way through the application installation on their device and initial configuration of their comfort preferences card, from that moment, it will be automatically integrated into the system.

Regarding performance, after verification it was also clearly achieved, as the application present in local systems (*Raspberry's* are used, to achieve a low cost solution, that has all the necessary requirements, to support the proposed work), and which controls all user entries in different locations, easily and quickly allows the communication to the server's.

The MAS solution performance, as it is critical and clearly lacks more computational power, will always be performed on the application server, and the results will be immediately communicated to each local system, so that they can be applied to each of the actuators present in the space. In this way, whenever it is necessary to increase performance for the MAS solution processing, it will also be possible and quick to scale the existing servers.

On figure 49 we can see how the system reacts to the user movement, and it´s practical functioning at a smart environment. Three distinct moments in time, for the same environment, are shown. At period 1 (18h00) user 1, 2 and 3 are on the environment and the two actuators adjust to their preferences. Between period 1 and 2, user 4 enters the environment, in that way at the beginning of period 2 (18h30) the environment adjusts to the new preferences to now satisfy the four present users. Between period 2

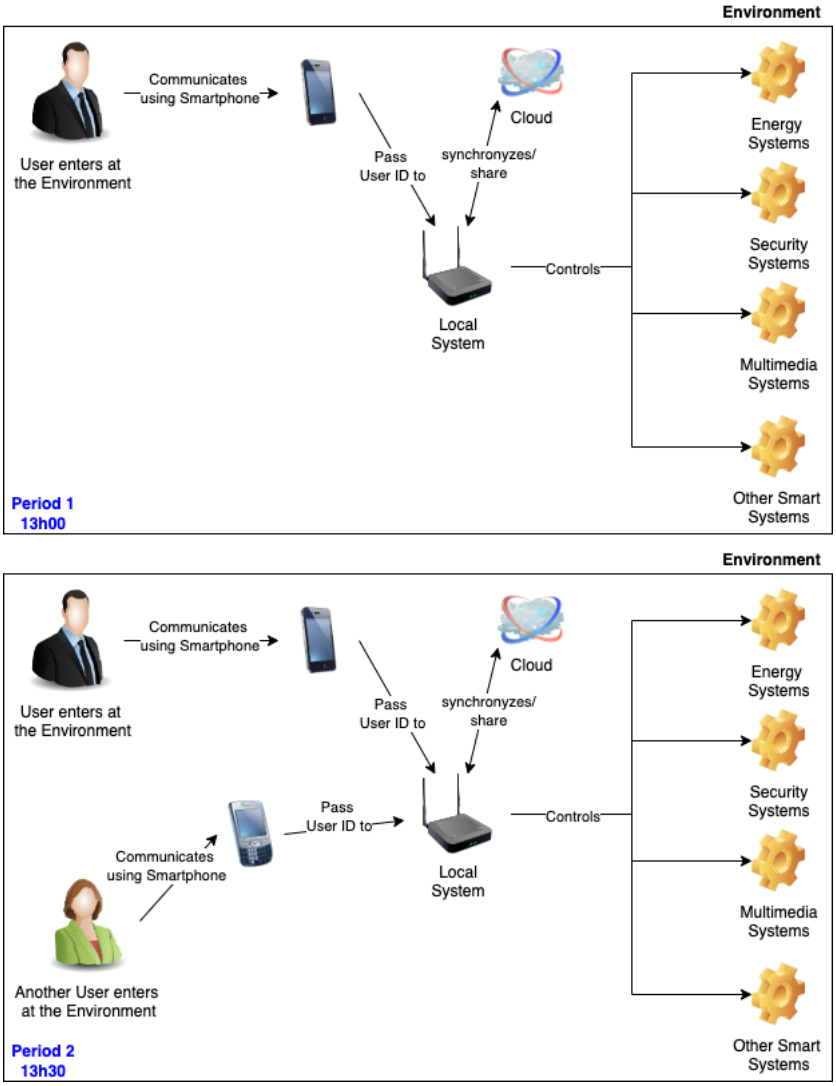


Figure 48: Environment scenario.

and 3, user 1 leave's the environment, in that way at the beginning of period 3 (19h00) the environment readjusts again to now satisfy the three present users preferences.

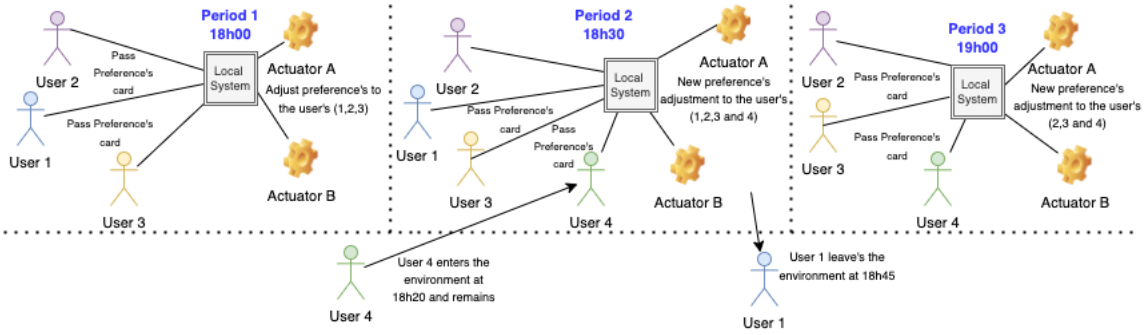


Figure 49: Smart environment functioning example.

3.4 AUPBH Framework

This section describes the proposed and developed framework, that was named: Aware of users, preferences, behaviours and habits (AUPBH). The following subsections details the used technologies, the MAS architecture and schema and the developed work related to conflicts management.

3.4.1 Introduction

The AI field continues with an exponential growth rate, and multi-agent systems have been used to solve several situations, related to AmI.

AmI is an ubiquitous, electronic and intelligent environment, recognized by different technologies/systems interconnection, in order to carry out different daily tasks in a transparent and autonomous way for the user [33]. Thus, multi-agent systems are made up of autonomous agents present in the environment and who have the ability to make decisions derived from interpreted stimulus and connection with other agents, to achieve common goals [142].

Currently there are different languages and platforms for the developments of this systems types, namely *3APL*, *Jack*, *Jade/Jadex*, *Jason*, among others [26]. *3APL*, *Jadex* and *Jason* use agents with cognitive reasoning models as an alternative to more traditional reactive models. Focusing on Belief-Desire-Intention (BDI) cognitive model, which allows the creation of intelligent agents capable of making decisions based on beliefs and perceptions, desires and intentions that the agent may have at a given moment.

There are already different literature works that present solutions for integrating MAS with AmI, and specifically with Smart Homes, using *Jade* [74], [23], which is reactive, and using *Jason* with *JaCaMo* [88], [89], [10].

Jason is a framework with its own language for the development of cognitive MAS, and using the customized *ARGO* agents architecture it is possible to bridge the gap between MAS and actuators/sensors present in a real scenario. It as an agent-oriented programming language, has an *AgentSpeak* Java interpreter for the development of intelligent cognitive agents using BDI.

The BDI consists of three basic constructions, following detailed:

- *Beliefs*, are information taken as truths for the agent, which can be internal, acquired through the relationship with other agents or through the perceptions observed in the environment;
- *Desires*, represent an agent's motivation to achieve a certain objective;
- *Intentions*, are the actions that the agent has committed to perform.

AgentSpeak is a programming language focused on the agent approach, which is based on principles of the BDI architecture. In addition to these concepts, the Procedural Reasoning System (PRS) allows the agent to build a real time reasoning system for performing complex tasks. *Jason's* agents have

a reasoning cycle based on events that are generated from capturing perceptions of the environment, messages exchanged with other agents and through their own conclusions based on their reasoning.

These events can be triggered using triggers that lead to the execution of plans (available in specific libraries) composed of several actions. Jason's agents are programmed based on the definition of objectives, intentions, beliefs, plans and actions internal to the agent and actions performed in the environment. A MAS in *Jason* does not traditionally have an interface for capturing perceptions directly from the real world using sensors. Because *Jason* only uses a simulated environment, for that fact the *ARGO* custom architecture is used for this.

A MAS using *Jason* and *ARGO* can be made up of traditional *Jason* and *ARGO* agents that work simultaneously. *Jason* agents can carry out plans and actions only at software level and communicate with other agents in the system (including *ARGO* agents). On the other hand an *ARGO* agent, is a traditional agent with additional characteristics, such as, for instance, the ability to communicate with the physical environment, perceive and modify it, and also filter the perceived information.

This thesis, also propose an autonomous Smart Home model controlled by cognitive agents using *Jason* framework and *ARGO* architecture to manage physical devices, since *ARGO* agents allow communication with different controllers like *Arduino* or *Raspberry* [126]. For this implementation, different usage scenarios are developed and detailed at section 4.2, each with the different systems actuators to support the preferences detailed at subsection 3.3.1.

To evaluate the MAS development, a series of performance tests were done considering parameters such as the number of agents, number of controllers, agents speed of reasoning, environment perception moment and information filtering, in order to explore different system implementation strategies. As well the user behaviour simulation described at section 3.2. And also a user satisfaction and energy consumption analysis is done and detailed at section 4.4.

To optimize the proposed solution predictions, a MAS architecture was defined. The roles that each agent should represent, as well the negotiation process to be taken, the different scenarios in which this negotiation should take place and the way it should be processed were specified at subsection 3.4.2. For the implementation development, two phases are defined as follows:

- Hardware (*Local System's*) installation;
- MAS development;

First, the entire physical structure must be prepared, where the local devices (*Raspberry*) equipped with the network technologies previously identified at subsection 3.3.3 after this preparation they can detect the users inside the space, detecting the smartphone or wearable devices of each user.

A prototype was thus implemented in a domestic house, considering all the MAS architecture detailed at subsection 3.4.2 and system actuators present in the house and detailed at subsection 4.2.3. For this purpose, a *Raspberry* is used per division, in this case three on the ground floor (living room/kitchen, office, bedroom) and three on the first floor (one at each environment), all this is detailed at section 4.2.

Regarding the temperature actuators, these divisions have a hydraulic radiant floor heating system supported by a heat pump and also fan coils, and a home automation system that controls the different actuators detailed at subsection 4.2.3.

To a more detailed view, a 3D model was designed, where the system operation for a specific space can be visualized, like can be seen in figure 50. Explaining the model, we can see different people present in the space, as well the present *Local System*, the different arrows illustrate the autonomous communication process between user's peripherals (smartphones) and the *Local System* and also the communication with the central server (Cloud), which will allow to have the needed information for the MAS work and in that way reach the optimum comfort preferences values to use in the actuators.

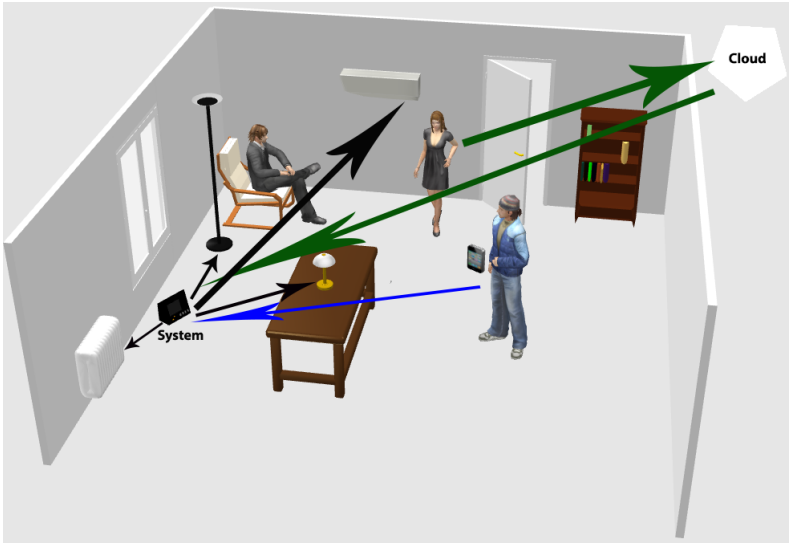


Figure 50: Example of the System in a environment.

For the MAS performance evaluation, are considered the following parameters:

- Number of agents used;
- Agent speed reasoning;
- Information filtering;
- Environment perception time.

The user satisfaction and energy consumption analysis is done and detailed at section 4.4. All the MAS implementation details are described in the following subsections.

3.4.2 Multi-Agent System Architecture

On figure 51 the proposed MAS architecture specification is show, the different modules are separated, to easily identify the purpose of each one, the agents containing it and its purpose are also detailed. Following at subsection 3.4.3 the MAS schema is described.

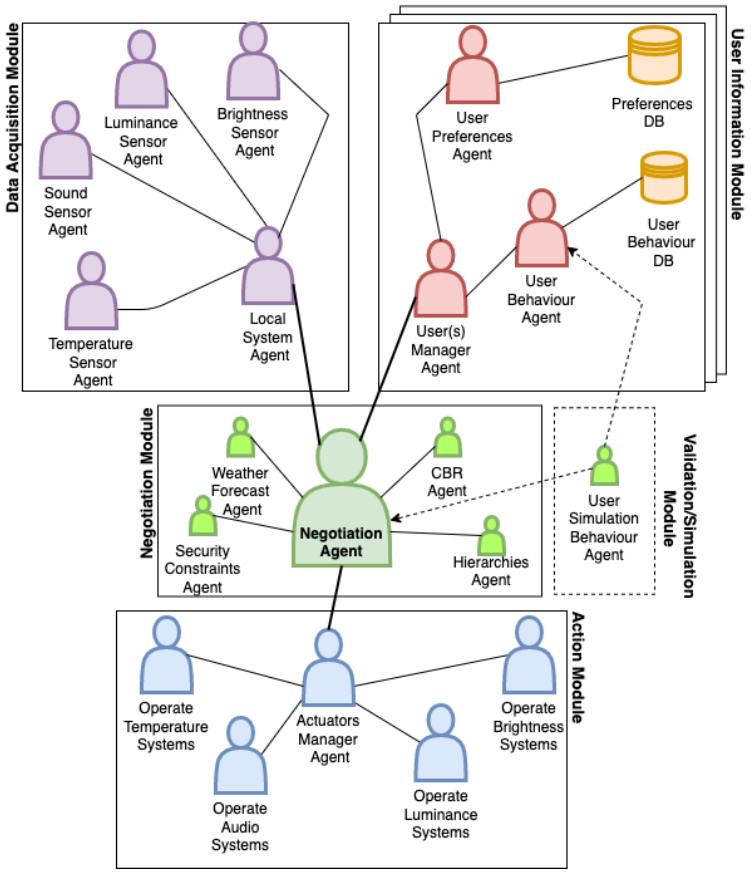


Figure 51: MAS architecture.

Next, the MAS modules are fully described:

- **Data acquisition module**, includes two agents types: the sensor agents will import necessary information from the sensors (temperature, luminance, brightness, sound) for the agents operation; the local system agent has all the system information for each local as detailed at table 8, and obtains the present users at each period (30 minutes).

  There will be one principal agent who will represent the local system, namely each individual environment, where it has the need to ensure individualized comfort conditions, such as a room inside a house, or a office inside a building. This agent will consider any directives that may exist for this environment, such as lower or upper limits to different comfort conditions, or also safety parameters that may be critical for a given space. This agent will have a obviously prevalence relative to others, since it will be the dominant for a given environment.

- **User information module**, includes three agents types: the user preferences agent that will represent each user preferences that must be used in the negotiation process; the user behaviour agent that will have the user history information, to support forecast solutions; the manager agent,

that will combine all the information and it will pass this to the negotiation agent, that will represent each use at the negotiation module.

Regarding the users, each one will be represented by an agent, this will receive user preferences from main system, for the place where it is, as well for the time in which it is. Also in this situation there will be a prioritization that identifies which user will have environment supremacy according to the defined hierarchies present on tables 10, 11 and 12, so it also has an reinforce in the negotiation process.

- **Negotiation module**, includes five agents types: the weather forecast agent gets information from a external API, the security constraints agent that has all the information related to security of each local system as detailed at table 8, the Case-based reasoning (CBR) Agent learn behaviour patterns along with the other variables, the hierarchies agent that has the hierarchy type of each user, finally the negotiation agent will use all this information to apply the formula with the included hierarchies, and at the end it will always validate the maximum values imposed by the security constraints agent.

  At this module will be the negotiation between different agents involved, namely conflicts management between different users. After the negotiation process ends we will have as result, the values to be applied at the environment, and that will be processed by the action module.

  In decision-making process, all users agents and agents representing the environment will be considered. With the different priorities that each of them has, and with this information will begin the negotiation process, that is done at the negotiation module by the negotiation agent, and considering the directives that are defined at the agents: Security Constraints Agent, CBR Agent and Hierarchies Agent.

  This module is also prepared to take advantage of information regarding the behavior of users, and with it achieve predictive comfort, more details on this aspect are available a section 5.7.

- **Action module**, after the negotiation module execution process, the values to be applied are obtained. These are used in this module and sent to actuators that will apply them in the different automation systems and actuators present at the environment. We have an agent for each of the four systems type, and a actuators manager agent, that validate the type of actuators that are present on each local, and according to this, it will use the appropriated operate agents, to operationalize the actuators presents on the space.

  After the negotiation result is achieved, will then be applied by the Actuators Manager Agent at the different system actuators present in the space, this is done in the Action Module.

- **User Behaviour Simulation module**, using the information generated by the user behaviour simulation described at section 3.2, it was possible to achieve the equation 3.1. This module is

seen as no part of the real time scenario, but it was displayed here as optional to demonstrate is importance in the negotiation process, and to feed the user behaviour agent.

3.4.3 Multi-Agent System Schema

The MAS that supports the system was developed using *JADE* [21], and implements four different agent role types:

- **Local System Agent**: Provide information of a single environment status. A new local system Agent is created for each environment that is introduced into the system;
- **Sensor Agent(s)**: Responsible for the retrieve the environment different conditions information, namely temperature, brightness, and others depending of each environment implemented sensors;
- **User(s) Manager Agent**: Responsible for passing all the information retrieved on the user information module to the negotiation agent, each user manager agent is associated with a single user present in the environment, and keeps track of the user preferences card;
- **Negotiation Agent**: Created in each environment, to manage the negotiation process detailed on subsection 3.4.4 using all the necessary information provided by the different Agents.

On figure 52, is summarized the developed schema, which includes the four agent roles type.

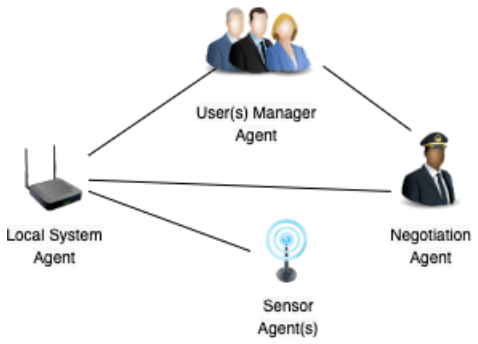


Figure 52: MAS schema.

3.4.4 Conflicts Management

At figure 51 is represented the different architecture modules, the agent that represents the local system receives its information, namely security information (maximum values of temperature, gases, and others). Also for each user present at the local, there will be an agent who represents him, it will receive information about the user preferences from the central system, that will be used for the negotiation process.

The negotiation process will then be done at the negotiation module. The negotiation result will then be passed to the different actuators present in the local, at the action module.

During this thesis, the environments focus were the domestic/family, professional environments (workplaces) and public spaces, where a large number of persons are usually present.

One of the used rules for conflict resolution was the preferences hierarchy. For the correct system functioning and to bring it as close as possible to reality, and knowing that in different environments there are naturally different hierarchies, which have a different control level over the environment. Thus, the hierarchies were defined, according to the tables 10, 11 and 12, for the different environments. These tables are just an example and were defined based on different principles that naturally exist, such as the differentiation between adult and children in home space. The existence of leadership at different levels and employees is based on the concept of work space. And the concept of the space owner and visitors in the different public spaces.

Of course, this kind of hierarchies can be customized on each local system, by its owner. Because naturally each space and its users have different specificities, which must be accommodated/guaranteed by this type of systems.

Starting with family contexts, it was considered to maximize adult elements (parents) preference value over the children, in a ratio of 1 to 0.75. Another hierarchy is the space preference value if it exists, in this case a proportion of 1.5 will be used. These cases may exist in spaces where there is some conditioning, such as kitchen's/Wc's, or other spaces that have some type of conditioning. All the proportions described and used for the rules, are detailed in table 10.

On the professional context, proportion values are also defined in a hierarchical way, and in this context the professional hierarchy of space will be used, as well as space preference value if it exists. The proportions described are detailed in table 11.

Regarding public and social spaces, the predominant value will obviously be the space value always defined by the space owner, with a proportion of 2. And each user will have a 0.15 proportion, because in these spaces is natural that there are little values variation, derived by high people movement. The proportions described are detailed in table 12. The equation used to achieve the optimum preference value to the different spaces is the following:

$$prefValue = \frac{\sum_{user=1}^{n} \{uPref * uHyerProp\} + (sPref * sProp)}{\sum_{user=1}^{n} \{uHyerProp\} + sProp} \tag{3.1}$$

At the equation 3.1, is depicted the equation for calculate the preference value to apply in the actuators present at the space. In this equation we have:

- n - number of users present in space;
- uPref - each user preference for the space;
- uHyerProp - each user hierarchy proportion;
- sPref - space preference;

- sProp - space proportion;

This equation was achieved, after using the work developed at the user simulation described at section 3.2, with this amount of information (720 days time window/1000 users), which originates more than three millions historical records, the equation is validated, and it was then used on the negotiation process. With this, it can be verified the user simulation work importance, that is described at section 3.2, which had enabled to have so many users and time records, without use the time window simulated (720 days) and without the financial cost to implement local systems to support this users amount (1000 users).

This equation uses the different users preferences and respective hierarchy proportions for each user, and sums the total, with space preference value multiplied by respective space proportion. These total, will be divided by the total sum of hierarchy proportions of all users in the space, summed with space proportions.

Type	Proportion
Adult	1
Child	0,75
Visitor	1
Space	1,5

Table 10: User's type/proportions - Home space.

Type	Proportion
Hierarchy_1	(100-1)
Hierarchy_2	(100-2)
Hierarchy_n	(100-n)
Space	150

Table 11: User's type/proportions - Work space.

Type	Proportion
User_1	0,15
User_2	0,15
User_n	0,15
Space	2

Table 12: User's type/proportions - Public/Social space.

As detailed in section 3.3.2, different constraints exist for each system. At figure 53 is detailed a example of constraints table that for each agent enters in the negotiation process.

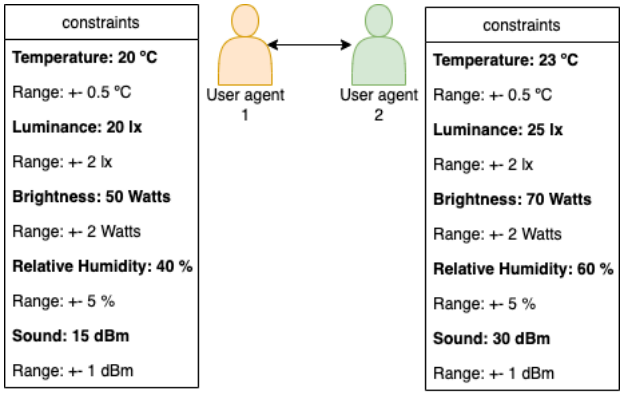


Figure 53: Two agents constraints table.

To be more concrete, on table 13 we see an example of temperature preference's for three users and a home space, and also each user type. With this information, at equation 3.2, it was calculated the temperature preference value that will be applied at that specific home space when these three users are present.

Preference	**User A** (Adult)	**User B** (Adult)	**User C** (Child)	**Home Space**
Temperature	20	23	22	19

Table 13: Example of preference value calculation - Home space.

$$prefValue = \frac{20 * 1 + 23 * 1 + 22 * 0.75 + (19 * 1.5)}{1 + 1 + 0.75 + 1.5} = \frac{88}{4.25} = 20.705 \tag{3.2}$$

Also in the same way, at table 14, we see a example of temperature preference's for three users and a work space, and also each user type. With this information, at equation 3.3, it was calculated the temperature preference value that will be applied at that specific work space when these three users are present.

Preference	**User A** (Hierarchy_1)	**User B** (Hierarchy_2)	**User C** (Hierarchy_2)	**Work Space**
Temperature	23	21	22	18

Table 14: Example of preference value calculation - Work space.

$$prefValue = \frac{23 * 99 + 21 * 98 + 22 * 98 + (18 * 150)}{99 + 98 + 98 + 150} = \frac{9191}{445} = 20.653 \tag{3.3}$$

3.5 System Data Privacy

With this solution, all the best practices defined by the European Union, in its most recent legislation regarding the General Data Protection Regulation (GDPR) (Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data), are guaranteed [111] [128].

In other words, the privacy of all stored information is safeguarded, as even if there is any unauthorized access to the stored information, it does not allow any user identification.

Compared to other systems on the commercial market (*Tado*, *Ring*, *Nest*, *Ecobee*, *Honeywell*, *Amazon*), the anonymization and safeguarding of information is clearly superior, since any of the other commercial solutions requires a registration process and respective user account, which includes: email, name, address, and others. This being clearly personal and sensitive information.

Following 3.5.1 and 3.5.2 subsections details all the developed work in this topic. And at subsection 3.5.3 all the implementation work to secure this system is also detailed.

3.5.1 Security and Privacy

The technological revolution that is felt, particularly in behavioral analysis field, IoT or big data, brings significant new challenges, including those related to the type of user information that can be collected, and the knowledge that can be obtained derived from the compilation of this information. Although not necessarily existing the user's authority to make this kind of information collection [101]. This revolution, especially in the IoT field has already clearly identified problems. In particular, the user data privacy and security. Foreseeing the dissemination of intelligent spaces, of which the user can, and want to take advantage of the interaction between systems, and the consequent sharing of personal data, this is a topic that needs a short-term resolution [75].

Obviously at this point there will be the requirements for concessions and commitments on the part of the user. Because it is understood that the solution will include the user's authorization in relation to the autonomous information sharing with the system and what is the information that it believes that it should be shared, blocked, and in concrete with which systems.

The IoT increases the personal privacy risk, and the confidentiality and integrity of data in organizations. Some IoT applications for consumers, particularly those related to health and wellness, which store sensitive personal information, and that consumers may not want to share.

In general, when using IoT applications users may not be aware of what type of information is being collected about them. For example, in the case of personalized offers in stores, knowledge of diverse user information is necessary, so that they are truly customized. In particular, the user's shopping record in the store, the products that the user visited in the store website, or information about the user's movement inside the store.

Users have to worry more than with possible embarrassment, about the misuse of his private information. Personal information related to health and well-being, and the historical consumption can affect areas as diverse as employment, access to credit, insurance prices, etc. Currently, many users have become more cautious about the personal information sharing with companies and other institutions. Privacy and security risks for both (consumers and organizations) have to be managed, to enjoy IoT full benefits [62].

Creating convincing value propositions, for which the information is collected and used, is critical for adoption. Insurance companies that use real automotive data by the insured to assess risk, claim that can reduce prices between ten and fifteen percent for most consumers. Transparency about the collection and use of data is as well crucial for confidence, such as the protection of the data collected is essential [113].

The IoT, increases the existing concerns at the safety level and introduces new risks. Because multiplies the already normal risks that exist in any data communication, as each device increases the possible attack area, and interoperability expands the potential attack scope. Each node is a possible entry point, and the interconnection can spread the damage. And so, the consequences of an attack on an IoT system that controls risk systems, can cause catastrophic damage.

As examples, a security system of an automated home that is compromised, or a medical monitor dysregulation, can cause life-threatening. An attack on a smart grid system could potentially cut off access to electric energy to millions of private households, as well as businesses, creating a massive economic damage and threats to health and safety. At the individual level, the security holes, in this kind of device may involve the misuse of personal data as well as theft [105].

Transparency, openness and ethical care will be essential to new approaches be accepted by the public. There will be a continuous need to address the practical implications and the proposed solution consequences, so that they fully comply with the best practices, about privacy and user data protection.

In this thesis applicability case, and because the sensitive user data collected, measures for these data safety will have to be taken. In the communication process is expected the security already implemented by communication technologies (BLE, NFC, WIFI Direct) that are used [60], [62].

Regarding user data monitoring, it is critical to consider on the proposed architecture/solution, the control and knowledge of the information shared by him. This will bring confidence to the user, because he will have control over his shared information and with what systems has been shared [18].

Normally, it is known that on latter case, there must be commitments by the user, so it has access to all system capabilities. But it is up to each one, set his own commitment threshold, between privacy and comfort that user intends to have by using the system.

3.5.2 Attack Vectors

With the objectives presented above, and considering the data and components that the system needs to achieve these objectives, an analysis was carried out, and for the architecture developed in this thesis,

the attack vectors have been identified, and these are following explained.

Next, at subsection 3.5.3 are identified and detailed the techniques to minimize all the attack vectors identified at this subsection.

Attack Vector A

In the case of an attack that get the information transmitted in the communication process between the application and the local system, exemplified in figure 54, the information obtained will be only the user's UUID. This information isolated has no value, because isn't more than a random generated set of characters.

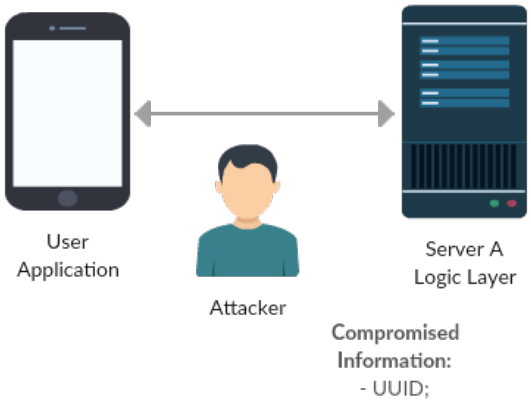


Figure 54: Attack Vector A.

Attack Vector B

If the attacker gains full access to a local system as shown in figure 55, the information exposed will be the users UUID present in the space, and the respective preference's cards, as well the local system ID. Also this information has little relevance.

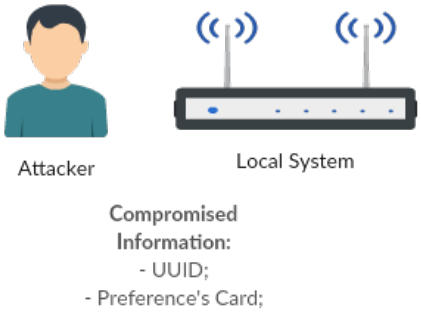


Figure 55: Attack Vector B.

Attack Vector C

In this case it is considered a hypothetical access to the server A, which contains the system logic layer. If that happens, the attacker will have access to the private key of the system, which will decode any UUID, or local system ID. Will also have access to all logical code of the system, as well as the access information to the database located on the server B. Figure 56 demonstrates the explained. This attack compromises all the information stored in the system.

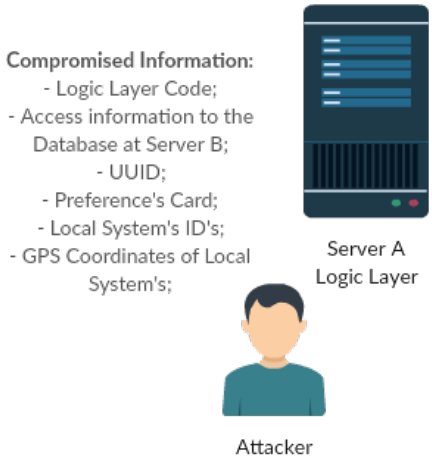


Figure 56: Attack Vector C.

Attack Vector D

An attack on the server B, affords access to all the information stored in the database. That includes the user's UUID with the preference's card's associated, the local system's ID's and local GPS coordinates. Also in the database is the entire user history, contextualized time and locally, which allows to collect diverse usage patterns types. On figure 57 is illustrated this scenario. This is also a critical attack.

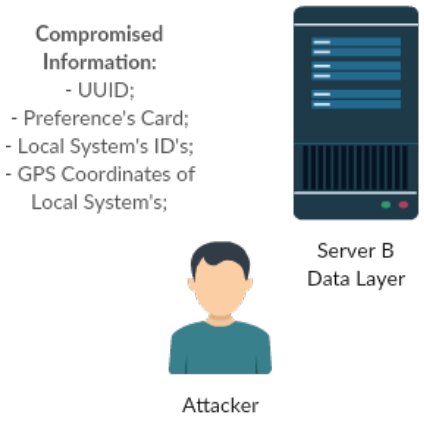


Figure 57: Attack Vector D.

3.5.3 Secure Attack Vectors

Currently IoT systems are in a big security risk. Especially because the developers, are not worried enough about the safety of such systems. However, with the growing trend of such systems and is integration in our everyday lives, this concern will have to increase as they start to appear isolated cases, which have harmed the users, both financially and in their safety and welfare. The proposed security architecture, to this IoT system, wants to avoid any of the presented risks, to this system users.

Systems that deal with personal data always bring privacy and security issues. And also the balance of these issues, with the need that persons have in interact with spaces in a transparent way and that those spaces smartly adapt to their preferences.

In this section, is proposed a solution to overcome these issues, and don't compromise the balance between security and personal comfort. All attack vectors identified in subsection 3.5.2, are minimized using the techniques identified in this section. Consequently increasing significantly the complexity to an attacker can gain access to useful information, or can link this information to take advantage, or even affect the system users.

As mentioned in section 1.5, one of this thesis priorities is to ensure privacy and data confidentiality. To achieve this goal, several mechanisms were designed in order to minimize the possible attack vectors. Figure 58 shows the overall context of the proposed architecture for the system. Includes the servers and existing communication mechanisms used and explained below.

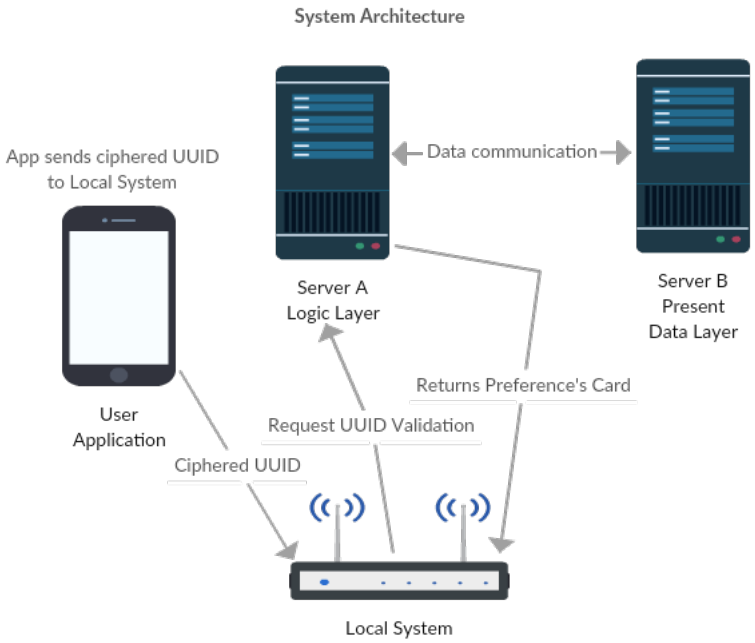


Figure 58: System Privacy - Architecture.

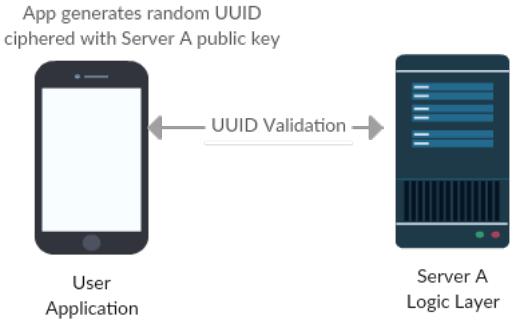


Figure 59: System - First Utilization.

- **Use of UUID**, to identify the user. The user identification process, it is necessary in this context to relate him to his preferences card, and is performed by generating a UUID in the first use of the system application. This unique UUID is randomly created by the application and is then validated their nonexistence on the server if the validation is positive, the UUID is associated with the user's preference card. If the validation is not positive a new random UUID is generated and the validation process will be held again.

  The application will allow the user to export the UUID created for his personal email or store it locally in another way, so that if it wants to use more than one device in the system or switch the device, this can be done. Note that only the randomly generated UUID and the preferences card are transmitted to the server, so there is no possibility of user identification [80].

- **Servers and component isolation**, two physical servers will be used. In order to separate the logic and data layer (database). Therefore possible individual attacks, which enable access to any of the servers do not compromise the entire system.

- **Data encryption**, all data transmitted between the servers are encrypted using Secure hash algorithm (SHA)-256 hash mechanisms, which introduces an extra security layer in protection of the data stored in the system [46].

- **Server hardening**, both servers only allow access through key mechanisms. Communication processes will be based on Hyper Text Transfer Protocol Secure (HTTPS) and Transport Layer Security (TLS) [46]. Other most common mechanisms for server hardening will be applied [140].

- **Communication with the local system**, as explained above, the communication between the user's smartphone and the local system, can be performed using BLE, NFC or Wi-Fi Direct. These technologies have their own security mechanisms implemented at the stack level, which can be properly configured in the local systems to maximize security. However, the UUID is also ciphered with the *Server A* public key before is sent to the Local System. With that we can guarantee that the UUID can't be captured in clear, and is only known by the smartphone application and the *Server A*.

- **GPS coordinates mask**, even though the user anonymization process is covered, for greater safety and because issues related to the user's location storage are critical. It is planned to convert the local systems GPS coordinates. This process is achieved by associating the coordinates to a randomly and periodically *Local System ID* change. Therefore the user's location information from a system, will be stored using the user UUID and the *system ID*, which due to its periodic change will not relate any information that can allow to achieve the user tracking.

These mechanisms implementation allows to significantly reduce the attack vectors identified at subsection 3.5.2. At the user data privacy level, the proposed architecture does not require to store any user

information. So, even if the data is compromised, will not be possible to identify the user, or make any relationship with that information, also all the detail about privacy is detailed at subsection 3.5.1.

3.6 Important Results

In addition to others, the main results of the work developed at this thesis, and presented in this section are the following:

- **Preference's Card**

  With the preference's card definition, an innovative form of the user have his preferences stored at any device has been achieved, and in order to pass them autonomously to any environment that can be automatically adapted.

  In addition, at any time, the user can update his preferences, and they will be used in all future environments that it will be present. This whole concept is innovative, and is not currently present on any market, research product or platform, with state of the art focused on comfort management considering only the devices present in each environment, and having the need to be programmed in a individual form.

- **System Information and Constrains**

  As regarding user's preferences, there are also certain characteristics and aspects that say only respect to the environment and devices that are inserted in it. They are defined in section 3.3.2, and with the solution proposed by this thesis, also innovative in the way of ensuring the safety of users and equipment's is achieved.

  As well anticipating its mobility, a concept already proposed by geolocation, but in another form, usually with the interaction between user device and a certain space only. With this proposal it was thus achieved the connection between different environments, and the movement that each user does between them. Here too, the literature and market don't propose any similar solution.

- **Multi-Agent System**

  A MAS light and scalable solution was achieved, as much the needs and that allows good performance using low cost hardware. Without which the global purpose of this solution would not make sense, because for it to be an effective solution, it is not enough to be technically viable, it must also be economically viable, whether for new or existing buildings.

  Thus all this system development, as well the way to feed him with the information needed for its operation has been developed, allowing a completely functional solution, which can be placed in production, for use by a commercial product/solution, as it is from the start planned in the development of this solution by the partner company.

It was also achieved the complete development of a solution to manage users preference's conflicts that naturally exist.

- **System Architecture**

  The entire architecture was developed with the intention of taking advantage of low cost hardware (Raspberry), and still achieving sufficient performance. All communication was developed, using NFC, BLE and Wi-Fi Direct. The entire services layer, which allows the collection and identification of present users, at every 30 minutes period, involving the different technologies and peripherals that users may be using (smartphones, wearables) was entirely developed.

  Thus, the entire system backend that supports the architecture and allows the good performance of the MAS solution, is fully developed and functional.

- **System Data Privacy**

  In this type of solutions involving user's sensitive and private information, especially in the anticipation of commercial purposes, it is essential to develop all accessory mechanisms that allow the privacy of all this information. Because currently the laws and regulations imposed by the European Union are increasingly and with more restrictions, with much more demanding when the product has a commercial purpose.

  Thus, the developed work in this thesis, can be considered as going beyond what is usually proposed in academic and scientific work, and safeguarding, that all current requirements of European regulation are met.

- **User behaviour Simulation**

  With the creation of behaviour simulated information, a truly interesting and capable alternative was achieved, in cases where there is no possibility of collecting real information by the different conditions already identified at section 3.2. This user's behavior and preferences simulation was clearly validated by the scientific community in different papers writhed and presented about it.

  The reviews were very positive, as well the possibility of taking advantage of the model to generate different information for different fields situations beyond the user preferences for which it was developed in this work. Thus being open the possibility of generalizing this simulated information, to make it as general and capable as possible, for use by the scientific community, in different cases of study.

4

Usage Scenarios

This chapter aims to present the proposal's validation by discussing two case studies, that are characterized and analyzed. The evaluation methodology is described, and it finishes with a detailed result analysis.

4.1 Smart space definition

In the literature we find different definitions for the smart space concept, among which the following stand out:

- A region of the real world that is extensively equipped with sensors, actuators and computing components [96];
- Work environments with embedded computers, information appliances, and multi-modal sensors allowing people to perform tasks efficiently by offering unprecedented levels of access to information and assistance from computers [117];
- Sentient, information-rich environment that sense and react to situational information to tailor themselves to meet users' expectations and preferences [5];
- An environment stipulated by intelligent agents, services, devices, and sensors to provide relevant services and information to meeting participants on the basis of their contexts [36];
- An environment that acts as an intelligent agent that perceives and acts on the environment through sensors and actuators to reason about and adapt to its inhabitants [42];
- An assistive environment that can sense itself and its residents and enact mappings between the physical world and remote monitoring and intervention services [64];
- A well-defined area that is embedded with computing infrastructure that enables sensing and controlling of the physical environment [121].

After analyzing all these definitions, and its application on this thesis field. It was formulated a customized definition to be applied on this thesis.

In this document, the definition proposed for smart space is the following: "**A environment that adapts in a autonomously, automatically and in a non-invasive way to achieve the user desires and needs**".

Adaptation means any, and all preferences adjustment that may be made in the space, namely those defined in section 3.3.1. All this, depends on the actuators present in the space, and the automation that they allow. In this way, the space should detect users entry and exit, automatically, and without the need for any interaction, and assess preferences of each one of them.

After that, the space must adjust to the present users, reaching the ideal preferences to apply, for the different actuators. Each smart space has to bee seen, as the physically individual and well-defined space, with their own actuators that can adjust preferences individually and exclusively for the space in question.

4.2 Evaluation scenarios

For the proposed framework analysis and evaluation, different scenarios were formulated. Initially it was applied in a two floors house.

In this way, it was possible to validate the domestic space concept, with a family composed of two adult users and a child, characterized in subsection 4.2.1. Their individual preferences were defined, and the MAS system analysis was carried out during a six months period.

The workspace concept was also defined, with different local systems being installed in the partner company's offices, and also detailed in subsection 4.2.2.

It was also planned to install some local systems, in partnership with the partner higher education institution, as well in a local health unit. But due to budget constraints, and costs associated with acquiring the high number of equipment's (*Raspberry's*) necessary for data acquisition, this was not possible.

This fact was also aggravated, due to the constraints introduced by the pandemic, having been completely impossible to access the health unit at that period, as well the higher education institution.

In section 4.4 the two defined scenarios results, are detailed, and explained for each of the aspects analyzed.

4.2.1 Home Scenario

Table 15 characterizes the different users that compose the home scenario, where the username, user type and proportion used in the equation 3.1 are defined.

Username	Type	Proportion
User1	Adult	1
User2	Adult	1
User3	Child	0,75

Table 15: Home Scenario - Users characterization.

For a greater detail of the installed local systems visible in figure 60, table 16 shows the name of each local system, its location and the associated divisions like open spaces that are somehow related, that is, the divisions in which the actuators present are controlled by the same local system.

Local System Name	Location	Associated divisions
Local_Livingroom	Living room	Living room/Kitchen
Local_Bedroom_GF	Bedroom_GF	Bedroom_GF
Local_Office	Office	Office
Local_Suite_FF	Suite_FF	Suite_FF/ Wc_Suite
Local_BedroomA_FF	BedroomA_FF	BedroomA_FF
Local_BedroomB_FF	BedroomB_FF	BedroomB_FF

Table 16: Home Scenario - Local system's characterization.

Figure 60 illustrates where the six different local systems (*Raspberry's*) are placed in the home scenario.

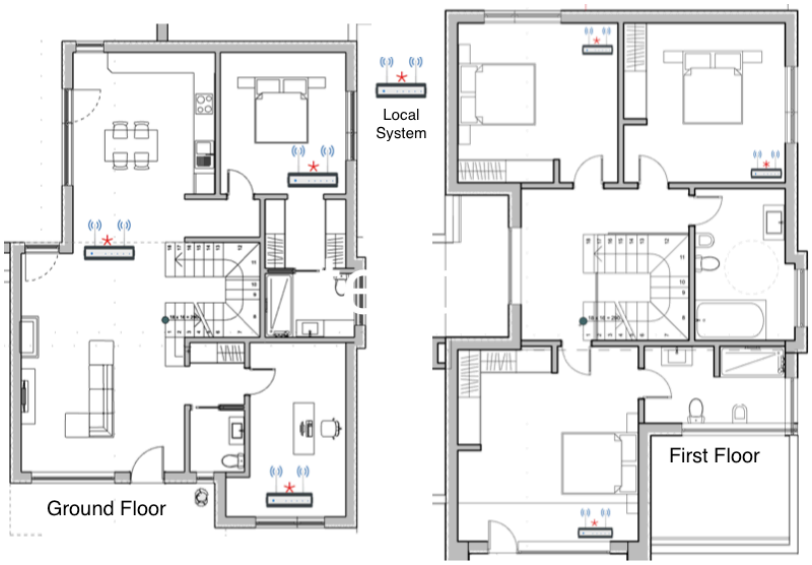


Figure 60: Home Scenario - Local System's installation.

Table 17 was developed, where all the entry records (samples) considered for analysis are represented, and they are divided by the six months under analysis (October 2021, November 2021, December 2021,

January 2022, February 2022 and March 2022).

Totalizing *15420* log records for the six months in question. Each of these samples represents one user entrance/presence, recorded by the local system. We can see an average of *84,45* samples registered for each day.

	Oct	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Total/ Average**
Nr. of Days	31	30	31	31	28	31	182
Nr. of Periods	992	960	992	992	896	992	5824
Total samples	3219	3033	1548	2988	1737	2895	15420
Average/Day	103,84	101,1	49,94	96,39	62,04	93,39	84,45

Table 17: Home Scenario - Total registered samples.

For greater detail, table 18 was created, containing each local system detail for the different months.

Local System Name	**Oct**	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Total**
Local_Livingroom	712	363	72	219	139	403	1908
Local_Bedroom_GF	1424	1453	217	1096	558	1207	5955
Local_Office	213	339	471	456	93	470	2042
Local_Suite_FF	167	291	11	345	491	268	1573
Local_BedroomA_FF	499	440	460	535	203	375	2512
Local_BedroomB_FF	204	147	317	337	253	172	1430
Total	3219	3033	1548	2988	1737	2895	15420

Table 18: Home Scenario - Local System's registered samples.

4.2.2 Work Scenario

Table 19 characterizes the six users that compose the work scenario, where the username, user type and proportion used in the equation 3.1 are defined.

Username	**Type**	**Proportion**
User10	Hierarchy_1	(100-1)
User20	Hierarchy_2	(100-2)
User30	Hierarchy_2	(100-2)
User40	Hierarchy_2	(100-2)
User50	Hierarchy_2	(100-2)
User60	Hierarchy_3	(100-3)

Table 19: Work Scenario - Users characterization.

For a greater detail of the installed local systems that are visible in figure 61, table 20 shows the name of each local system, its location and the associated divisions like open spaces that are somehow related, that is, the divisions in which the actuators present are controlled by the same local system.

Local System Name	**Location**	**Associated divisions**
Local_Office_A	Office_A	Office_A
Local_Office_B	Office_B	Office_B
Local_Office_C	Office_C	Office_C

Table 20: Work Scenario - Local system's characterization.

Figure 61 illustrates where the three different local systems (*Raspberry's*) were placed in the partner company's, one at each one of the offices as detailed at table 20.

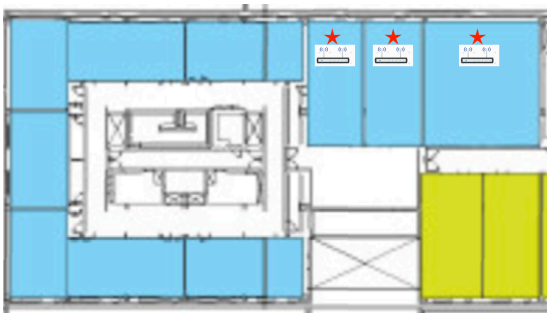


Figure 61: Work Scenario - Local System's installation.

Table 21 was developed, where all the entry records (samples) considered for analysis are represented, and they are divided by the six months under analysis (October 2021, November 2021, December 2021, January 2022, February 2022 and March 2022).

Totalizing *36578* log records for the six months in question. Each of these samples represents one user entrance/presence, recorded by the local system. We can see an average of *200,98* samples registered for each day.

	Oct	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Total/ Average**
Nr. of Days	31	30	31	31	28	31	182
Nr. of Periods	682	660	682	682	616	682	4004
Total samples	6024	8170	3676	6420	4968	7320	36578
Average/Day	194,32	272,33	118,58	207,1	177,43	236,13	200,98

Table 21: Work Scenario - Total registered samples.

For greater detail, table 22 was created, containing each local system detail for the different months.

Local System Name	Oct	Nov	Dec	Jan	Feb	Mar	Total
Local_Office_A	2512	2458	1281	1650	2335	2725	12961
Local_Office_B	2512	4915	1281	3300	2334	2724	17066
Local_Office_C	1000	797	1114	1470	299	1871	6551
Total	6024	8170	3676	6420	4968	7320	36578

Table 22: Work Scenario - Local System's registered samples.

4.2.3 System Actuators

For the different actuators operation, in the home scenario defined in 4.2.1, different valences were used, and are following detailed:

Temperature/Relative Humidity

Namely in terms of heating, this was achieved through a hydraulic underfloor heating that is divided into different circuits to cover the different house areas, as well for its control, six thermostats were used that allow in real time to send, using an API, the desired temperature. The thermostat and its operation mode, can be seen at figures 62 and 63.

Figure 62: Thermostat desired temperature.

Figure 63: Thermostat current temperature.

For cooling and relative humidity control, also six fan coils were used, one for each area, and controlled by individual thermostats, which also allow the desired temperature definition through an API, which can be seen at figure 64.

Figure 64: Fan coil thermostat.

Luminance/Brightness

For luminance and brightness, *Shelly* bulbs that have WIFI connection are used, that allow to control different luminance and brightness present at each individual environment, in the same way they have an API to integrate with other smart home systems, and that allow its direct control. This device can be seen at figure 65.

Figure 65: Smart bulb.

Sound

For sound, were used *Amazon Echo* speakers which have WIFI connection, and allow to control the sound volume and also the played music (sound source, playlist or gender) present at each individual environment, in the same way they have an API to integrate with other smart home systems, and that allow its direct control. This device can be seen at figure 66.

Figure 66: Smart speaker.

Security Systems

Also, was tested the possibility to use some security systems, and enable/disable this according to the user detection at the environment. The used device can be seen at figure 67.

Figure 67: Smart security system.

4.3 Evaluation methodology

To measure satisfaction, with the comfort preferences applied by a given local system, the different criteria was defined and presented at table 23, in order to identify when the conditions applied by the MAS results, satisfy the different users present in the space. Considering different actuators inertia as mentioned above, the preferences calculation to be applied at a given moment will always be carried out for periods of 30 minutes.

In this way, to validate user satisfaction we need to consider the manually change of the preferences at the space, after each automatic application by the system. As well the difference between the automatic applied, and manual value requested by the user for that period, as it is known that the more uncomfortable the space is for any preference or in a preference's set, the greater will be the adjustment differential performed in the manual change. In the same way, if no manual change is done, full satisfaction will be considered (100%).

Table 23, defines the percentage values that will be used to calculate the satisfaction, for each manual adjustment (higher or lower) of one change rate, the 100% of satisfaction will be discounted by the percentage value defined for each preference. To arrive at this value, the equation 4.1 was used, for each of the preferences used at the system, and in this way table 23 was populated with the value for each preference.

$$Insatisfaction = \frac{mProp * (nClicks * cRange)}{(maxV - minV)} \tag{4.1}$$

The Equation 4.1, is used to calculate the insatisfaction percentage, that will be discounted to the user satisfaction for each preference. In this equation we have:

- mProp - metric Proportion (Value: 200);
- nClicks - number of manual adjustment clicks;
- cRange - change range;
- maxV - maximum preference value;
- minV - minimum preference value.

For the *metric proportion 200 was used*, for other values, the correspondent preference value presented at table 23 is used. And the insatisfaction (%), is in this way achieved for each one of the preferences.

The insatisfaction degree calculated at table 23, represents the percentage value of insatisfaction for each preference when only *one manual adjustment click occurs*.

With all this information, next is presented two examples of insatisfaction calculation. Thus, we can verify with a simple example, for the temperature preference, in which the unsatisfied user makes a

Preference	**Min. value** (minV)	**Max. value** (maxV)	**Change range** (cRange)	**Insatisfaction degree** (%)
Temperature	15	28	+- 0.5	7,69%
Luminance	0	40	+- 2	10%
Brightness	0	100	+- 2	4%
Relative Humidity	20	80	+- 5	16.66%
Sound	0	30	+- 1	6,66%

Table 23: Satisfaction metrics.

decrement change of 2°C (4*0.5), that is, he will perform four manual adjustment clicks on the thermostat, each of the clicks will decrement 0.5°C. In this case, the insatisfaction equation would have the parameters identified in the equation 4.2, resulting in a total insatisfaction percentage of 30.77% for this example period.

So the insatisfaction, for this period will be 69.23% (100% - 30.77%).

$$Insatisfaction = \frac{200 * (4 * 0.5)}{(28 - 15)} = 30.77\% \tag{4.2}$$

In the same way, for the relative humidity preference, we can verify with a simple example, in which the unsatisfied user makes a decrement change of 10% (2*5), that is, he will perform two manual adjustment clicks on the thermostat, each of the clicks will decrement 5%. In this case, the insatisfaction equation would have the parameters identified in the equation 4.3, resulting in a total insatisfaction percentage of 33.33% for this example period.

So the insatisfaction, for this period will be 66.67% (100% - 30.33%).

$$Insatisfaction = \frac{200 * (2 * 5)}{(80 - 20)} = 33.33\% \tag{4.3}$$

4.4 Results analysis

To assess the results, the scenarios identified in section 4.2 were defined, and implemented. Thus, a six-month period was defined for the identified scenarios analysis, as well the users present. To verify satisfaction, equation 4.1 was used, and thus, the average satisfaction was calculated, for the total number of users and time period.

For the spaces characterized in section 4.2, information was then collected over a six months period. Thus, it was possible to carry out all the statistical analysis, in order to execute the results compilation presented below at section 4.4.1 and 4.4.2 and at tables 32 and 36.

Thus, as can be seen in the presented results, that have been analyzed considering the satisfaction metrics presented in table 23. It can be concluded that for both case scenarios a high degree of satisfaction was achieved, in the order of *95.12%* for the home environment, and *88.17%* for the work environment.

As previously mentioned, the results presented are preliminary and subject to industrial secrecy by the partner company. Therefore, all possible information is presented, considering the company's intention to commercialize the developed product, there are thus several restrictions on more data availability.

4.4.1 Home Scenario

Thus, all manual changes made during the testing phase were analyzed, and the satisfaction metric was calculated, by period of time/place. The average satisfaction was also measured, for the different periods: morning (8am-1pm), afternoon (1pm-7pm) and night (7pm-12pm).

Satisfaction

To assess the results, the scenarios identified in section 4.2 were defined. Also, the period definition for analyzing the identified scenarios was six months, as well the users present. To verify satisfaction, the equation 4.1 was used, and thus, the average satisfaction was reached, for the total number of users and time period, the results are shown at table 24. At figure 68 we can see the plot of this information.

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)	**Global Average**
Nr. of Days	182	182	182	182
Nr. of Periods	1820	2184	1820	1941,33
Avg. Insatisfaction	1,26%	3,16%	10,22%	4,88%
Avg. Satisfaction	98,74%	96,84%	89,78%	95,12%

Table 24: Home Scenario - Global Average Satisfaction - 6 Months.

At table 25, we can see the different details for the temperature preference in the six months period analyzed. It is detailed the number of days and periods, and the average for satisfaction and insatisfaction for each one of the three periods (morning, afternoon and night).

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)
Nr. of Days	182	182	182
Nr. of Periods	1820	2184	1820
Avg. Insatisfaction	2,96%	1,76%	0,42%
Avg. Satisfaction	97,04%	98,24%	99,58%

Table 25: Home Scenario - Temperature Average Satisfaction - 6 Months.

At table 26, we can see the different details for the luminance preference in the six months period analyzed. It is detailed the number of days and periods, and the average for satisfaction and insatisfaction for each one of the three periods (morning, afternoon and night).

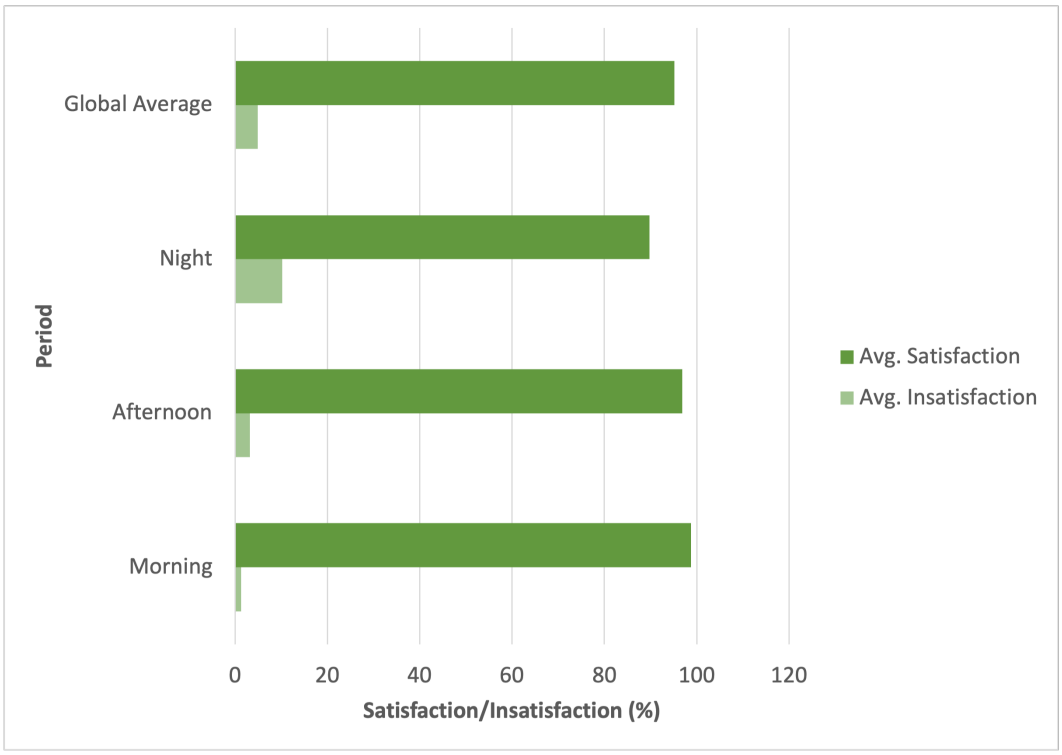


Figure 68: Home Scenario - Global Average Satisfaction - 6 Months.

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)
Nr. of Periods	1820	2184	1820
Avg. Insatisfaction	2,75%	0,92%	4,95%
Avg. Satisfaction	97,25%	99,08%	95,05%

Table 26: Home Scenario - Luminance Average Satisfaction - 6 Months.

At table 27, we can see the different details for the brightness preference in the six months period analyzed. It is detailed the number of days and periods, and the average for satisfaction and insatisfaction for each one of the three periods (morning, afternoon and night).

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)
Nr. of Periods	1820	2184	1820
Avg. Insatisfaction	1,54%	1,47%	0,88%
Avg. Satisfaction	98,46%	98,53%	99,12%

Table 27: Home Scenario - Brightness Average Satisfaction - 6 Months.

At table 28, we can see the different details for the relative humidity preference in the six months

period analyzed. It is detailed the number of days and periods, and the average for satisfaction and insatisfaction for each one of the three periods (morning, afternoon and night).

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)
Nr. of Periods	1820	2184	1820
Avg. Insatisfaction	2,75%	2,29%	1,83%
Avg. Satisfaction	97,25%	97,71%	98,17%

Table 28: Home Scenario - Relative Humidity Average Satisfaction - 6 Months.

At table 29, we can see the different details for the sound preference in the six months period analyzed. It is detailed the number of days and periods, and the average for satisfaction and insatisfaction for each one of the three periods (morning, afternoon and night).

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)
Nr. of Periods	1820	2184	1820
Avg. Insatisfaction	0,37%	0,91%	1,1%
Avg. Satisfaction	99,63%	99,09%	98,9%

Table 29: Home Scenario - Sound Average Satisfaction - 6 Months.

To summarize, and have a greater detail, one day was randomly selected from the period under study, and in table 30, average for satisfaction and insatisfaction for each one of the three periods (morning, afternoon and night) is presented.

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Night** (7pm-12pm)	**Global Average**
Nr. of Periods	10	12	10	10,67
Avg. Insatisfaction	20%	0%	20%	13,33%
Avg. Satisfaction	80%	100%	80%	86,67%

Table 30: Home Scenario - Global Average Satisfaction - One Day.

Consumption

Also regarding energy savings, and knowing that it is currently a factor that isn't and cannot be neglected by any individual user or any business entity.

Considering the costs increase with different energy types, as well the ecological footprint that its production represents, the savings metric was also calculated, always considering that the purpose of this solution would not have this as prime factor, but indeed the maximum user comfort.

But knowing from the start that with all the introduced automatism's (detection of users present at the space, adjustment to minimum reference values in empty spaces, etc.) by the proposed solution, a decrease in consumption would be expected by itself.

Compared to solutions that only implement pre-programmed fixed adjustments and which most of the time don't include any automatism, such as simply allowing to detect absence periods, for example in the workspace, such as vacations, holidays or others, in this scenario, savings are expected to be even more significant.

To check exact values, the month global consumption was been verified for each analyzed space, and compared with the same month global consumption, after applying the solution.

At table 31 we can see the mean value for the baseline day consumption, and the day consumption for the analyzed period, and also the difference in kWh, and the savings in percentage value. At figure 69 we can see the plot of this information.

Scenario	**Baseline** (kWh)	**Period analyzed** (kWh)	**Difference** (kWh)	**Savings** (%)
Home	35,2	32,05	3,15	9,84

Table 31: Home Scenario - Day Energy consumption (mean value).

At table 32 we can see the total consumption value for the baseline, and for the 6 months period analyzed for the home scenario, and also the difference in kWh, and the savings in percentage value. At figure 70 we can see the plot of this information.

4.4.2 Work Scenario

Satisfaction

To assess the results, the scenarios identified in section 4.2 were defined. Also, the period definition for analyzing the identified scenarios was six months, as well the users present. To verify satisfaction, the equation 4.1 was used, and thus, the average satisfaction was reached, for the total number of users and time period, the results are shown at table 33. At figure 71 we can see the plot of this information.

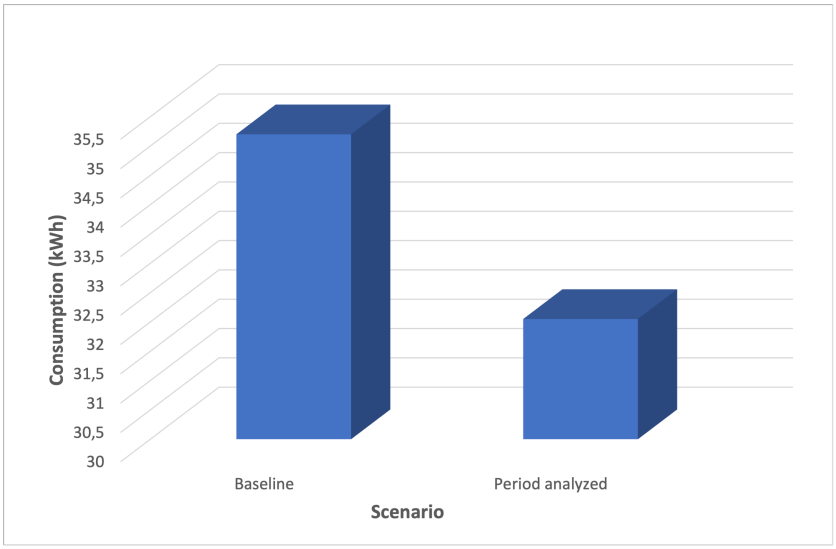


Figure 69: Home Scenario - Day Energy consumption (mean value).

	Oct	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Total**
Nr. of Days	31	30	31	31	28	31	182
Baseline (kWh)	806	960	1240	1426	952	930	6314
Period analyzed (kWh)	682	870	1209	1209	868	868	5706
Difference (kWh)	124	90	31	217	84	62	608
Savings (%)	18,18	10,34	2,56	17,95	9,68	7,14	10,66

Table 32: Home Scenario - Energy consumption - 6 Months.

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Global Average**
Nr. of Days	182	182	182
Nr. of Periods	1820	2184	2002
Avg. Insatisfaction	5,93%	17,72%	11,83%
Avg. Satisfaction	94,07%	82,28%	88,17%

Table 33: Work Scenario - Global Average Satisfaction - 6 Months.

To summarize, and have a greater detail, one day was randomly selected from the period under study, and in table 34, average for satisfaction and insatisfaction for each one of the two periods (morning and afternoon) is presented.

Consumption

At table 35 we can see the mean value for the baseline day consumption, the day consumption for

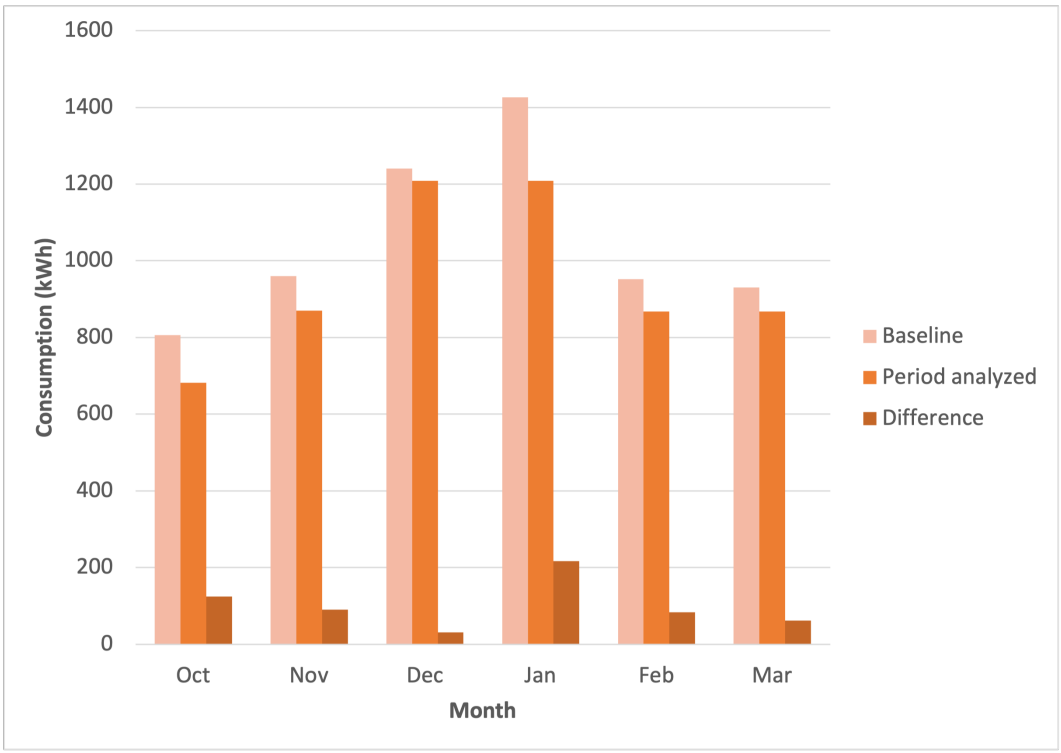


Figure 70: Home Scenario - Energy consumption - 6 Months.

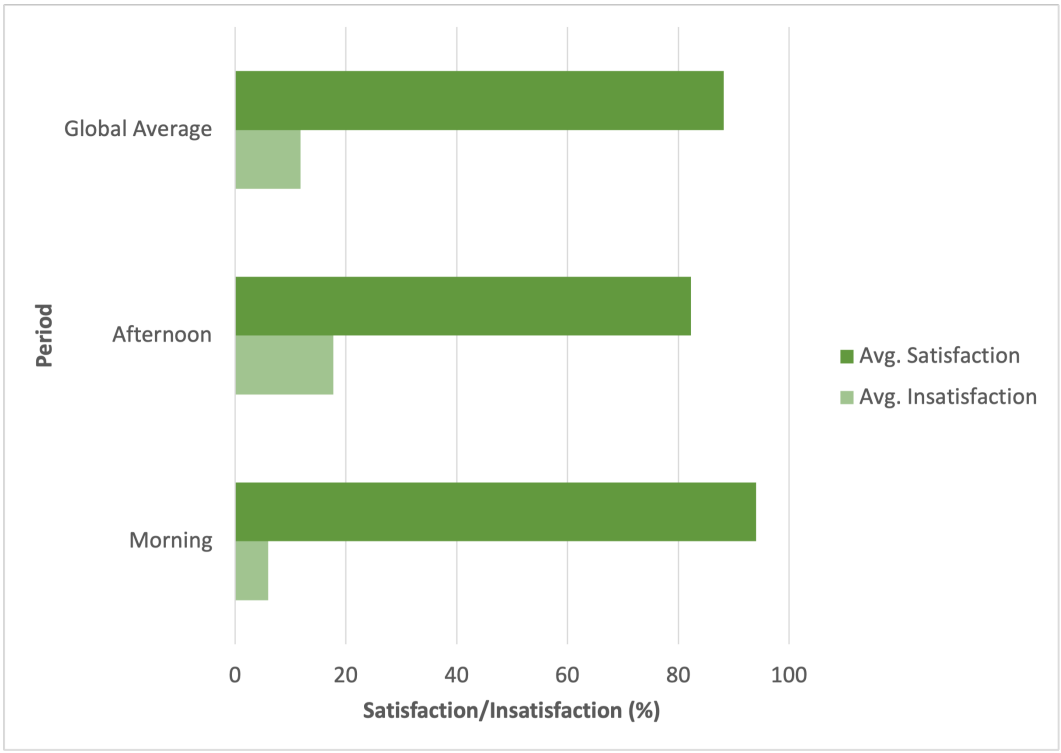


Figure 71: Work Scenario - Global Average Satisfaction - 6 Months.

the analyzed period, and also the difference in kWh, and the savings in percentage value. At figure 72 we can see the plot of this information.

Time period	**Morning** (8am-1pm)	**Afternoon** (1pm-7pm)	**Global Average**
Nr. of Periods	10	12	11
Avg. Insatisfaction	20%	8,33%	14,17%
Avg. Satisfaction	80%	91,67%	85,83%

Table 34: Work Scenario - Global Average Satisfaction - One Day.

Scenario	**Baseline** (kWh)	**Period analyzed** (kWh)	**Difference** (kWh)	**Savings** (%)
Work	42,5	36,4	6,1	16,76

Table 35: Work Scenario - Day Energy consumption (mean value).

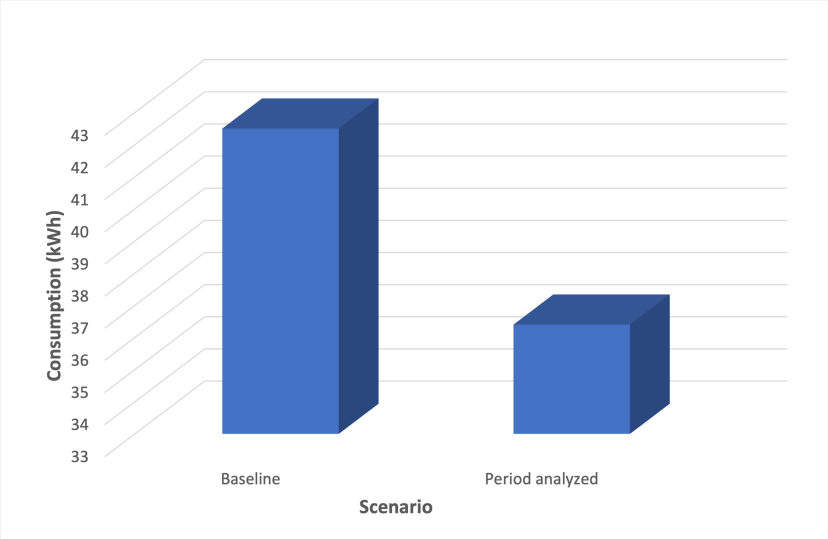


Figure 72: Work Scenario - Day Energy consumption (mean value).

At table 36 we can see the total consumption value for the baseline, and for the 6 months period analyzed for the work scenario, and also the difference in kWh, and the savings in percentage value. At figure 73 we can see the plot of this information.

4.5 Summary

The proposal described in chapter 3 was validated here through two case studies analysis. They were created, based on that they are normally the two more representative used scenarios, and where this solution could have a significant added value.

	Oct	Nov	Dec	Jan	Feb	Mar	Total
Nr. of Days	31	30	31	31	28	31	182
Baseline (kWh)	992	1050	1519	1643	1036	992	7232
Period analyzed (kWh)	899	840	1364	1612	868	868	6451
Difference (kWh)	93	210	155	31	168	124	781
Savings (%)	10,34	25	11,36	1,92	19,35	14,29	12,11

Table 36: Work Scenario - Energy consumption - 6 Months.

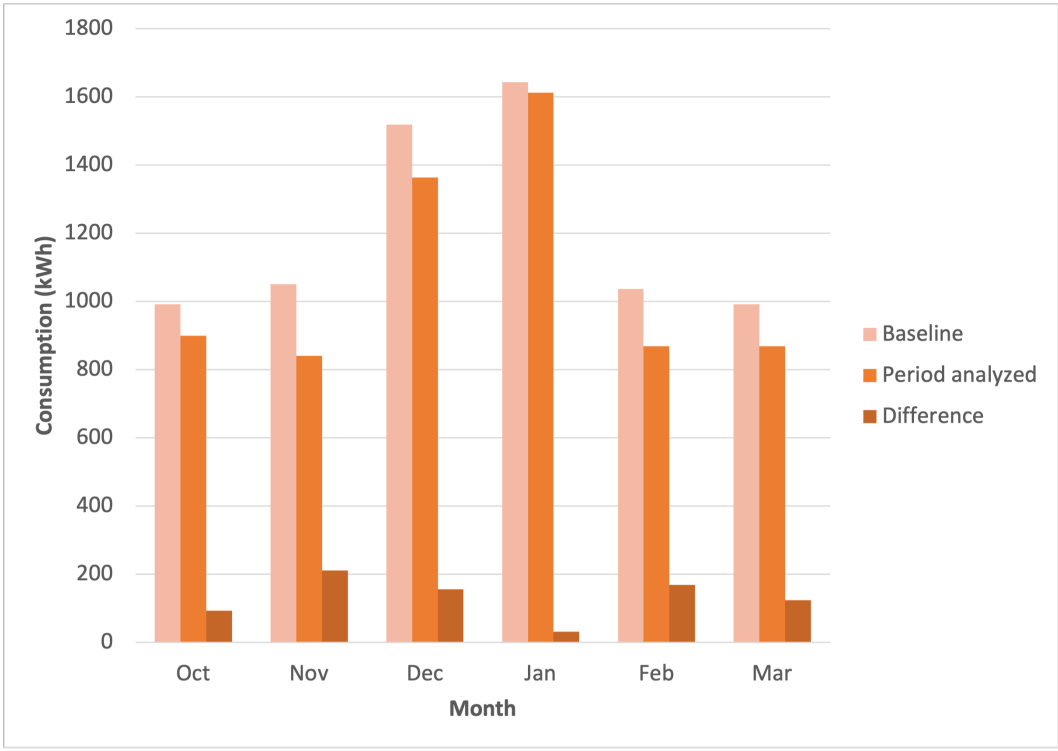


Figure 73: Work Scenario - Energy consumption - 6 Months.

In this chapter, the two scenarios results were thus defined and analyzed. It was detailed the characterization and period analyzed in the different smart space scenarios, the detailed methodology, and the analysis was done considering satisfaction and consumption.

Analyses were made, and the results were presented using tables and charts. Algorithms were presented to describe how the framework deals with uncertain situations after being identified in the case studies.

We can see from the results analysis, the system high efficiency, with high percentages of satisfaction for both scenarios under analysis. With this we can have a reality sample of what this system could mean to increase comfort, as well the convenience and quality of people's lives.

The analyzed period (6 months) is not very extensive, it can be seen anyway, as sufficient for this type of spaces (domestic, small company) analysis, where users remain in some way very constant, and where there is thus no significant variance in their preferences.

5

Conclusions

This chapter closes the thesis, a summary with some final considerations is presented, and the thesis goals and achievements are highlighted and briefly discussed. The scientific contributions are described, and also some points to future research directions, based on the results so far reached are proposed in the last section.

5.1 Summary

All this work led to different visits involving different institutions. In particular, visits were made to the Grupo de investigación en Bioinformática, Sistemas Informáticos Inteligentes y Tecnología Educativa (BISITE) research group at the University of Salamanca (USAL), to exchange experiences and collaborate in the solution presented development.

A close collaboration was also carried out with the Research Centre in Digitalization and Intelligent Robotics (CeDRI) research center of the Polytechnic Institute of Bragança (IPB), namely with the participation in different conferences and workshops developed. Likewise, with the *Algorimi* research center at the University of Minho (UM), and with its members.

Finally, I would like to highlight the different involvement of the partner company *Techwelf*, which has always leveraged the development of this thesis. All these partnerships were relevant for the knowledge sharing and experiences exchange, and also a significant enrichment of the developed work.

Context-aware systems are, gradually, becoming a reality, intelligent devices that naturally interact with users are already available on the market. They will help the consolidation of AmI through intelligent environments. To build this kind of application, it is necessary to develop context-aware systems, which can sense what is happening in the environment and behave based on that. The system should use data of the surroundings to execute services aiming to assist users. However, this reasoning is not a trivial task. Moreover this complexity can be augmented due to the fact that data collected from sensors can be incomplete or incorrect. In this way, context-aware systems may not be able to process situations due to the data incompleteness or its reliability.

This doctoral thesis researches about some open questions about different aspects in context aware systems, namely some related with smart spaces and its adaptation to the present users comfort preferences. As main objective, we can see the user presence detection, and consequently, the automatically collection of its different comfort preferences. For achieve this objective, several research questions were addressed as well a research hypothesis is defined as *the feasibility of ensuring comfort conditions according to the preferences of each individual and in a transparent, non-intrusive and safe way*.

Following, diverse work was developed to answer these questions, always pursuing the proposed objectives. During this work, several phenomenons were identified as being responsible, for influence the objectives achievement, namely the users different behaviour, its quotidian movement and related uncertainty, the different conflicts preferences for the different users, the difficulty to identify the user presence and absence in a automatic and non invasive way, transparent ways of sharing preferences and also users privacy.

If the proposed system is not prepared to deal with all the identified aspects, it will not make the proper decisions as well the space adaptations to the user comfort preferences, that satisfy the present users.

IoT and AI researches, are currently in evidence and are currently trend topics. Build environments with intelligent devices acting together in an autonomous way, to improve users quality of live, is one of the goals. The research results, as defined in the hypothesis at section 1.4, include user detection, user preferences management, conflicts management, and finally the system must adapt the environment to the different present users. With this, the objectives proposed in Chapter 1 are achieved, allowing the autonomous systems improvement and directly contributing to the IoT paradigm evolution.

With the state of the art organization, the characteristics, contributions, and each research topic relevance was described. Also describing how they were incorporated in the proposed solution. At Chapter 2 all the support concepts and technologies were discussed, as well the AmI concepts and projects. It is then defined what we understand that is a user in a smart space, and this concept was defined and proposed, considering the study domain.

Chapter 3, shows all the characterization and developed methods, starting with the general requirements definition, where the relevant information is defined, and the different ways/approaches (predictive, reactive) of achieving comfort are explained. Next the preferences card concept is introduced, and also the system information and constrains, that supports the developed architecture with all the needed information about users and spaces. Also all the implementation details are here described, from the user identification process to the preferences conflict management.

Continuing with the user behaviour simulation developed to have several information, and support the developed work as well the respective validation and statistical analysis. All the related to the MAS developed: from the assumptions, used framework, architecture, and how the preferences conflict resolution is done. And all the attack vectors identified in this system, and the proposed mitigation techniques, to improve the system data privacy. This chapter finishes with all the important results and scientific contributions of this thesis.

This Ph.D. thesis aimed to present an approach to have a smart space, for that propose a definition was created, and different usage scenarios are characterized for doing a proper evaluation, also a methodology is described, and it ends with a result analysis that includes a statistical validation, all these topics can be seen at Chapter 4.

5.2 Discussion and Conclusions

With this work, the total development of a system architecture, that allows to support all the necessary communication process, general requirements and information management is done and detailed at section 3.3. Also to achieve a solution to optimize conflicts management, a framework that includes a MAS architecture and respective cognitive model for a Smart Home was achieved, and had been developed using *Jason* and *ARGO*. This work is detailed at section 3.4.

The proposed system includes different functionalities, which were developed to achieve the proposed objectives and new functionalities are also added during the development. Of the initially proposed objectives in section 1.5, the following developed functionalities stand out:

- **Detection of present users**, by passing each user UUID, the system detects present users, with update readings being made every 30 minutes period, as described at subsection 3.1.3;
- **User exit**, as a reading/scan is performed every 30 minutes, the system also detects any user who is no longer present in the environment;
- **Passing the preference card**, whether online or offline, as explained in figure 37, the system collects the preference card from the present users;
- **Calculation of the preference values to be applied**, using the developed MAS system, equation 3.1, the different user's preference card's information that are present in the space, and the respective hierarchies, is achieved the optimal value that will be applied by existing actuators for each preference.

In a complementary way, during the development, new features were also added, which were understood as an asset to make the developed system more complete, namely the following:

- **Space air quality management**, using the maximum and minimum value for different preferences, as well the detection of different levels of gases (CO2, CO) to measure the air quality, and continuously ensure the safety of users;
- **Actuators management**, with the maximum and minimum values parameterization for each actuator present in the space, as well with the complete operation automatism, without the need for constant users intervention, the durability and maintenance of the equipment is thus guaranteed, which also naturally translates into financial savings for users;

- **Security and privacy**, all technical details to ensure the users information security, and privacy has been ensured, as described in subsection 3.5.3;

- **Possibility of predicting the preferences to be applied in the space**, with the AI model application, and available history, the system can be quickly expanded to start adapting spaces in a predictive, instead of reactive way. Thereby, there would be a significant optimization of the system quality, and additionally of the comfort measured by the user.

Considering this research applicability, on a product development to have commercial applicability, is mandatory that innovation is present to surpass all the competitors, and in this way have economic viability to the company. During this research, the different competitors, and all the related products are analyzed, namely its features, capabilities an functioning.

We are talking about the big players in the world market, namely *Amazon*, *Google*, *Honeywell*, and others. All these have related products, but any of them, use this kind of approach, they doesn't have this concept of preferences card, user detection, conflicts management, security constraints and others. They basically have a smartphone application that replaces the usual manual controller that is normally present on the wall. With this the user can define schedules, and different preferences, for each schedule, but this is always restricted to the defined, and doesn't have any automatism that makes this approach smarter.

So nowadays with all AI revolution, the user consider these products as "dumb" solutions, very distant of what the current technology allows. Considering this, all this features and techniques introduced by this approach, can be seen as real innovation, and state of the art on this kind of solutions for smart spaces.

With the applicability of this proposal, on a product that will be introduced on the market, we anticipate a new paradigm creation, on this kind of solutions development to adapt the space to the user comfort. Also this approach can be used in other space types, like cars or public transports, as detailed on research future directions at section 5.7.

It can be concluded saying that all the proposed objectives are achieved, and a full proposal to develop a innovative market product is done. In the current context, where more than ever academic research has to be objective and pragmatic, creating innovative technologies and products that effectively contribute to the competitiveness and growth of companies, this thesis fits perfectly in that purpose, as it was proposed in its initial conception together with the partner company *Techwelf*.

5.3 Hypothesis validation

The hypothesis presented at section 1.4 was addressed throughout this work chapters. Nevertheless, following is brief discussed the hypothesis validation:

- *How can we characterize an environment ?*
  The environment has characterized using different characteristics, namely the area, present actuators that support different comfort features, and how it was inserted in a big space.

- *How is human comfort defined ?*
  We have define comfort as the point when the user doesn't have any need to adjust any preference, at the environment where it is present.

- *What are the human preferences set ?*
  They are diverse, and very different between each user, but we have defined the ones that make most influence, namely temperature, luminance, brightness, relative humidity, sound. And with that we have defined the user preferences card.

- *Can we automatically detect user preferences ?*
  Yes, if we know the present user, in that way it can be detected the preference value that user defines for each preference, and it can be assumed that it is this preferred preference value, and we can do a match between user and preference.

- *Is it possible to solve/minimize user comfort preferences conflicts ?*
  Yes, it is perfectly possible to minimize and even solve the conflicts, depending on the number of user, and the amplitude between there preferences. And so, using different techniques, as hierarchies, average and others, we can calculate the optimal preference value to apply.

- *Can human comfort be measured ?*
  Yes, we can calculate satisfaction. Namely, reading if and how many times the user manually changes the preference value at the actuators. In this way we can assume he has discomfort, and it will be proportional to the amplitude that preference is changed.

Considering all the answered questions, and the hypothesis presented at section 1.4: *Smart Spaces as a transparent, non-intrusive, and safe way, to promote the satisfaction and comfort of users, according to the preferences of each individual.* It can be concluded, that this thesis achieve the proposed hypothesis, namely developing a framework to allow a smart space creation, that supports the different users comfort in a transparent, non-intrusive, and safe way.

5.4 Contributions

Considering a doctoral thesis importance, and the fact that the proponent has a position as university professor, the contribution includes two main perspectives, which are described in the following sections.

5.4.1 Direct contributions

The main results and direct contributions of the work developed at this thesis, are the following:

- *Preference's Card*;
- *System Information and Constraints*;
- *Multi-Agent System*;
- *System Architecture*;
- *System Data Privacy*;
- *User behaviour Simulation*.

All these results are fully detailed and explained at section 3.6.

5.4.2 Additional contributions

The proponent works as invited Professor of graduation courses at the IPB. Since February of 2016, the proponent is developing his doctoral research at the MAP Doctoral Program in Computer Science, a joint PhD degree by the Universities of Minho, Aveiro and Porto.

Considering this, the outcomes expected from this work shall be used by graduation and post-graduation students of all levels (master and doctoral), in scientific researches developed at IPB, UM and other educational institutions, through cooperation initiatives.

5.5 Academic Outcomes

At this section are identified, all the academic outcomes namely: publications, scientific events participation, organizing committees and projects advisor.

5.5.1 Related publications

Submitted, and waiting notification decision:

- *Smart Environment: Using a Multi-Agent System to Manage Users and Spaces Preferences Conflicts*
  The 19th IEEE International Conference on Ubiquitous Intelligence and Computing (UIC 2022)
- *A Multi-Agent System to achieve predictive comfort at an Adaptive Environment System*
  15th International Conference on Agents and Artificial Intelligence (ICAART 2023)

Accepted for publication:

- *Adaptive Environment System to manage comfort preferences and conflicts*
  International Conference on Optimization, Learning Algorithms and Applications (OL2A 2022)
- *Manage users and spaces security constraints on a multi-agent system in a Adaptive Environment System*
  Symposium of Applied Science for Young Researchers (SASYR 2022)
- *Adaptive System to manage user comfort preferences and conflicts at everyday environments*
  19th International Conference on Distributed Computing and Artificial Intelligence (DCAI 2022)
- *Adaptive System To Manage Everyday User Comfort Preferences*
  VII Ibero-American Congress on Entrepreneurship, Energy, Environment and Technology (CIEEMAT 2022)

Published:

- Oliveira, P.F., Novais, P., Matos, P., Using jason framework to develop a multi-agent system to manage users and spaces in an adaptive environment system (2021), Advances in Intelligent Systems and Computing, 1239 AISC, pp. 137-145., DOI: 10.1007/978-3-030-58356-9_14, Springer Science and Business Media Deutschland GmbH, 11th International Symposium on Ambient Intelligence, ISAmI 2020, ISSN: 21945357, ISBN: 9783030583552
- Oliveira, P.F., Novais, P., Matos, P., Manage comfort preferences conflicts using a multi-agent system in an adaptive environment system (2021), Advances in Intelligent Systems and Computing, 1239 AISC, pp. 284-288., DOI: 10.1007/978-3-030-58356-9_32, Springer Science and Business Media Deutschland GmbH, 11th International Symposium on Ambient Intelligence, ISAmI 2020, ISSN: 21945357, ISBN: 9783030583552
- Oliveira, P., Novais, P., Matos, P., Generating Real Context Data to Test User Dependent Systems - Application to Multi-agent Systems (2019), Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11523 LNAI, pp. 180-187., DOI: 10.1007/978-3-030-24209-1_15, Springer Verlag, 17th International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAMS 2019, ISSN: 03029743, ISBN: 9783030242084
- Oliveira, P.F., Novais, P., Matos, P. , A multi-agent system to manage users and spaces in a adaptive environment system (2019), Communications in Computer and Information Science, 1047, pp. 330-333., DOI: 10.1007/978-3-030-24299-2_31, Springer Verlag, 17th International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAMS 2019, ISSN: 18650929, ISBN: 9783030242985

- Oliveira, P., Pedrosa, T., Novais, P., Matos, P., Towards to secure an IoT adaptive environment system (2019), Advances in Intelligent Systems and Computing, 801, pp. 349-352., DOI: 10.1007/978-3-319-99608-0_43, Springer Verlag, 15th International Conference on Distributed Computing and Artificial Intelligence, DCAI 2018, ISSN: 21945357, ISBN: 9783319996073

- Oliveira P., Matos P., Novais P., Behaviour Analysis in Smart Spaces, 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress, ISAmI 2020, ISBN: 978-1-5090-2772-9, ISSN: 2150-329X, pp 880-887, 2016. Indexed: ISI Web of Science, Scopus, DBLP

- Oliveira, P., Novais, P., Matos, P., Challenges in smart spaces: Aware of users, preferences, behaviours and habits (2017), Advances in Intelligent Systems and Computing, 619, pp. 268-271., DOI: 10.1007/978-3-319-61578-3_34, Springer Verlag, 15th International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAMS 2017, ISSN: 21945357, ISBN: 9783319615776

5.5.2 Other publications

- *Security constraints on a multi-agent system to manage users and spaces in a Adaptive Environment System*
  12th International Symposium on Ambient Intelligence (ISAmI'21)

- *Using Jason framework to develop a multi-agent system to manage users and spaces in an Adaptive Environment System*
  11th International Symposium on Ambient Intelligence (ISAmI'20)

- *Manage comfort preferences conflicts using a multi-agent system in an Adaptive Environment System*
  11th International Symposium on Ambient Intelligence (ISAmI'20)

- *Sistema multi-agente para a gestão de utilizadores e espaços num ambiente adaptativo*
  VI Encontro de Jovens Investigadores

- *Generating real context data to test user dependent systems - application to multi-agent systems*
  17th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'19)

- *A multi-agent system to manage users and spaces in a Adaptive Environment System*
  17th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'19)

- *Towards to secure an IoT Adaptive Environment System*
  16th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'18)

- *Challenges in Smart Spaces: Aware of users, preferences, behaviours and habits*
  15th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'17)
- *Planeamento e modelação de um ambiente inteligente para fábricas inteligentes*
  IV Encontro de Jovens Investigadores
- *BLEGen - A Code Generator for Bluetooth Low Energy Services*
  7th International Conference on Computer Science and Information Technology (ICCSIT 2014)
- *Aplicação de Bluetooth Low Energy no controlo e monitorização de dispositivos de muito baixo consumo*
  II Encontro de Jovens Investigadores
- *Code generator for bluetooth low energy services*
  11th International Conference Applied Computing
- *Espaços Inteligentes: Conhecedores de utilizadores, preferências, comportamentos e hábitos numa abordagem não invasiva*
  Conferência Internacional em Processos de Co-Criação no Ensino Superior (In2CoP)
- *Aumentando a segurança de Ambientes Inteligentes*
  IV Encontro de Jovens Investigadores

5.5.3 Scientific events participation

During the Ph.D. course, the candidate has participated in several scientific events, such as summer schools, symposiums and conferences. Besides presenting papers with the ongoing research status, this events added the possibility of doing some scientific network and discussion with other related fields researchers. These events are at the following list:

- International Summer School on Deep Learning (DeepLearn2017) in 2017, Bilbao, Spain;
- 15th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'17) in 2017, Porto, Portugal;
- 16th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'18) in 2018, Toledo, Spain;
- 17th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'19) in 2019, Ávila, Spain;
- 11th International Symposium on Ambient Intelligence (ISAmI'20) in 2020, L' Aquila, Italy;
- 12th International Symposium on Ambient Intelligence (ISAmI'21) in 2021, Salamanca, Spain;

- 19th International Conference on Distributed Computing and Artificial Intelligence (DCAI'22) in 2022, L' Aquila, Italy;
- VII Ibero-American Congress on Entrepreneurship, Energy, Environment and Technology (CIEEMAT) in 2022, Bragança, Portugal;
- Symposium of Applied Science for Young Researchers (SASYR 2022) in 2022, Viana do Castelo, Portugal;
- International Conference on Optimization, Learning Algorithms and Applications (OL2A 2022) in 2022, Póvoa do Varzim, Portugal.

5.5.4 Committees/Projects advisor

During the Ph.D. course, the candidate has participated in several conferences organizing committees, some journal editorial boards, and has been advisor of different graduation final projects. These are identified in the following subsections.

Editorial Board

- International Journal of Network Security & Its Applications (*IJNSA*)
  ISSN: 0974-9330, 0975-2307
  http://airccse.org/journal/editorial.html
- Journal of Artificial Intelligence and Big Data (*JAIBD*)
  ISSN: 2771-2389
  DOI prefix: 10.31586/jaibd
  https://www.scipublications.com/journal/index.php/jaibd/editors

Organizing Committees

- *Symposium of Applied Science for Young Researchers (SASYR 2022)*
  http://sasyr.ipb.pt/
- *International Conference on Optimization, Learning Algorithms and Applications (OL2A 2022)*
  http://ol2a.ipb.pt/
- *International Workshop on Additive Manufacturing and STEAM Education (IWAM 2022)*
  http://iwam.ipb.pt/
- *Seminário Luso-Brasileiro de Ensino Superio (SemLB 2022)*
  http://semlb.ipb.pt/

- *International Conference on Co-Creation Processes in Higher Education (In2CoP 2020)*
  http://in2cop.ipb.pt/
- *VI Encontro de Jovens Investigadores do Instituto Politécnico de Bragança (EJI 2019)*
  http://eji.ipb.pt/
- *V Encontro de Jovens Investigadores do Instituto Politécnico de Bragança (EJI 2018)*
  http://eji.ipb.pt/
- *IV Encontro de Jovens Investigadores do Instituto Politécnico de Bragança (EJI 2017)*
  http://eji.ipb.pt/
- *III Encontro de Jovens Investigadores do Instituto Politécnico de Bragança (EJI 2016)*
  http://eji.ipb.pt/
- *XVII ENCUENTRO AECA (AECA 2016)*
  http://xviiencuentroaeca.ipb.pt/
- *VII Congresso Mundial de Estilos de Aprendizagem (CMEA 2016)*
  http://cmea.ipb.pt/
- *XVI Festival Nacional de Robótica 2016*
  https://robotica2016.ipb.pt/

Projects Advisor

- *Plataforma de maximização de aproveitamento fotovoltaico*, (2022)
  http://projinf.estig.ipb.pt/~a48261
- *IPB Student Mobile App*, (2022)
  http://projinf.estig.ipb.pt/~a31611a40286
- *IPBeacons at Campus - Mobile Application*, (2021)
  http://projinf.estig.ipb.pt/~a40528a40534
- *IPB mobile*, (2021)
  http://projinf.estig.ipb.pt/~a35468a33884
- *Dashboard para análise de desempenho académico*, (2021)
  http://projinf.estig.ipb.pt/~a39964a37912
- *IPBeacons no Campus – Aplicação móvel*, (2020)
  http://projinf.estig.ipb.pt/~a36749a39236

- *Plataforma de Gestão Integrada de Redes Sociais*, (2020)
  http://projinf.estig.ipb.pt/~a36229
- *Deteção automatizada e não invasiva de utilizadores num ambiente*, (2019)
  http://projinf.estig.ipb.pt/~a35471

5.6 Limitations

Regarding limitations, some can be highlighted, namely because this is a doctorate in a company, and with the company perspectives of commercializing the solution and patenting the work developed. There were some constraints derived from industrial secrecy, namely the possibility of implementing the prototype for testing in uncontrolled public access places. In this sense, tests were carried out in a domestic, and in a professional environment (office) belonging to the company.

Also regarding the number of users needed and the respective information, as highlighted in section 3.2, due to the extreme difficulty of attracting sufficient number of users for validation, especially also with the pandemic effects, a simulation was carried out with the user data, which allowed testing the MAS with data as real as possible and his statistically validation.

This limitation was thus overcome, which was also associated with the lack of hardware (*Raspberry's*) that would represent the local systems and would collect all the necessary information, because for the amount of information needed (hundreds, one for each frequented place by each user).

5.7 Future Research Directions

As any thesis, this one can be improved and evolved in different directions. Considering the results achieved in this thesis and the scientific contributions presented, it is possible to address some future research directions to complement and extend this thesis. During this thesis development, different research lines have been proposed and explored, in this way it can be more evolved, improved or also viewed with a different perspective.

Namely, for future research, the aim is to continue analyzing human and climate factors that affects the comfort. For instance user distance to the actuators is a factor that can be optimized, because is understood that this distance affects thermic sensation that user gets, and also for instance the sound, or other preferences.

Another work can be done according to maximize user comfort achieving the best energy savings. In this case further research is required to provide satisfactory (comfort/energy savings) agreements. Gamification can be one technique to incorporate, in a way that user has involvement in obtaining rewards for efficient energy behaviour, this can create a competition environment by the different users at the same space, and in this way the energy efficiency can be improved, and at the same way the consumption reduced.

Also is important to refer the use of geofencing techniques to start predict user behaviors, and to anticipate is presence at the environments, in that way the environment can do an automatic adjustment to accommodate best comfort when user arrives, minimizing all actuators latencies that normally exist. Also with the user history, and application of AI models it can be performed the user routines discovery, with this we can surpass the use of more invasive techniques like geofencing.

Also nowadays, we can use this kind of techniques, to see if a user is doing its normal routine, namely by inspecting what is the distance from the user current location to the next space where is expected to enter. If this distance is much more high than the expected, we can predict that the user is outside is routine, for instance it can be traveling.

With this information, it can be achieved a predictive comfort as described at subsection 3.1.7, reducing significantly the consumption, because it can be predicted when the user is outside its normal routine, and for instance in that way we can turn off an entire day of climate control, if the system assess that the user doesn't go to the office that day.

This solution can also evolve, for using during user mobility, namely when user is driving at his own private car, or in public transports. The same principle apply in this kind of environments, and the preferences card can in the same way be shared with the vehicle, and the vehicle do his own adaptation according to the present users.

The proposed solution can also be seen integrated with the big players platforms like *Google*, *Amazon* and *Apple*. Because as identified at subsection 2.2.2, users already have this kind of devices (*Apple Homepod*, *Apple TV*, *Amazon Echo*, *Google Nest*, and others) at their homes. In this way, this devices can be used to serve as local system's, and integrate the proposed solution at their platforms described at subsection 2.2.4, like: *Alexa*, *Homekit*, or *Android Things*. This kind of integration, it will reduce costs to the user, and it will be a big achievement to the company, at the product commercialization.

Bibliography

[1] E. Aarts and F. Grotenhuis. “Ambient intelligence 2.0: Towards synergetic prosperity”. In: *Journal of Ambient Intelligence and Smart Environments* 3.1 (2011), pp. 3–11 (cit. on p. 1).

[2] E. H. Aarts and J. L. Encarnação. *True visions: The emergence of ambient intelligence*. Springer Science & Business Media, 2006 (cit. on p. 37).

[3] E. H. Aarts and B. E. de Ruyter. “New research perspectives on Ambient Intelligence.” In: *JAISE* 1.1 (2009), pp. 5–14 (cit. on pp. 1, 37).

[4] L. Aarts E. Appelo. “Ambient Intelligence: thuisomgevingen van de toekomst”. In: *IT Monitor 9/1999* (1999) (cit. on p. 38).

[5] J. Al-Muhtadi et al. “Cerberus: a context-aware security scheme for smart spaces”. In: *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003.(PerCom 2003)*. IEEE. 2003, pp. 489–496 (cit. on p. 87).

[6] V Alfonso et al. *Market Trends: TSPs Must Invest in the Rapidly Evolving IoT Ecosystems Now*. 2013 (cit. on p. 2).

[7] W.-F. Alliance. “Wi-fi certified wi-fi direct”. In: *White paper* (2010) (cit. on p. 26).

[8] Z. Alliance. “Zigbee alliance”. In: *WPAN industry group, http://www. zigbee. org/. The industry group responsible for the ZigBee standard and certification* (2010) (cit. on p. 28).

[9] Amazon. *Amazon Alexa*. https://developer.amazon.com/en-US/alexa. 2020-05 (cit. on p. 21).

[10] J. P. B. Andrade et al. “Uma abordagem com sistemas multiagentes para controle autônomo de casas inteligentes”. In: *XIII Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)* (2016) (cit. on p. 69).

[11] Apple. *Apple Homepod*. https://www.apple.com/homepod-2018/. 2020-05 (cit. on p. 21).

[12] J. Aron. “How innovative is Apple’s new voice assistant, Siri?” In: *New Scientist* 212.2836 (2011), p. 24 (cit. on p. 31).

[13] J. C. Augusto et al. “Intelligent environments: a manifesto”. In: *Human-Centric Computing and Information Sciences* 3.1 (2013), pp. 1–18 (cit. on p. 1).

[14] J. Augusto Wrede and P. Mccullagh. “Ambient Intelligence: Concepts and Applications”. In: *Comput. Sci. Inf. Syst.* 4 (2007-01), pp. 1–27. doi: 10.2298/CSIS0701001A (cit. on p. 37).

[15] A. Aztiria. “Learning frequent behaviours of the users in intelligent environments”. In: *Journal of Ambient Intelligence and Smart Environments* 2.4 (2010), pp. 435–436 (cit. on p. 1).

[16] A. Aztiria, A. Izaguirre, and J. C. Augusto. “Learning patterns in ambient intelligence environments: a survey”. In: *Artificial Intelligence Review* 34.1 (2010), pp. 35–51 (cit. on pp. 1, 49, 54).

[17] A. Aztiria et al. “Discovering frequent user–environment interactions in intelligent environments”. In: *Personal and Ubiquitous Computing* 16.1 (2012), pp. 91–103 (cit. on p. 1).

[18] S. Babar et al. “Proposed security model and threat taxonomy for the internet of things (IoT)”. In: *Recent Trends in Network Security and Applications*. Springer, 2010, pp. 420–429 (cit. on pp. 58, 79).

[19] M. Baldauf, S. Dustdar, and F. Rosenberg. “A survey on context-aware systems”. In: *International Journal of Ad Hoc and Ubiquitous Computing* 2.4 (2007), pp. 263–277 (cit. on p. 51).

[20] D. Bandyopadhyay and J. Sen. “Internet of things: Applications and challenges in technology and standardization”. In: *Wireless Personal Communications* 58.1 (2011), pp. 49–69 (cit. on pp. 2, 9, 10).

[21] F. Bellifemine, A. Poggi, and G. Rimassa. “Developing multi-agent systems with JADE”. In: *Intelligent Agents VII Agent Theories Architectures and Languages*. Springer, 2001, pp. 89–103 (cit. on pp. 6, 74).

[22] F. L. Bellifemine, G. Caire, and D. Greenwood. *Developing multi-agent systems with JADE*. Vol. 7. John Wiley & Sons, 2007 (cit. on p. 6).

[23] K.-I. Benta et al. “Agent based smart house platform with affective control”. In: *Proceedings of the 2009 Euro American Conference on Telematics and Information Systems: New Opportunities to increase Digital Citizenship*. 2009, pp. 1–7 (cit. on p. 69).

[24] R. M. Bergner. “What is behavior? And so what?” In: *New ideas in psychology* 29.2 (2011), pp. 147–155 (cit. on p. 46).

[25] S. Bluetooth. “Bluetooth Core Specification Version 4.0”. In: *Specification of the Bluetooth System* (2010) (cit. on pp. 23, 55, 64).

[26] R. H. Bordini, J. F. Hübner, and M. Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*. Vol. 8. John Wiley & Sons, 2007 (cit. on p. 69).

[27] J. Cámara et al. “Reasoning about sensing uncertainty and its reduction in decision-making for self-adaptation”. In: *Science of Computer Programming* 167 (2018), pp. 51–69 (cit. on p. 47).

[28] A. Campbell. "The future of bacteriophage biology". In: *Nature Reviews Genetics* 4.6 (2003), pp. 471–477 (cit. on p. 46).

[29] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano. "Device-to-device communications with Wi-Fi Direct: overview and experimentation". In: *Wireless Communications, IEEE* 20.3 (2013), pp. 96–104 (cit. on p. 64).

[30] M. Cárdenas, J. G. Sanz, and J. Pavón. "Testing Ambient Assisted Living Solutions with Simulations". In: *IFIP International Conference on Testing Software and Systems*. Springer. 2018, pp. 56–61 (cit. on p. 57).

[31] D. Carneiro and P. Novais. *Behavioral Biometrics and Ambient Intelligence: New Opportunities for Context-Aware Applications*. 2017 (cit. on p. 46).

[32] D. Carneiro et al. "Enriching conflict resolution environments with the provision of context information". In: *Expert Systems* (2013) (cit. on p. 2).

[33] A.-C. Chaouche et al. "A higher-order agent model with contextual planning management for ambient systems". In: *Transactions on Computational Collective Intelligence XVI*. Springer, 2014, pp. 146–169 (cit. on p. 69).

[34] Y. P. Chaubey. *Resampling-based multiple testing: Examples and methods for p-value adjustment*. 1993 (cit. on p. 62).

[35] E. S. Chen et al. "PalmCIS: a wireless handheld application for satisfying clinician information needs". In: *Journal of the American Medical Informatics Association* 11.1 (2004), pp. 19–28 (cit. on p. 18).

[36] H. Chen et al. "Intelligent agents meet the semantic web in smart spaces". In: *IEEE Internet computing* 8.6 (2004), pp. 69–79 (cit. on p. 87).

[37] M. Chui, M. Löffler, and R. Roberts. "The internet of things". In: *McKinsey Quarterly* 2.2010 (2010), pp. 1–9 (cit. on pp. 2, 10).

[38] D. J. Cook. "Learning setting-generalized activity models for smart spaces". In: *IEEE intelligent systems* 2010.99 (2010), p. 1 (cit. on p. 1).

[39] D. J. Cook, J. C. Augusto, and V. R. Jakkula. "Ambient intelligence: Technologies, applications, and opportunities". In: *Pervasive and Mobile Computing* 5.4 (2009), pp. 277–298 (cit. on pp. 37, 44).

[40] C. K. Crutzen. "Invisibility and the meaning of ambient intelligence". In: *The International Review of Information Ethics* 6 (2006), pp. 52–62 (cit. on p. 37).

[41] K. Curran, A. Millar, and C. Mc Garvey. "Near field communication". In: *International Journal of Electrical and Computer Engineering (IJECE)* 2.3 (2012), pp. 371–382 (cit. on pp. 25, 26).

[42] S. K. Das et al. “The role of prediction algorithms in the MavHome smart home architecture”. In: *IEEE Wireless Communications* 9.6 (2002), pp. 77–84 (cit. on p. 87).

[43] H. De Man. “Ambient intelligence: gigascale dreams and nanoscale realities”. In: *Solid-State Circuits Conference, 2005. Digest of Technical Papers. ISSCC. 2005 IEEE International*. IEEE. 2005, pp. 29–35 (cit. on p. 44).

[44] A. De Paola et al. “Sensor 9 k: A testbed for designing and experimenting with WSN-based ambient intelligence applications”. In: *Pervasive and Mobile Computing* 8.3 (2012), pp. 448–466 (cit. on p. 41).

[45] P. Díaz et al. “The ecosense project: An intelligent energy management system with a wireless sensor and actor network”. In: *Sustainability in Energy and Buildings*. Springer, 2011, pp. 237–245 (cit. on p. 41).

[46] T. Dierks. “The transport layer security (TLS) protocol version 1.2”. In: *IEEE* (2008) (cit. on p. 83).

[47] J. Donovan. *Bluetooth Low-Energy: An Introduction*. 2014. url: http://low-powerwireless.com/blog/2010/07/08/bluetooth-low-energy-an-introduction/ (visited on 2015-11-10) (cit. on p. 23).

[48] K. Ducatel et al. “Scenarios for ambient intelligence in 2010”. In: (2001) (cit. on pp. 37, 39).

[49] D. Evans. “The internet of things: How the next evolution of the internet is changing everything”. In: *CISCO white paper* 1 (2011), pp. 13–14 (cit. on p. 10).

[50] E. Fleisch et al. “What is the internet of things? An economic perspective”. In: *Economics, Management, and Financial Markets* 5.2 (2010), pp. 125–157 (cit. on pp. 1, 10).

[51] M. Friedewald et al. “The brave new world of ambient intelligence: An analysis of scenarios regarding privacy, identity and security issues”. In: *Security in Pervasive Computing*. Springer, 2006, pp. 119–133 (cit. on p. 44).

[52] M. Frontczak and P. Wargocki. “Literature survey on how different factors influence human comfort in indoor environments”. In: *Building and environment* 46.4 (2011), pp. 922–937 (cit. on p. 50).

[53] R. M. Furr. “Personality psychology as a truly behavioural science”. In: *European Journal of Personality* 23.5 (2009), pp. 369–401 (cit. on p. 46).

[54] K. Gama, L. Touseau, and D. Donsez. “Combining heterogeneous service technologies for building an Internet of Things middleware”. In: *Computer Communications* 35.4 (2012), pp. 405–417 (cit. on pp. 1, 9).

[55] B. Gardner, G.-J. de Bruijn, and P. Lally. “A systematic review and meta-analysis of applications of the self-report habit index to nutrition and physical activity behaviours”. In: *Annals of Behavioral Medicine* 42.2 (2011), pp. 174–187 (cit. on p. 45).

[56] B. Gardner, G.-J. de Bruijn, and P. Lally. “Habit, identity, and repetitive action: A prospective study of binge-drinking in UK students”. In: *British journal of health psychology* 17.3 (2012), pp. 565–581 (cit. on p. 45).

[57] O. Ghag and S. Hegde. “A Comprehensive Study of Google Wallet as an NFC Application”. In: *International Journal of Computer Applications* 58.16 (2012), pp. 37–42 (cit. on p. 25).

[58] D. Gislason. *Zigbee wireless networking*. Newnes, 2008 (cit. on p. 65).

[59] M. Gomes et al. “Studying the effects of stress on negotiation behavior”. In: *Cybernetics and Systems* 45.3 (2014), pp. 279–291 (cit. on pp. 1, 2).

[60] C. Gomez, J. Oller, and J. Paradells. “Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology”. In: *Sensors* 12.9 (2012), pp. 11734–11753 (cit. on p. 79).

[61] D.-M. Han and J.-H. Lim. “Design and implementation of smart home energy management systems based on zigbee”. In: *IEEE Transactions on Consumer Electronics* 56.3 (2010), pp. 1417–1425 (cit. on p. 65).

[62] E. Haselsteiner and K. Breitfuß. “Security in near field communication (NFC)”. In: *Workshop on RFID security*. 2006, pp. 12–14 (cit. on p. 79).

[63] H. Hassani. “Research methods in computer science: The challenges and issues”. In: *arXiv preprint arXiv:1703.04080* (2017) (cit. on p. 7).

[64] S. Helal et al. “The gator tech smart house: A programmable pervasive space”. In: *Computer* 38.3 (2005), pp. 50–60 (cit. on p. 87).

[65] A. R. Hevner. “A three cycle view of design science research”. In: *Scandinavian journal of information systems* 19.2 (2007), p. 4 (cit. on p. 7).

[66] A. R. Hevner et al. “Design science in information systems research”. In: *MIS quarterly* (2004), pp. 75–105 (cit. on p. 7).

[67] R. A. Hinde. “Animal behavior: a synthesis of ethology and comparative psychology.” In: (1966) (cit. on p. 45).

[68] P. Hoes et al. “User behavior in whole building simulation”. In: *Energy and buildings* 41.3 (2009), pp. 295–302 (cit. on p. 57).

[69] P. HomeLab. “365 days’ Ambient Intelligent research in HomeLab”. In: *Philips Research. April 2003* (2003) (cit. on p. 42).

[70] A. S. Huang and L. Rudolph. *Bluetooth essentials for programmers*. Cambridge University Press, 2007 (cit. on p. 23).

[71] O. IDC-Smartphone. *market share 2014*. 2015 (cit. on p. 18).

[72] B. Insider. *THE INTERNET OF THINGS 2015 REPORT: Examining how the IoT will affect the world*. 2015. url: http://www.businessinsider.com/internet-of-things-2015-forecasts-of-the-industrial-iot-connected-home-and-more-2015-10 (visited on 2015-11-10) (cit. on pp. 11–15).

[73] S. Iyengar, R. R. Brooks, and G Karjoth. "International journal of distributed sensor networks". In: *International Journal of Distributed Sensor Networks* 4.1 (2008), pp. 1–3 (cit. on p. 2).

[74] E. Kazanavicius, V. Kazanavicius, and L. Ostaseviciute. "Agent-based framework for embedded systems development in smart environments". In: *Proceedings of the 15th International Conference on Information and Software Technologies IT*. 2009, pp. 194–200 (cit. on p. 69).

[75] R. Khan et al. "Future internet: the internet of things architecture, possible applications and key challenges". In: *Frontiers of Information Technology (FIT), 2012 10th International Conference on*. IEEE. 2012, pp. 257–260 (cit. on p. 78).

[76] H.-S. Kim, S. Kumar, and D. E. Culler. "Thread/OpenThread: A compromise in low-power wireless multihop network architecture for the Internet of Things". In: *IEEE Communications Magazine* 57.7 (2019), pp. 55–61 (cit. on pp. 30, 34).

[77] M. Knight. "How safe is Z-Wave?[Wireless standards]". In: *Computing and Control Engineering* 17.6 (2006), pp. 18–23 (cit. on p. 29).

[78] S. Konishi and G. Kitagawa. *Information criteria and statistical modeling*. Springer Science & Business Media, 2008 (cit. on p. 61).

[79] J.-y. Kwak et al. "SAVES: A sustainable multiagent application to conserve building energy considering occupants". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2012, pp. 21–28 (cit. on p. 41).

[80] P Leach, M. Mealling, and R. Salz. "RFC 4122: A universally unique identifier (UUID) URN namespace". In: *Proposed Standard, July* (2005) (cit. on p. 83).

[81] N. Lee. *Bluetooth 4.0: What is it, and does it matter?* 2011. url: http://www.cnet.com/news/bluetooth-4-0-what-is-it-and-does-it-matter/ (visited on 2015-11-10) (cit. on p. 23).

[82] D. A. Levitis, W. Z. Lidicker Jr, and G. Freund. "Behavioural biologists do not agree on what constitutes behaviour". In: *Animal behaviour* 78.1 (2009), pp. 103–110 (cit. on p. 46).

[83] D. Li, C. C. Menassa, and V. R. Kamat. "Personalized human comfort in indoor building environments under diverse conditioning modes". In: *Building and Environment* 126 (2017), pp. 304–317 (cit. on p. 50).

[84] S. Long et al. “Rapid prototyping of mobile context-aware applications: The cyberguide case study”. In: *Proceedings of the 2nd annual international conference on Mobile computing and networking*. 1996, pp. 97–107 (cit. on p. 47).

[85] H.-D. Ma. “Internet of things: Objectives and scientific challenges”. In: *Journal of Computer science and Technology* 26.6 (2011), pp. 919–924 (cit. on pp. 9, 10).

[86] E. Maeda et al. “The World of Mushrooms—Transdisciplinary Approach to Human-Computer Interaction with Ambient Intelligence”. In: *NTT Technical Review* 4.12 (2006), pp. 17–25 (cit. on p. 37).

[87] J. Manyika et al. *The Internet Of Things: Mapping The Value Beyond The Hype*. 2015 (cit. on pp. 2, 11).

[88] R. Martins and F. Meneguzzi. “A smart home model to demand side management”. In: *Workshop on Collaborative Online Organizations (COOS’13)@ AAMAS*. 2013 (cit. on p. 69).

[89] R. Martins and F. Meneguzzi. “A smart home model using JaCaMo framework”. In: *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*. IEEE. 2014, pp. 94–99 (cit. on p. 69).

[90] S. D. McArthur et al. “Multi-agent systems for power engineering applications—Part II: Technologies, standards, and tools for building multi-agent systems”. In: *Power Systems, IEEE Transactions on* 22.4 (2007), pp. 1753–1759 (cit. on pp. 6, 56).

[91] F. Mish and J. Morse. “Merriam-Webster’s Collegiate Dictionary, tenth edition”. In: *Springfield: Merriam-Webster, Inc., p. 36* (1990) (cit. on p. 37).

[92] G. Mone. “Intelligent living”. In: *Communications of the ACM* 57.12 (2014), pp. 15–16 (cit. on p. 32).

[93] G. W. Musumba and H. O. Nyongesa. “Context awareness in mobile computing: A review”. In: *International Journal of Machine Learning and Applications* 2.1 (2013), p. 5 (cit. on p. 47).

[94] P. Newman. “HOW THE IoT CONTINUES TO TRANSFORM BUSINESS, HOMES, AND CITIES THROUGH NEXT-GENERATION DIGITAL SOLUTIONS”. In: *Business Insider* (2019). url: https://store.businessinsider.com/products/the-internet-of-things-report (visited on 2019-06-06) (cit. on pp. 2, 15–18).

[95] P. Nilsen et al. “Creatures of habit: accounting for the role of habit in implementation research on clinical behaviour change”. In: *Implementation Science* 7.1 (2012), pp. 1–6 (cit. on p. 45).

[96] P. Nixon, S. Dobson, and G. Lacey. “Managing smart environments”. In: *Proceedings of the Workshop on Software Engineering for Wearable and Pervasive Computing*. 2000 (cit. on p. 87).

[97] P. Novais et al. “Inter-organization cooperation for ambient assisted living”. In: *Journal of Ambient Intelligence and Smart Environments* 2.2 (2010), pp. 179–195 (cit. on p. 44).

[98] P. Novais et al. “A framework for monitoring and assisting seniors with memory disabilities”. In: *Ambient Assisted Living* (2011) (cit. on p. 39).

[99] P. Oliveira, P. Matos, and P. Novais. “Behaviour analysis in smart spaces”. In: *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE. 2016, pp. 880–887 (cit. on p. 4).

[100] P. Oliveira, P. Novais, and P. Matos. “Challenges in Smart Spaces: Aware of users, preferences, behaviours and habits”. In: *International Conference on Practical Applications of Agents and Multi-Agent Systems*. Springer. 2017, pp. 268–271 (cit. on p. 4).

[101] P. Oliveira et al. “Towards to Secure an IoT Adaptive Environment System”. In: *International Symposium on Distributed Computing and Artificial Intelligence*. Springer. 2018, pp. 349–352 (cit. on p. 78).

[102] J. A. Ouellette and W. Wood. “Habit and intention in everyday life: The multiple processes by which past behavior predicts future behavior.” In: *Psychological bulletin* 124.1 (1998), p. 54 (cit. on p. 45).

[103] A. Pantelopoulos and N. G. Bourbakis. “A survey on wearable sensor-based systems for health monitoring and prognosis”. In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 40.1 (2010), pp. 1–12 (cit. on p. 19).

[104] K. Peffers et al. “Design science research evaluation”. In: *International Conference on Design Science Research in Information Systems*. Springer. 2012, pp. 398–410 (cit. on p. 7).

[105] C. Perera et al. “Privacy of Big Data in the Internet of Things Era”. In: *arXiv preprint arXiv:1412.8339* (2014) (cit. on p. 79).

[106] A. Pichot. “Qu’est-ce que le comportement?” In: *Revue européenne des sciences sociales* 37.115 (1999), pp. 117–126 (cit. on p. 46).

[107] A. Pinto et al. “eHealthCare-A Medication Monitoring Approach for the Elderly People”. In: *International Conference on Wireless Mobile Communication and Healthcare*. Springer. 2022, pp. 221–234 (cit. on p. 13).

[108] D. Preuveneers and P. Novais. “A survey of software engineering best practices for the development of smart applications in Ambient Intelligence”. In: *Journal of Ambient Intelligence and Smart Environments* 4.3 (2012), pp. 149–162 (cit. on p. 1).

[109] J. Rech and K.-D. Althoff. “Artificial intelligence and software engineering: Status and future trends”. In: *KI* 18.3 (2004), pp. 5–11 (cit. on p. 37).

[110] B. Reed. “A brief history of smartphones”. In: *PC World* (2010) (cit. on p. 18).

[111] P. Regulation. “Regulation (EU) 2016/679 of the European Parliament and of the Council”. In: *Regulation (eu)* 679 (2016), p. 2016 (cit. on p. 78).

[112] A Rehman et al. “Survey of wearable sensors with comparative study of noise reduction ecg filters”. In: *Int. J. Com. Net. Tech* 1.1 (2013), pp. 61–81 (cit. on pp. 20, 49).

[113] A. Riahi et al. “A systemic approach for IoT security”. In: *Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on*. IEEE. 2013, pp. 351–355 (cit. on p. 79).

[114] J. van’t Riet et al. “The importance of habits in eating behaviour. An overview and recommendations for future research”. In: *Appetite* 57.3 (2011), pp. 585–596 (cit. on p. 45).

[115] J. A. Rincon et al. “A new emotional robot assistant that facilitates human interaction and persuasion”. In: *Knowledge and Information Systems* 60.1 (2019), pp. 363–383 (cit. on p. 39).

[116] F. Rivera-Illingworth, V. Callaghan, and H. Hagras. “Detection of normal and novel behaviours in ubiquitous domestic environments”. In: *The Computer Journal* 53.2 (2010), pp. 142–151 (cit. on p. 1).

[117] L. Rosenthal and V. Stanford. “NIST smart space: pervasive computing initiative”. In: *Proceedings IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE 2000)*. IEEE. 2000, pp. 6–11 (cit. on p. 87).

[118] S. J. Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010 (cit. on p. 38).

[119] P. Salvo et al. “A wearable sensor for measuring sweat rate”. In: *Sensors Journal, IEEE* 10.10 (2010), pp. 1557–1558 (cit. on p. 19).

[120] T Sánchez López et al. “Adding sense to the Internet of Things-An architecture framework for Smart Object systems”. In: *Personal and Ubiquitous Computing* (2011), pp. 1–18 (cit. on p. 9).

[121] M. Satyanarayanan. “Pervasive computing: Vision and challenges”. In: *IEEE Personal communications* 8.4 (2001), pp. 10–17 (cit. on p. 87).

[122] B. Schilit, N. Adams, and R. Want. “Context-aware computing applications”. In: *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*. IEEE. 1994, pp. 85–90 (cit. on pp. 51, 57).

[123] B. Sharma and M. S. Obaidat. “Comparative analysis of IoT based products, technology and integration of IoT with cloud computing”. In: *IET Networks* 9.2 (2020), pp. 43–47 (cit. on pp. 32, 33).

[124] H. A. Simon. *The Sciences of the Artificial, reissue of the third edition with a new introduction by John Laird*. MIT press, 2019 (cit. on p. 7).

[125] P. Spachos and K. Plataniotis. “BLE beacons in the smart city: Applications, challenges, and research opportunities”. In: *IEEE Internet of Things Magazine* 3.1 (2020), pp. 14–18 (cit. on p. 22).

[126] M. F. Stabile and J. S. Sichman. “Evaluating perception filters in BDI Jason agents”. In: *2015 Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE. 2015, pp. 116–121 (cit. on p. 70).

[127] W. Storms, J. Shockley, and J. Raquet. “Magnetic field navigation in an indoor environment”. In: *Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2010*. IEEE. 2010, pp. 1–10 (cit. on p. 50).

[128] N. Streitz et al. “Grand challenges for ambient intelligence and implications for design contexts and smart societies”. In: *Journal of Ambient Intelligence and Smart Environments* 11.1 (2019), pp. 87–107 (cit. on p. 78).

[129] J. A. Tauber et al. “Indoor location systems for pervasive computing”. In: *Massachusetts Institute of Technology. Area exam report* (2002) (cit. on p. 50).

[130] N. Tinbergen. “On aims and methods of ethology”. In: *Zeitschrift für tierpsychologie* 20.4 (1963), pp. 410–433 (cit. on p. 45).

[131] I. Unwala, Z. Taqvi, and J. Lu. “Thread: An iot protocol”. In: *2018 IEEE Green Technologies Conference (GreenTech)*. IEEE. 2018, pp. 161–167 (cit. on pp. 30, 31).

[132] A. Vasilakos and W. Pedrycz. *Ambient intelligence, wireless networking, and ubiquitous computing*. Artech House, Inc., 2006 (cit. on p. 37).

[133] B. Verplanken and W. Wood. “Interventions to break and create consumer habits”. In: *Journal of public policy & marketing* 25.1 (2006), pp. 90–103 (cit. on p. 45).

[134] J. Villeneuve. *Survey of Home Automation Standards*. Tech. rep. University of North Texas, 2014 (cit. on p. 32).

[135] K. I.-K. Wang, W. H. Abdulla, and Z. Salcic. “Ambient intelligence platform using multi-agent system and mobile ubiquitous hardware”. In: *Pervasive and Mobile Computing* 5.5 (2009), pp. 558–573 (cit. on pp. 51, 57).

[136] R. Want. “Near field communication”. In: *IEEE Pervasive Computing* 1.3 (2011), pp. 4–7 (cit. on pp. 24, 26, 64).

[137] M. Weiser. “The computer for the 21st century”. In: *Scientific american* 265.3 (1991), pp. 94–104 (cit. on p. 38).

[138] G. Wenninger. “Lexicon der Psychologie (deel 2).” In: (2001) (cit. on p. 46).

[139] R. West and J. Brown. “Theory of addiction”. In: (2013) (cit. on p. 45).

[140] D. White and A. Rea. *Server Hardening Tactics for Increased Security*. Tech. rep. Working Paper, 2003 (cit. on p. 83).

[141] W. Wood and D. T. Neal. “The habitual consumer”. In: *Journal of Consumer Psychology* 19.4 (2009), pp. 579–592 (cit. on p. 45).

[142] M. Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009 (cit. on p. 69).

[143] D. Wright. “The dark side of ambient intelligence”. In: *info* 7.6 (2005), pp. 33–51 (cit. on p. 44).

[144] D. Yan et al. “Occupant behavior modeling for building performance simulation: Current state and future challenges”. In: *Energy and Buildings* 107 (2015), pp. 264–278 (cit. on p. 57).

[145] G. M. Youngblood, D. J. Cook, and L. B. Holder. “Managing adaptive versatile environments”. In: *Pervasive and Mobile Computing* 1.4 (2005), pp. 373–403 (cit. on p. 39).

[146] E Zelkha. “The future of information appliances and consumer devices”. In: *Palo Alto Ventures, Palo Alto, California* (1998) (cit. on p. 37).

[147] G. Zimmerman. “Modeling and simulation of individual user behavior for building performance predictions”. In: *Proceedings of the 2007 Summer Computer Simulation Conference*. Society for Computer Simulation International. 2007, pp. 913–920 (cit. on p. 57).

A

SQL Scripts

A.1 Create_DB_PhD

```
1
2  SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
3  SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
4  SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
5
6  -- -----------------------------------------------------
7  -- Schema DB_DataLayer_PhD
8  -- -----------------------------------------------------
9  DROP SCHEMA IF EXISTS `DB_DataLayer_PhD` ;
10
11 -- -----------------------------------------------------
12 -- Schema DB_DataLayer_PhD
13 -- -----------------------------------------------------
14 CREATE SCHEMA IF NOT EXISTS `DB_DataLayer_PhD` DEFAULT CHARACTER SET utf8 ;
15 USE `DB_DataLayer_PhD` ;
16
17 -- -----------------------------------------------------
18 -- Table `DB_DataLayer_PhD`.`User_Information`
19 -- -----------------------------------------------------
20 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`User_Information` ;
21
22 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`User_Information` (
23 `idUser` INT NOT NULL AUTO_INCREMENT,
24 `User_UUID` CHAR(36) NOT NULL,
25 PRIMARY KEY (`idUser`),
26 UNIQUE INDEX `User_UUID_UNIQUE` (`User_UUID` ASC),
27 UNIQUE INDEX `idUser_UNIQUE` (`idUser` ASC))
28 ENGINE = InnoDB;
29
30
```

```
31 -- -----------------------------------------------------
32 -- Table `DB_DataLayer_PhD`.`Preferences_IDS`
33 -- -----------------------------------------------------
34 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Preferences_IDS` ;
35
36 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Preferences_IDS` (
37 `idPreference` INT NOT NULL,
38 `Description_Preference` VARCHAR(45) NULL,
39 `Unities_Preference` VARCHAR(30) NULL,
40 `Data_Type_Preference` VARCHAR(45) NULL,
41 PRIMARY KEY (`idPreference`))
42 ENGINE = InnoDB;
43
44
45 -- -----------------------------------------------------
46 -- Table `DB_DataLayer_PhD`.`Preference_Rank`
47 -- -----------------------------------------------------
48 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Preference_Rank` ;
49
50 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Preference_Rank` (
51 `idPreference_Rank` INT NOT NULL,
52 `Pref_Rank_Description` VARCHAR(45) NULL,
53 PRIMARY KEY (`idPreference_Rank`))
54 ENGINE = InnoDB;
55
56
57 -- -----------------------------------------------------
58 -- Table `DB_DataLayer_PhD`.`Preferences_Card`
59 -- -----------------------------------------------------
60 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Preferences_Card` ;
61
62 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Preferences_Card` (
63 `ID_UserFk` INT NOT NULL,
64 `ID_PrefsFk` INT NOT NULL COMMENT ' ',
65 `Preference_Value` VARCHAR(45) NULL,
66 `Preference_Rank` INT NOT NULL,
67 PRIMARY KEY (`ID_UserFk`, `ID_PrefsFk`, `Preference_Rank`),
68 INDEX `ID_User_idx` (`ID_UserFk` ASC),
69 INDEX `ID_Prefs_idx` (`ID_PrefsFk` ASC),
70 INDEX `ID_Pref_Rank_FK_idx` (`Preference_Rank` ASC),
71 CONSTRAINT `ID_UserFK`
72 FOREIGN KEY (`ID_UserFk`)
73 REFERENCES `DB_DataLayer_PhD`.`User_Information` (`idUser`)
74 ON DELETE NO ACTION
75 ON UPDATE NO ACTION,
```

```
76  CONSTRAINT `ID_PrefsFK`
77  FOREIGN KEY (`ID_PrefsFk`)
78  REFERENCES `DB_DataLayer_PhD`.`Preferences_IDS` (`idPreference`)
79  ON DELETE NO ACTION
80  ON UPDATE NO ACTION,
81  CONSTRAINT `ID_Pref_Rank_FK`
82  FOREIGN KEY (`Preference_Rank`)
83  REFERENCES `DB_DataLayer_PhD`.`Preference_Rank` (`idPreference_Rank`)
84  ON DELETE NO ACTION
85  ON UPDATE NO ACTION)
86  ENGINE = InnoDB;
87
88
89  -- -----------------------------------------------------
90  -- Table `DB_DataLayer_PhD`.`Local_System_Information`
91  -- -----------------------------------------------------
92  DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Local_System_Information` ;
93
94  CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Local_System_Information` (
95  `idLocal_System` INT NOT NULL,
96  `Desc_LocalSystem` VARCHAR(45) NULL,
97  `GPS_Latitude` VARCHAR(45) NULL,
98  `GPS_Longitude` VARCHAR(45) NULL,
99  PRIMARY KEY (`idLocal_System`))
100 ENGINE = InnoDB;
101
102
103 -- -----------------------------------------------------
104 -- Table `DB_DataLayer_PhD`.`History_Pref_User`
105 -- -----------------------------------------------------
106 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`History_Pref_User` ;
107
108 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`History_Pref_User` (
109 `ID_User_FK` INT NOT NULL,
110 `ID_System_FK` INT NOT NULL,
111 `Time` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '    ',
112 `Time_Period` TIME NOT NULL,
113 `ID_Prefs_FK` INT NOT NULL COMMENT '  ',
114 `Preference_Value` VARCHAR(45) NULL,
115 `UserUpdateValue` INT NULL DEFAULT 0,
116 PRIMARY KEY (`ID_User_FK`, `ID_System_FK`, `Time`, `ID_Prefs_FK`),
117 INDEX `ID_System_idx` (`ID_System_FK` ASC),
118 INDEX `ID_User_idx` (`ID_User_FK` ASC),
119 INDEX `ID_Prefs_idx` (`ID_Prefs_FK` ASC),
120 CONSTRAINT `ID_System_FK`
```

```
121 FOREIGN KEY (`ID_System_FK`)
122 REFERENCES `DB_DataLayer_PhD`.`Local_System_Information` (`idLocal_System`)
123 ON DELETE NO ACTION
124 ON UPDATE NO ACTION,
125 CONSTRAINT `ID_User_FK`
126 FOREIGN KEY (`ID_User_FK`)
127 REFERENCES `DB_DataLayer_PhD`.`User_Information` (`idUser`)
128 ON DELETE NO ACTION
129 ON UPDATE NO ACTION,
130 CONSTRAINT `ID_Prefs_FK`
131 FOREIGN KEY (`ID_Prefs_FK`)
132 REFERENCES `DB_DataLayer_PhD`.`Preferences_IDS` (`idPreference`)
133 ON DELETE NO ACTION
134 ON UPDATE NO ACTION)
135 ENGINE = InnoDB;
136
137
138 -- -----------------------------------------------------
139 -- Table `DB_DataLayer_PhD`.`Utilization_Preferences`
140 -- -----------------------------------------------------
141 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Utilization_Preferences` ;
142
143 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Utilization_Preferences` (
144 `User_ID_FK` INT NOT NULL,
145 `Local_ID_FK` INT NOT NULL,
146 `Time_Period` TIME NOT NULL,
147 `ID_Prefs_FK` INT NOT NULL COMMENT '    ',
148 `Preference_Value` VARCHAR(45) NULL,
149 PRIMARY KEY (`User_ID_FK`, `Local_ID_FK`, `Time_Period`, `ID_Prefs_FK`),
150 INDEX `Local_ID_idx` (`Local_ID_FK` ASC),
151 INDEX `User_ID_idx` (`User_ID_FK` ASC),
152 INDEX `ID_Prefs_idx` (`ID_Prefs_FK` ASC),
153 CONSTRAINT `Local_ID`
154 FOREIGN KEY (`Local_ID_FK`)
155 REFERENCES `DB_DataLayer_PhD`.`Local_System_Information` (`idLocal_System`)
156 ON DELETE NO ACTION
157 ON UPDATE NO ACTION,
158 CONSTRAINT `User_ID`
159 FOREIGN KEY (`User_ID_FK`)
160 REFERENCES `DB_DataLayer_PhD`.`User_Information` (`idUser`)
161 ON DELETE NO ACTION
162 ON UPDATE NO ACTION,
163 CONSTRAINT `ID_Prefs`
164 FOREIGN KEY (`ID_Prefs_FK`)
165 REFERENCES `DB_DataLayer_PhD`.`Preferences_IDS` (`idPreference`)
```

```
166 ON DELETE NO ACTION
167 ON UPDATE NO ACTION)
168 ENGINE = InnoDB;
169
170
171 -- -----------------------------------------------------
172 -- Table `DB_DataLayer_PhD`.`History_Pref_Local_System`
173 -- -----------------------------------------------------
174 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`History_Pref_Local_System` ;
175
176 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`History_Pref_Local_System` (
177 `ID_SystemFk` INT NOT NULL,
178 `Time_Period` TIME NOT NULL,
179 `ID_PreferencesFk` INT NOT NULL,
180 `Preference_Value` VARCHAR(45) NULL,
181 PRIMARY KEY (`ID_SystemFk`, `Time_Period`, `ID_PreferencesFk`),
182 INDEX `ID_System_idx` (`ID_SystemFk` ASC),
183 INDEX `ID_Prefs_idx` (`ID_PreferencesFk` ASC),
184 CONSTRAINT `ID_SystemFk`
185 FOREIGN KEY (`ID_SystemFk`)
186 REFERENCES `DB_DataLayer_PhD`.`Local_System_Information` (`idLocal_System`)
187 ON DELETE NO ACTION
188 ON UPDATE NO ACTION,
189 CONSTRAINT `ID_PreferencesFk`
190 FOREIGN KEY (`ID_PreferencesFk`)
191 REFERENCES `DB_DataLayer_PhD`.`Preferences_IDS` (`idPreference`)
192 ON DELETE NO ACTION
193 ON UPDATE NO ACTION)
194 ENGINE = InnoDB;
195
196
197 -- -----------------------------------------------------
198 -- Table `DB_DataLayer_PhD`.`Local_System_Present_Preferences`
199 -- -----------------------------------------------------
200 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Local_System_Present_Preferences` ;
201
202 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Local_System_Present_Preferences` (
203 `LocalSystemID_FK` INT NOT NULL,
204 `Time_Period` TIME NOT NULL,
205 `IdPreference_FK` INT NOT NULL,
206 `Preference_Value` VARCHAR(45) NULL,
207 PRIMARY KEY (`LocalSystemID_FK`, `Time_Period`, `IdPreference_FK`),
208 INDEX `LocalSystemID_idx` (`LocalSystemID_FK` ASC),
209 INDEX `IdPreference_FK_idx` (`IdPreference_FK` ASC),
210 CONSTRAINT `LocalSystemID_FK`
```

```
211 FOREIGN KEY (`LocalSystemID_FK`)
212 REFERENCES `DB_DataLayer_PhD`.`Local_System_Information` (`idLocal_System`)
213 ON DELETE NO ACTION
214 ON UPDATE NO ACTION,
215 CONSTRAINT `IdPreference_FK`
216 FOREIGN KEY (`IdPreference_FK`)
217 REFERENCES `DB_DataLayer_PhD`.`Preferences_IDS` (`idPreference`)
218 ON DELETE NO ACTION
219 ON UPDATE NO ACTION)
220 ENGINE = InnoDB;
221
222
223 -- -----------------------------------------------------
224 -- Table `DB_DataLayer_PhD`.`Sensors_Location`
225 -- -----------------------------------------------------
226 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Sensors_Location` ;
227
228 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Sensors_Location` (
229 `ID_Location` INT NOT NULL,
230 `Desc_Location` VARCHAR(45) NULL,
231 PRIMARY KEY (`ID_Location`))
232 ENGINE = InnoDB;
233
234
235 -- -----------------------------------------------------
236 -- Table `DB_DataLayer_PhD`.`Preferences_Live_Measurement`
237 -- -----------------------------------------------------
238 DROP TABLE IF EXISTS `DB_DataLayer_PhD`.`Preferences_Live_Measurement` ;
239
240 CREATE TABLE IF NOT EXISTS `DB_DataLayer_PhD`.`Preferences_Live_Measurement` (
241 `ID_LocalSystem` INT NOT NULL,
242 `ID_Location_Measured` INT NOT NULL,
243 `Date_Measure` DATE NOT NULL,
244 `Present_Time_Measure` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
245 `Time_Period` TIME NOT NULL,
246 `ID_Prefer_Measured` INT NOT NULL,
247 `Pref_Value_Measured` VARCHAR(45) NULL,
248 PRIMARY KEY (`ID_LocalSystem`, `ID_Location_Measured`, `Time_Period`, `
    ↪ ID_Prefer_Measured`, `Present_Time_Measure`),
249 INDEX `Id_Loc_Measured_Fk_idx` (`ID_Location_Measured` ASC),
250 INDEX `Id_Preference_Fk_idx` (`ID_Prefer_Measured` ASC),
251 CONSTRAINT `ID_Loc_SystemFk`
252 FOREIGN KEY (`ID_LocalSystem`)
253 REFERENCES `DB_DataLayer_PhD`.`Local_System_Information` (`idLocal_System`)
254 ON DELETE NO ACTION
```

```
255 ON UPDATE NO ACTION,
256 CONSTRAINT `Id_Loc_Measured_Fk`
257 FOREIGN KEY (`ID_Location_Measured`)
258 REFERENCES `DB_DataLayer_PhD`.`Sensors_Location` (`ID_Location`)
259 ON DELETE NO ACTION
260 ON UPDATE NO ACTION,
261 CONSTRAINT `Id_Preference_Fk`
262 FOREIGN KEY (`ID_Prefer_Measured`)
263 REFERENCES `DB_DataLayer_PhD`.`Preferences_IDS` (`idPreference`)
264 ON DELETE NO ACTION
265 ON UPDATE NO ACTION)
266 ENGINE = InnoDB;
267
268 USE `DB_DataLayer_PhD`;
269
270 DELIMITER $$
271
272 USE `DB_DataLayer_PhD`$$
273 DROP TRIGGER IF EXISTS `DB_DataLayer_PhD`.`Preferences_Live_Measurement_BEFORE_INSERT`
    ↪ $$
274 USE `DB_DataLayer_PhD`$$
275 CREATE DEFINER = CURRENT_USER TRIGGER `DB_DataLayer_PhD`.`
    ↪ Preferences_Live_Measurement_BEFORE_INSERT` BEFORE INSERT ON `
    ↪ Preferences_Live_Measurement` FOR EACH ROW
276 BEGIN
277 SET NEW.Date_Measure = NOW();
278 END$$
279
280
281 DELIMITER ;
282
283 SET SQL_MODE=@OLD_SQL_MODE;
284 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
285 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

A.2 Insert_All_DB_Data.sql

```
1
2  INSERT INTO `DB_DataLayer_PhD`.Local_System_Information
3  (idLocal_System,Desc_LocalSystem,GPS_Latitude,GPS_Longitude)
4  VALUES
5  (1,"raspA","20","-10"),(2,"raspB","40","-20"),(3,"Work System","50","-30");
6
7
8  INSERT INTO `DB_DataLayer_PhD`.Preferences_IDS
9  (idPreference, Description_Preference, Unities_Preference, Data_Type_Preference)
10 VALUES
11 (1,"Temperature","C","Integer"),
12 (2,"Humidity","%","Integer"),
13 (3,"Musical Genre","String","String"),
14 (4,"Musical Playlist","String","String"),
15 (5,"Radio Station","String","String"),
16 (6,"Tv Station","String","String"),
17 (7,"Media Genre","String","String")
18 ON DUPLICATE KEY UPDATE
19 Description_Preference = VALUES(Description_Preference),
20 Unities_Preference = VALUES(Unities_Preference),
21 Data_Type_Preference = VALUES(Data_Type_Preference);
22
23 INSERT INTO `DB_DataLayer_PhD`.Preference_Rank(idPreference_Rank,Pref_Rank_Description
       ↪ )
24 VALUES
25 (1,"Rank 1"),
26 (2,"Rank 2"),
27 (3,"Rank 3");
28
29 INSERT INTO `DB_DataLayer_PhD`.Sensors_Location
30 (`ID_Location`,`Desc_Location`)
31 VALUES
32 (1,"Room Temperature"),
33 (2,"Room Humidity"),
34 (3,"Bathroom Temperature"),
35 (4,"Bathroom Humidity");
```

B

User Behaviour Simulation - Code

B.1 randomsClass.java

```
1  package Gen_Users;
2
3  public class randomsClass {
4
5  private String idUser;
6  private String day;
7  private String idLocalSystem;
8  private String randomTimeDelayEnterWorkHours;
9  private String randomTimeDelayEnterHomeHours;
10 private String randomTimeDelayExitHomeHours;
11 private String randomTimeDelayLazerHours;
12
13 public randomsClass() {
14 }
15
16 public randomsClass(String idUser, String idLocalSystem, String day, String
       ↪ randomTimeDelayEnterWorkHours, String randomTimeDelayEnterHomeHours, String
       ↪ randomTimeDelayExitHomeHours, String randomTimeDelayLazerHours) {
17 super();
18 this.idUser = idUser;
19 this.idLocalSystem = idLocalSystem;
20 this.day = day;
21 this.randomTimeDelayEnterWorkHours = randomTimeDelayEnterWorkHours;
22 this.randomTimeDelayEnterHomeHours = randomTimeDelayEnterHomeHours;
23 this.randomTimeDelayExitHomeHours = randomTimeDelayExitHomeHours;
24 this.randomTimeDelayLazerHours = randomTimeDelayLazerHours;
25 }
26
27 public String getidUser() {
28 return idUser;
```

```
29 }
30
31 public void setidUser(String idUser) {
32 this.idUser = idUser;
33 }
34
35 public String getidLocalSystem() {
36 return idLocalSystem;
37 }
38
39 public void setidLocalSystem(String idLocalSystem) {
40 this.idLocalSystem = idLocalSystem;
41 }
42
43 public String getday() {
44 return day;
45 }
46
47 public void setday(String day) {
48 this.day = day; }
49
50 public String getrandomTimeDelayEnterWorkHours() {
51 return randomTimeDelayEnterWorkHours; }
52
53 public void setrandomTimeDelayEnterWorkHours(String randomTimeDelayEnterWorkHours) {
54 this.randomTimeDelayEnterWorkHours = randomTimeDelayEnterWorkHours; }
55
56 public String getrandomTimeDelayEnterHomeHours() {
57 return randomTimeDelayEnterHomeHours; }
58
59 public void setrandomTimeDelayEnterHomeHours(String randomTimeDelayEnterHomeHours) {
60 this.randomTimeDelayEnterHomeHours = randomTimeDelayEnterHomeHours; }
61
62 public String getrandomTimeDelayExitHomeHours() {
63 return randomTimeDelayExitHomeHours; }
64
65 public void setrandomTimeDelayExitHomeHours(String randomTimeDelayExitHomeHours) {
66 this.randomTimeDelayExitHomeHours = randomTimeDelayExitHomeHours; }
67
68 public String getrandomTimeDelayLazerHours() {
69 return randomTimeDelayLazerHours; }
70
71 public void setrandomTimeDelayLazerHours(String randomTimeDelayLazerHours) {
72 this.randomTimeDelayLazerHours = randomTimeDelayLazerHours;
73 } }
```

B.2 Generate_User_Simulation.java

```
1
2  package Gen_Users;
3
4  import java.sql.Timestamp;
5  import java.io.File;
6  import java.io.FileInputStream;
7  import java.io.FileNotFoundException;
8  import java.io.FileOutputStream;
9  import java.io.IOException;
10 import java.sql.Time;
11 import java.text.ParseException;
12 import java.text.SimpleDateFormat;
13 import java.util.ArrayList;
14 import java.util.Calendar;
15 import java.util.Date;
16 import java.util.Dictionary;
17 import java.util.HashMap;
18 import java.util.Hashtable;
19 import java.util.List;
20 import java.util.Map;
21 import java.util.Random;
22 import java.util.Set;
23
24 import javax.ws.rs.core.MediaType;
25
26 import com.sun.jersey.core.impl.provider.entity.StringProvider;
27 import org.apache.poi.ss.usermodel.Cell;
28 import org.apache.poi.ss.usermodel.Row;
29 import org.apache.poi.xssf.usermodel.XSSFSheet;
30 import org.apache.poi.xssf.usermodel.XSSFWorkbook;
31
32 import com.sun.jersey.api.client.Client;
33 import com.sun.jersey.api.client.ClientResponse;
34 import com.sun.jersey.api.client.WebResource;
35 import com.sun.jersey.api.client.config.ClientConfig;
36 import com.sun.jersey.api.client.config.DefaultClientConfig;
37 import com.sun.jersey.api.client.filter.HTTPBasicAuthFilter;
38
39 import java.io.FileWriter;
40
41
42 public class Generate_User_Simulation {
43
```

```
44 static final String REST_URI_LOC_SYSTEM = "http://192.168.217.160:8080/WS/Rest/
       ↪ LocalSystemWS";
45
46 static final String REST_URI_USER = "http://192.168.217.160:8080/WS/Rest/userWS";
47
48 static final String INSERT_LOC_SYSTEM_PATH = "insertLocalSystem";
49
50 static final String INSERT_HIST_PREF_USER_PATH = "
       ↪ insertHistoryPrefUserWithIdUserAndTime";
51
52 //private static final int idLocal_System = 156;
53 //private static final String Desc_LocalSystem = "Sistema Local Testes";
54 private static final String GPS_Latitude = "100";
55 private static final String GPS_Longitude = "-50";
56
57 int idLocalSystem = 5000;
58
59 List<Integer> idsUsersAtSystem = new ArrayList<Integer>();
60
61 List<Integer> idsPublicLocalSystems = new ArrayList<Integer>();
62
63 int numberOfUsersAtSystem = 50;
64 int firstIdUserAtSystem = 201;
65
66 int numMinLocSystemsByUser = 1;
67 int numMaxLocSystemsByUser = 4; //4+1 = 5 Local systems Max
68
69 List<Time> arrayEnterWorkHours = new ArrayList<Time>();
70 List<Time> arrayEnterHomeHours = new ArrayList<Time>();
71 List<Time> arrayExitHomeHours = new ArrayList<Time>();
72 List<Time> arrayLazerHours = new ArrayList<Time>();
73
74
75 private String[] arrayEnterWorkHours2 = {"08:00", "09:00", "16:00", "00:00"};
76
77 private String[] arrayEnterHomeHours2 = {"17:10", "18:10", "00:10", "08:10"};
78
79 private String[] arrayExitHomeHours2 = {"07:50", "08:50", "15:50", "23:50"};
80
81 private String[] arrayLazerHours2 = {"20:00", "21:00", "10:00", "15:00"};
82
83 private int[] arrayUsersType = {0, 1, 2, 3};
84
85 int idUserType = 0;
86
```

```
87  int idMasterUser;
88  int idUser;
89
90  int TypeOfSystem;
91
92  Dictionary<Integer, Integer> dictUsersType = new Hashtable<Integer, Integer>();
93  Dictionary<Integer, Integer> dictUsersNumberOfSystems = new Hashtable<Integer, Integer
        ↪ >();
94
95  int delayEnterWorkMin = -10;
96  int delayEnterWorkMax = 10;
97
98  int delayEnterHomeMin = -30;
99  int delayEnterHomeMax = 30;
100
101 int delayExitHomeMin = -40;
102 int delayExitHomeMax = 40;
103
104 int delayLazerHoursMin = -90;
105 int delayLazerHoursMax = 90;
106
107 private static Map<Integer, Object[]> dataExcelA = new HashMap<Integer, Object[]>();
108 private static Map<Integer, Object[]> dataExcelB = new HashMap<Integer, Object[]>();
109
110 private static List<randomsClass> dataExcelC = new ArrayList<randomsClass>();
111
112
113 public static void main(String[] args) throws Exception {
114 dataExcelB.put(1, new Object[]{"idMasterUser", "TypeOfSystem", "randomUsersAtSameHome"
        ↪ , "randomUsersAtSameWork"});
115
116 dataExcelC.add(new randomsClass("idUser", "idLocalSystem", "day", "
        ↪ randomTimeDelayEnterWorkHours", "randomTimeDelayEnterHomeHours", "
        ↪ randomTimeDelayExitHomeHours", "randomTimeDelayLazerHours"));
117
118
119 long tStart = 0;
120 long tEnd = 0;
121
122
123 tStart = System.currentTimeMillis();
124
125 Generate_User_Simulation genenerateUserSimul = new Generate_User_Simulation();
126
127
```

```
128
129 genenerateUserSimul.generateSimulatedUsers();
130
131 tEnd = System.currentTimeMillis();
132 long tDelta = tEnd - tStart;
133 double elapsedMinutes = (tDelta / 1000.0) / 60;
134 System.out.printf("Method initializeUtilization_Preferences Done In %f Minutes! \n",
        ↪ elapsedMinutes);
135
136
137 genenerateUserSimul.saveInfoToExcelA(dataExcelA, 0);
138 genenerateUserSimul.saveInfoToExcelA(dataExcelB, 1);
139
140
141 String fileName = "/home/poliveira/Excel_Simulation/randomNumbers_UserSimulation.csv";
142
143 genenerateUserSimul.writeCsvFile(fileName, dataExcelC);
144
145 }
146
147 private void generateSimulatedUsers() throws ParseException
148
149 {
150 dataExcelA.put(1, new Object[]{"idUser", "randomNumberOfLocalSystemsByUser", "
        ↪ randomIndexUserType"});
151
152 Generate_User_Simulation genenerateUserSimul = new Generate_User_Simulation();
153
154 for (idUser = firstIdUserAtSystem; idUser <= (firstIdUserAtSystem+
        ↪ numberOfUsersAtSystem); ) {
155
156 Random random0 = new Random();
157 int randomNumberOfLocalSystemsByUser = random0.nextInt(numMaxLocSystemsByUser -
        ↪ numMinLocSystemsByUser + 1) + numMinLocSystemsByUser;
158
159 int randomIndex = random0.nextInt(arrayUsersType.length);
160
161 idUserType = arrayUsersType[randomIndex];
162
163 dictUsersType.put(idUser, idUserType);
164
165 dictUsersNumberOfSystems.put(idUser, randomNumberOfLocalSystemsByUser);
166
167
168 System.out.println("idUser: " + idUser);
```

```
169
170 System.out.println("idUserType: " + idUserType);
171
172 System.out.println("randomNumberOfLocalSystemsByUser: " +
    ↪ randomNumberOfLocalSystemsByUser);
173
174
175 dataExcelA.put((idUser - 195), new Object[]{String.valueOf(idUser), String.valueOf(
    ↪ randomNumberOfLocalSystemsByUser), String.valueOf(randomIndex)});
176
177
178 String DescLocalSystem;
179
180 idMasterUser = idUser;
181
182 switch (randomNumberOfLocalSystemsByUser) {
183 case 1:
184
185 TypeOfSystem = 1;
186
187 System.out.println("Home");
188 idLocalSystem++;
189
190
191 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
192
193 DescLocalSystem = ("Home: " + idMasterUser + " +" + idsUsersAtSystem);
194
195 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
196
197 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
    ↪  TypeOfSystem);
198
199 break;
200
201 case 2:
202
203 System.out.println("Home + Work");
204
205 for (int l = 1; l <= 2; l++) {
206 idLocalSystem++;
207
208 if (l == 1) {
209 TypeOfSystem = 1;
210
```

```
211 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
212
213 DescLocalSystem = ("Home: " + idMasterUser + " +" + idsUsersAtSystem);
214
215 System.out.println("idUser " + idUser + " idLocalSystem " + idLocalSystem);
216 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
217
218 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
        ↪ TypeOfSystem);
219
220 }
221 if (l == 2) {
222 TypeOfSystem = 2;
223
224 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
225
226
227 DescLocalSystem = ("Work: " + idMasterUser + " +" + idsUsersAtSystem);
228
229 System.out.println("idUser " + idUser + " idLocalSystem " + idLocalSystem);
230
231 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
232
233 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
        ↪ TypeOfSystem);
234 }
235 }
236
237 break;
238 case 3:
239
240 System.out.println("Home + Work + 1 Public System");
241
242 for (int l = 1; l <= 3; l++) {
243 idLocalSystem++;
244
245 if (l == 1) {
246 TypeOfSystem = 1;
247
248 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
249
250 DescLocalSystem = ("Home: " + idMasterUser + " +" + idsUsersAtSystem);
251 System.out.println("idUser " + idUser + " idLocalSystem " + idLocalSystem);
252
253 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
```

```
254
255 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
        ↪ TypeOfSystem);
256 }
257 if (l == 2) {
258 TypeOfSystem = 2;
259
260 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
261
262 DescLocalSystem = ("Work: " + idMasterUser + " +" + idsUsersAtSystem);
263 System.out.println("idUser " + idUser + " idLocalSystem " + idLocalSystem);
264
265 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
266 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
        ↪ TypeOfSystem);
267 }
268 if (l == 3) {
269 TypeOfSystem = 3;
270
271 idsPublicLocalSystems.add(idLocalSystem);
272
273 DescLocalSystem = ("Public System");
274
275 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
276
277 for (int i = 0; i < idsPublicLocalSystems.size(); i++) {
278 System.out.println("idsPublicLocalSystems " + idsPublicLocalSystems.get(i));
279
280 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem,
        ↪ idsPublicLocalSystems.get(i), TypeOfSystem);
281
282 }
283 }
284 }
285 break;
286 case 4:
287
288 System.out.println("Home + Work + 2 Public Systems");
289
290 for (int l = 1; l <= 4; l++) {
291 idLocalSystem++;
292 if (l == 1) {
293 TypeOfSystem = 1;
294
295 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
```

```
296 DescLocalSystem = ("Home: " + idMasterUser + " +" + idsUsersAtSystem);
297 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
298
299 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
        ↪ TypeOfSystem);
300 }
301 if (l == 2) {
302 TypeOfSystem = 2;
303
304 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
305 DescLocalSystem = ("Work: " + idMasterUser + " +" + idsUsersAtSystem);
306 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
307
308 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
        ↪ TypeOfSystem);
309 }
310 if (l == 3) {
311 TypeOfSystem = 3;
312
313 DescLocalSystem = ("Public System");
314 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
315
316 idsPublicLocalSystems.add(idLocalSystem);
317 for (int i = 0; i < idsPublicLocalSystems.size(); i++) {
318 System.out.println("idsPublicLocalSystems " + idsPublicLocalSystems.get(i));
319
320 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem,
        ↪ idsPublicLocalSystems.get(i), TypeOfSystem);
321 }
322 }
323 if (l == 4) {
324 TypeOfSystem = 3;
325
326 DescLocalSystem = ("Public System");
327 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
328
329 idsPublicLocalSystems.add(idLocalSystem);
330 for (int i = 0; i < idsPublicLocalSystems.size(); i++) {
331 System.out.println("idsPublicLocalSystems " + idsPublicLocalSystems.get(i));
332
333 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem,
        ↪ idsPublicLocalSystems.get(i), TypeOfSystem);
334 }
335 }
336 }
```

```
337
338 break;
339 case 5:
340
341 System.out.println("Home + Work + 3 Public Systems");
342
343 for (int l = 1; l <= 5; l++) {
344 idLocalSystem++;
345
346 if (l == 1) {
347 TypeOfSystem = 1;
348 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
349 DescLocalSystem = ("Home: " + idMasterUser + " +" + idsUsersAtSystem);
350 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
351
352 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
    ↪ TypeOfSystem);
353 }
354 if (l == 2) {
355 TypeOfSystem = 2;
356
357 addMoreUsersToSystem(idMasterUser, TypeOfSystem);
358 DescLocalSystem = ("Work: " + idMasterUser + " +" + idsUsersAtSystem);
359 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
360
361 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem, idLocalSystem,
    ↪ TypeOfSystem);
362 }
363 if (l == 3) {
364 TypeOfSystem = 3;
365
366 DescLocalSystem = ("Public System");
367 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
368
369 idsPublicLocalSystems.add(idLocalSystem);
370 for (int i = 0; i < idsPublicLocalSystems.size(); i++) {
371 System.out.println("idsPublicLocalSystems " + idsPublicLocalSystems.get(i));
372
373 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem,
    ↪ idsPublicLocalSystems.get(i), TypeOfSystem);
374 }
375 }
376 if (l == 4) {
377 TypeOfSystem = 3;
378
```

```
379 DescLocalSystem = ("Public System");
380 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
381
382 idsPublicLocalSystems.add(idLocalSystem);
383 for (int i = 0; i < idsPublicLocalSystems.size(); i++) {
384 System.out.println("idsPublicLocalSystems " + idsPublicLocalSystems.get(i));
385
386 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem,
        ↪ idsPublicLocalSystems.get(i), TypeOfSystem);
387 }
388 }
389 if (l == 5) {
390 TypeOfSystem = 3;
391
392 DescLocalSystem = ("Public System");
393 genenerateUserSimul.AddLocalSystemWS(idLocalSystem, DescLocalSystem);
394
395 idsPublicLocalSystems.add(idLocalSystem);
396 for (int i = 0; i < idsPublicLocalSystems.size(); i++) {
397 System.out.println("idsPublicLocalSystems " + idsPublicLocalSystems.get(i));
398
399 genenerateUserSimul.insertHistoryRecord(idMasterUser, idsUsersAtSystem,
        ↪ idsPublicLocalSystems.get(i), TypeOfSystem);
400 }
401 }
402 }
403 break;
404 default:
405 break;
406 }
407 ++idUser;
408 }
409
410 }
411
412 private void AddLocalSystemWS(int idLocalSystem, String DescLocalSystem) {
413 try {
414
415 ClientConfig clientConfig = new DefaultClientConfig();
416
417 clientConfig.getClasses().add(StringProvider.class);
418 Client client = Client.create(clientConfig);
419
420 client.addFilter(new HTTPBasicAuthFilter("pfo", "Pfo214"));
421
```

```
422 WebResource service = client.resource(REST_URI_LOC_SYSTEM);
423
424 WebResource insertLocalSystemService = service.path(INSERT_LOC_SYSTEM_PATH).path(
        ↪ idLocalSystem + "/" + DescLocalSystem + "/" + GPS_Latitude + "/" +
        ↪ GPS_Longitude);
425
426 ClientResponse response = insertLocalSystemService.accept(MediaType.APPLICATION_JSON).
        ↪ post(ClientResponse.class);
427
428 if (response.getStatus() != 200) {
429 throw new RuntimeException("Failed : HTTP error code : "
430 + response.getStatus());
431 }
432
433 System.out.println("Output from Server to insertLocalSystemService -> %s \n" +
        ↪ idLocalSystem);
434 String output = response.getEntity(String.class);
435 System.out.println(output);
436 System.out.println("Local System Added -> " + idLocalSystem + "\n");
437
438 } catch (Exception e) {
439
440 e.printStackTrace();
441
442 }
443 }
444
445
446 private void insertHistoryRecord(int idMasterUser, List<Integer> idsUsersAtSystem, int
        ↪  idLocalSystem, int TypeOfSystem) throws ParseException {
447 arrayEnterWorkHours.add(Time.valueOf("08:00:00"));
448 arrayEnterWorkHours.add(Time.valueOf("09:00:00"));
449 arrayEnterWorkHours.add(Time.valueOf("16:00:00"));
450 arrayEnterWorkHours.add(Time.valueOf("00:00:00"));
451
452 arrayEnterHomeHours.add(Time.valueOf("17:10:00"));
453 arrayEnterHomeHours.add(Time.valueOf("18:10:00"));
454 arrayEnterHomeHours.add(Time.valueOf("00:10:00"));
455 arrayEnterHomeHours.add(Time.valueOf("08:10:00"));
456
457 arrayExitHomeHours.add(Time.valueOf("07:50:00"));
458 arrayExitHomeHours.add(Time.valueOf("08:50:00"));
459 arrayExitHomeHours.add(Time.valueOf("15:50:00"));
460 arrayExitHomeHours.add(Time.valueOf("23:50:00"));
461
```

```
462 arrayLazerHours.add(Time.valueOf("20:00:00"));
463 arrayLazerHours.add(Time.valueOf("21:00:00"));
464 arrayLazerHours.add(Time.valueOf("10:00:00"));
465 arrayLazerHours.add(Time.valueOf("15:00:00"));
466
467
468 Generate_User_Simulation genenerateUserSimul = new Generate_User_Simulation();
469
470 System.out.println("idusertye -- " + idUserType);
471
472 System.out.println(arrayEnterWorkHours);
473
474 Time enterWork = arrayEnterWorkHours.get(idUserType);
475
476 Time enterHome = arrayEnterHomeHours.get(idUserType);
477 Time exitHome = arrayExitHomeHours.get(idUserType);
478 Time lazerHours = arrayLazerHours.get(idUserType);
479
480 Calendar calToday = Calendar.getInstance();
481
482 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
483
484 String strDateToday = sdf.format(calToday.getTime());
485
486 int numberOfHistoryDays = 60;
487
488
489
490 idsUsersAtSystem.add(idMasterUser);
491
492 for (int u = 0; u < idsUsersAtSystem.size(); u++) {
493
494 idUser = idsUsersAtSystem.get(u);
495
496 Date dateTodayToInsertPlusOneDay = sdf.parse(strDateToday);
497
498
499 for (int day = 1; day <= numberOfHistoryDays; day++) {
500 dateTodayToInsertPlusOneDay = new Date(dateTodayToInsertPlusOneDay.getTime() + (1000 *
    ↪  60 * 60 * 24));
501
502 Random random1 = new Random();
503 Random random2 = new Random();
504 Random random3 = new Random();
505 Random random4 = new Random();
```

```
506
507 int randomTimeDelayEnterWorkHours = random1.nextInt(delayEnterWorkMax -
        ↪ delayEnterWorkMin + 1) + delayEnterWorkMin;
508
509 int randomTimeDelayEnterHomeHours = random2.nextInt(delayEnterHomeMax -
        ↪ delayEnterHomeMin + 1) + delayEnterHomeMin;
510
511 int randomTimeDelayExitHomeHours = random3.nextInt(delayExitHomeMax - delayExitHomeMin
        ↪  + 1) + delayExitHomeMin;
512
513 int randomTimeDelayLazerHours = random4.nextInt(delayLazerHoursMax -
        ↪ delayLazerHoursMin + 1) + delayLazerHoursMin;
514
515
516 dataExcelC.add(new randomsClass(String.valueOf(idUser), String.valueOf(idLocalSystem),
        ↪  String.valueOf(day), String.valueOf(randomTimeDelayEnterWorkHours), String.
        ↪ valueOf(randomTimeDelayEnterHomeHours), String.valueOf(
        ↪ randomTimeDelayExitHomeHours), String.valueOf(randomTimeDelayLazerHours)));
517
518
519 Calendar cal = Calendar.getInstance();
520
521 cal.setTime(enterWork);
522 cal.add(Calendar.MINUTE, randomTimeDelayEnterWorkHours);
523 Date enterWorkToInsert = cal.getTime();
524
525
526
527 cal.setTime(enterHome);
528 cal.add(Calendar.MINUTE, randomTimeDelayEnterHomeHours);
529 Date enterHomeToInsert = cal.getTime();
530
531 cal.setTime(exitHome);
532 cal.add(Calendar.MINUTE, randomTimeDelayExitHomeHours);
533 Date exitHomeToInsert = cal.getTime();
534
535 cal.setTime(lazerHours);
536 cal.add(Calendar.MINUTE, randomTimeDelayLazerHours);
537 Date lazerHoursToInsert = cal.getTime();
538
539
540 Timestamp dateEnterWorkToInsert = combineDateTime(dateTodayToInsertPlusOneDay,
        ↪ enterWorkToInsert);
541 Timestamp dateEnterHomeToInsert = combineDateTime(dateTodayToInsertPlusOneDay,
        ↪ enterHomeToInsert);
```

```
542 Timestamp dateExitHomeToInsert = combineDateTime(dateTodayToInsertPlusOneDay,
        ↪ exitHomeToInsert);
543 Timestamp dateEnterLazerToInsert = combineDateTime(dateTodayToInsertPlusOneDay,
        ↪ lazerHoursToInsert);
544
545 switch (TypeOfSystem) {
546 case 1:
547
548 genenerateUserSimul.insertHistoryPrefUserWS(idUser, idLocalSystem,
        ↪ dateEnterHomeToInsert);
549 System.out.println("insertHistoryPrefUserWS_Home: " + idUser + " - " + idLocalSystem +
        ↪  " - " + dateEnterHomeToInsert);
550
551 break;
552 case 2:
553
554 genenerateUserSimul.insertHistoryPrefUserWS(idUser, idLocalSystem,
        ↪ dateEnterWorkToInsert);
555 System.out.println("insertHistoryPrefUserWS_Work: " + idUser + " - " + idLocalSystem +
        ↪  " - " + dateEnterWorkToInsert);
556
557 break;
558 case 3:
559
560 genenerateUserSimul.insertHistoryPrefUserWS(idUser, idLocalSystem,
        ↪ dateEnterLazerToInsert);
561 System.out.println("insertHistoryPrefUserWS_PublicSystem: " + idUser + " - " +
        ↪ idLocalSystem + " - " + dateEnterLazerToInsert);
562
563 break;
564 default:
565 break;
566 }
567
568 }
569 }
570 idsUsersAtSystem.clear();
571
572 }
573
574 private void insertHistoryPrefUserWS(int idUser, int idLocalSystem, Timestamp
        ↪ DateTimeToInsert) {
575 try {
576 ClientConfig clientConfig = new DefaultClientConfig();
577
```

```
578 clientConfig.getClasses().add(StringProvider.class);
579
580 Client client = Client.create(clientConfig);
581
582 client.addFilter(new HTTPBasicAuthFilter("pfo", "Pfo214"));
583
584 WebResource service = client.resource(REST_URI_USER);
585
586 WebResource insertHistoryPrefUserWS = service.path(INSERT_HIST_PREF_USER_PATH).path(
        ↪ idUser + "/" + idLocalSystem + "/" + DateTimeToInsert);
587
588 System.out.println("insertHistoryPrefUserWS: " + insertHistoryPrefUserWS);
589
590 ClientResponse response = insertHistoryPrefUserWS.accept(MediaType.APPLICATION_JSON).
        ↪ post(ClientResponse.class);
591
592 if (response.getStatus() != 200) {
593 throw new RuntimeException("Failed : HTTP error code : "
594 + response.getStatus());
595 }
596
597 System.out.println("Output from Server to insertHistoryPrefUserWS.... \n");
598 String output = response.getEntity(String.class);
599 System.out.println(output);
600 System.out.println("..... \n");
601
602 } catch (Exception e) {
603
604 e.printStackTrace();
605
606 }
607 }
608
609 private Timestamp combineDateTime(Date date, Date time) {
610 Calendar calendarA = Calendar.getInstance();
611 calendarA.setTime(date);
612 Calendar calendarB = Calendar.getInstance();
613 calendarB.setTime(time);
614
615 calendarA.set(Calendar.HOUR_OF_DAY, calendarB.get(Calendar.HOUR_OF_DAY));
616 calendarA.set(Calendar.MINUTE, calendarB.get(Calendar.MINUTE));
617 calendarA.set(Calendar.SECOND, calendarB.get(Calendar.SECOND));
618 calendarA.set(Calendar.MILLISECOND, calendarB.get(Calendar.MILLISECOND));
619
620 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
```

```
621
622 Date result = calendarA.getTime();
623
624 String strDateResult = sdf.format(result);
625
626 Timestamp TimestampDateResult = Timestamp.valueOf(strDateResult);
627
628 return TimestampDateResult;
629 }
630
631 private void addMoreUsersToSystem(int idMasterUser, int TypeOfSystem) {
632
633 int NrMinUsersAtHome = 0;
634 int NrMaxUsersAtHome = 2;
635
636 int NrMinUsersAtWork = 0;
637 int NrMaxUsersAtWork = 3;
638
639 Random random0 = new Random();
640 Random random1 = new Random();
641
642 int randomUsersAtSameHome = random0.nextInt((NrMaxUsersAtHome) + 1) + NrMinUsersAtHome
    ↪ ;
643 int randomUsersAtSameWork = random1.nextInt((NrMaxUsersAtWork) + 1) + NrMinUsersAtWork
    ↪ ;
644
645 dataExcelB.put((idUser - 195), new Object[]{String.valueOf(idMasterUser), String.
    ↪ valueOf(TypeOfSystem), String.valueOf(randomUsersAtSameHome), String.valueOf(
    ↪ randomUsersAtSameWork)});
646
647 switch (TypeOfSystem) {
648 case 1:
649
650 for (int plusUsers = 0; plusUsers <= randomUsersAtSameHome; plusUsers++) {
651 idUser++;
652 idsUsersAtSystem.add(idUser);
653 }
654 break;
655
656 case 2:
657 for (int plusUsers = 0; plusUsers <= randomUsersAtSameWork; plusUsers++) {
658 idUser++;
659 idsUsersAtSystem.add(idUser);
660 }
661 break;
```

```
662
663 default:
664 break;
665 }
666 }
667
668 private void saveInfoToExcelA(Map<Integer, Object[]> dataExcelInfo, int SheetNumber)
669
670 {
671
672 System.out.println("\n Writing on XLSX file Started ...");
673
674 try {
675
676 File myFile = new File("/home/poliveira/Excel_Simulation/randomNumbers_UserSimulation.
    ↪ xlsx");
677
678 FileInputStream fis = new FileInputStream(myFile);
679
680 // Finds the workbook instance for XLSX file
681 XSSFWorkbook myWorkBook = new XSSFWorkbook(fis);
682
683 // Return first sheet from the XLSX workbook
684
685 myWorkBook.removePrintArea(SheetNumber);
686
687
688 XSSFSheet mySheet = myWorkBook.getSheetAt(SheetNumber);
689
690 //Iterate over data and write to sheet
691 Set<Integer> keyset = dataExcelInfo.keySet();
692 int rownum = 0;
693 for (Integer key : keyset) {
694 Row row = mySheet.createRow(rownum++);
695 Object[] objArr = dataExcelInfo.get(key);
696 int cellnum = 0;
697 for (Object obj : objArr) {
698 Cell cell = row.createCell(cellnum++);
699 if (obj instanceof String)
700 cell.setCellValue((String) obj);
701 else if (obj instanceof Integer)
702 cell.setCellValue((Integer) obj);
703 }
704 }
705
```

```
706 // open an OutputStream to save written data into XLSX file
707 FileOutputStream os = new FileOutputStream(myFile);
708 myWorkBook.write(os);
709 System.out.println("Writing on XLSX file Finished ...\n");
710
711 // Close workbook, OutputStream and Excel file to prevent leak
712 os.close();
713 myWorkBook.close();
714 fis.close();
715
716 } catch (FileNotFoundException fe) {
717 fe.printStackTrace();
718 } catch (IOException ie) {
719 ie.printStackTrace();
720 }
721 }
722
723 private void saveInfoToExcelB(List<randomsClass> dataExcelInfo, int SheetNumber)
724 {
725
726 System.out.println("\n Writing on XLSX file Started ...");
727
728 try {
729 File myFile = new File("/home/poliveira/Excel_Simulation/randomNumbers_UserSimulation.
    ↪ xlsx");
730
731
732 FileInputStream fis = new FileInputStream(myFile);
733
734 // Finds the workbook instance for XLSX file
735 XSSFWorkbook myWorkBook = new XSSFWorkbook(fis);
736
737 // Return first sheet from the XLSX workbook
738
739 myWorkBook.removePrintArea(SheetNumber);
740
741 //myWorkBook.createSheet
742
743 XSSFSheet mySheet = myWorkBook.getSheetAt(SheetNumber);
744
745 int rowIndex = 0;
746
747 for (randomsClass randClass : dataExcelInfo) {
748 Row row = mySheet.createRow(rowIndex++);
749 int cellIndex = 0;
```

```
750
751 row.createCell(cellIndex++).setCellValue(randClass.getidUser());
752 row.createCell(cellIndex++).setCellValue(randClass.getidLocalSystem());
753 row.createCell(cellIndex++).setCellValue(randClass.getday());
754 row.createCell(cellIndex++).setCellValue(randClass.getrandomTimeDelayEnterWorkHours())
        ↪ ;
755 row.createCell(cellIndex++).setCellValue(randClass.getrandomTimeDelayEnterHomeHours())
        ↪ ;
756 row.createCell(cellIndex++).setCellValue(randClass.getrandomTimeDelayExitHomeHours());
757 row.createCell(cellIndex++).setCellValue(randClass.getrandomTimeDelayLazerHours());
758 }
759
760 // open an OutputStream to save written data into XLSX file
761 FileOutputStream os = new FileOutputStream(myFile);
762 myWorkBook.write(os);
763 System.out.println("Writing on XLSX file Finished ...\n");
764
765 // Close workbook, OutputStream and Excel file to prevent leak
766 os.close();
767 myWorkBook.close();
768 fis.close();
769
770 } catch (FileNotFoundException fe) {
771 fe.printStackTrace();
772 } catch (IOException ie) {
773 ie.printStackTrace();
774 }
775 }
776
777 private void writeCsvFile(String fileName, List<randomsClass> dataExcelInfo) {
778
779 //CSV file header
780
781 String FILE_HEADER = "id,firstName,lastName,gender,age";
782
783 String DELIMITER = ";";
784
785 String NEW_LINE_SEPARATOR = "\n";
786
787 FileWriter fileWriter = null;
788
789 try {
790 fileWriter = new FileWriter(fileName);
791
792 //Write the CSV file header
```

```
793 fileWriter.append(FILE_HEADER.toString());
794
795 //Add a new line separator after the header
796 fileWriter.append(NEW_LINE_SEPARATOR);
797
798 //Write a new student object list to the CSV file
799 for (randomsClass randClass : dataExcelInfo) {
800 fileWriter.append(randClass.getidUser());
801 fileWriter.append(DELIMITER);
802 fileWriter.append(randClass.getidLocalSystem());
803 fileWriter.append(DELIMITER);
804 fileWriter.append(randClass.getday());
805 fileWriter.append(DELIMITER);
806 fileWriter.append(randClass.getrandomTimeDelayEnterWorkHours());
807 fileWriter.append(DELIMITER);
808 fileWriter.append(randClass.getrandomTimeDelayEnterHomeHours());
809 fileWriter.append(DELIMITER);
810 fileWriter.append(randClass.getrandomTimeDelayExitHomeHours());
811 fileWriter.append(DELIMITER);
812 fileWriter.append(randClass.getrandomTimeDelayLazerHours());
813 fileWriter.append(NEW_LINE_SEPARATOR);
814
815 }
816 System.out.println("CSV file was created successfully !!!");
817
818 } catch (Exception e) {
819 System.out.println("Error in CsvFileWriter !!!");
820 e.printStackTrace();
821 } finally {
822
823 try {
824 fileWriter.flush();
825 fileWriter.close();
826 } catch (IOException e) {
827 System.out.println("Error while flushing/closing fileWriter !!!");
828 e.printStackTrace();
829 }
830 }
831 }
832 }
```

C

User Behaviour Simulation - Validation

C.1 Function_Test_ChiQ.R

```
1
2  rm(list=ls(all=TRUE))
3  print("########################")
4  # leitura dos dados a partir de ficheiro
5
6  library(plyr)
7
8  colors = c("gray")
9
10 histPercent <- function(dataTable, labelPvalue, colLabelName) {
11 IntDataTable <- as.numeric(dataTable)
12 #breaks=min(dataTable):max(dataTable)
13 H <- hist(IntDataTable, plot=FALSE)
14 H\$density <- with(H, 100 * density* diff(breaks)[1])
15 labs <- paste(round(H\$density), "\%", sep="")
16 labTitle<- paste("Histogram\n", labelPvalue, sep="")
17 plot(H,freq = FALSE, col=colors, labels = labs, main=labTitle, xlab=colLabelName, ylim
       ↪ =c(0, 1.08*max(H\$density)))
18 }
19
20 calculateChisq<-function(fileName, colName)
21 {
22 inputfile = read.csv(fileName, header = TRUE, sep=";")
23 attach(inputfile)
24
25 inputfile.df <- data.frame(inputfile)
26
27 dataColName <- inputfile.df[, c(colName)]
28
29 #print(inputfile.subColName)
```

```
30
31
32 #dataTable = as.data.frame(colName)
33
34 # tmp=subset(inputfile,inputfile\$colName)
35
36 dataTable = as.data.frame(colName)
37
38 #dataTable2 = dataTable[dataTable\$colName != "-10"]
39
40 dataTableFreq = count(dataTable,colName)
41
42 print(dataTableFreq)
43
44 nrRowsFreq = nrow(dataTableFreq)
45
46 # print(nrRowsFreq)
47
48 freqRel = 1/nrRowsFreq
49
50 # print(freqRel)
51
52 freqVector <- rep(dataTableFreq\$freq)
53
54 # print(freqVector)
55
56 pk=0
57 for (i in 1:nrRowsFreq)
58 {
59 (pk[i]=freqRel)
60 }
61
62 # print(pk)
63
64 nk =freqVector
65
66 testPvalue <- chisq.test(nk,p=pk, rescale.p = TRUE)
67
68 fileName<-file(paste(colName, "_Pvalue",".txt",sep=""))
69 capture.output(testPvalue, file=fileName, append=FALSE)
70 #write(t(testPvalue),file=fileConn)
71 #writeLines(c(testPvalue), fileConn)
72 #close(fileConn)
73
74
```

```
75  #print(testPvalue)
76  #str(testPvalue)
77
78  labelPvalue=(paste("p-value = ", testPvalue\$p.value, "\n", sep=""))
79  #labelPvalue=(paste(testPvalue\$method, "\n", "p-value = ", testPvalue\$p.value, "\n",
        ↪ sep=""))
80
81  #print(labelPvalue)
82
83  png(paste("Plot_",colName,"_Percent",".png",sep=""))
84  #png(paste("Plot_",colName,".png",sep=""))
85  histPercent(dataColName, labelPvalue, colName)
86  dev.off()
87  png(paste("Plot_",colName,"_Freq",".png",sep=""))
88  hist(dataColName,right=TRUE, col=colors, main="Histogram", xlab=colName)
89  # breaks=min(dataColName):max(dataColName)
90  #axis(side = 1,at = foo\$mids)
91  #lines(density(dataColName), col="blue")
92  #curve(dnorm(dataColName, mean=mean.dataColName, sd=sd.dataColName), col="darkblue",
        ↪ add=TRUE)
93  dev.off()
94  # print(hist(yn,right=FALSE, col=colors, main="yn title", xlab="yn label"))
95  # histPercent(yn)
96  #plot(density(yn))
97
98  detach(inputfile)
99  }
100 filenameA = 'simulationRandomsCsvA.csv'
101 filenameB = 'simulationRandomsCsvB.csv'
102 filenameC = 'simulationRandomsCsvC.csv'
103
104 u<-calculateChisq(filenameC, 'randomTimeDelayEnterWorkHours')
105 u<-calculateChisq(filenameC, 'randomTimeDelayEnterHomeHours')
106 u<-calculateChisq(filenameC, 'randomTimeDelayExitHomeHours')
107 u<-calculateChisq(filenameC, 'randomTimeDelayLazerHours')
108
109 u<-calculateChisq(filenameA, 'randomNumberOfLocalSystemsByUser')
110 u<-calculateChisq(filenameA, 'randomIndexUserType')
111
112 u<-calculateChisq(filenameB, 'randomUsersAtSameHome')
113 u<-calculateChisq(filenameB, 'randomUsersAtSameWork')
114
115
116
117 print("####################################")
```

C.2 Function_Plot.R

```
1
2  rm(list=ls(all=TRUE))
3
4  # leitura dos dados a partir de ficheiro
5  inputfile = read.csv("Data_Rand_TimeDelay_EnterWork.csv", header = TRUE)
6  attach(inputfile)
7
8  dataTable = as.data.frame(yn)
9
10 dataTableFreq = count(dataTable,"yn")
11
12 dataTableFreq
13
14 nrRowsFreq = nrow(dataTableFreq)
15
16 nrRowsFreq
17
18 freqRel = 1/nrRowsFreq
19
20 freqRel
21
22 freqVector <- rep(dataTableFreq$freq)
23
24 freqVector
25
26 pk=0
27 for (i in 1:nrRowsFreq)
28 {
29 (pk[i]=freqRel)
30 }
31 pk
32
33 nk =freqVector
34
35 chisq.test(nk,p=pk)
36
37
38 colors = c("orange")
39
40 png('Plot_Data_Rand_TimeDelay_EnterWork.png')
41 hist(yn,right=FALSE, col=colors, main="yn title", xlab="yn label")
42 dev.off()
43
```

```
#plot(density(yn))

detach(inputfile)

################################################
## Teste do qui-quadrado

# nk - frequencia observada na classe/categoria k
nk <- c(8, 9, 11, 12, 10)

# pk - frequencia relativa esperada na classe/categoria k
pk <- c(0.2, 0.2, 0.2, 0.2, 0.2)

# realizacao do teste
chisq.test(nk,p=pk)



################################################
## Teste Kolmogorov-Smirnov
## yn - valores observados

# geracao aleatoria de n observacoes de uma variavel uniforme de parametros a e b
# yn<-runif(n,a,b)



# parametros da distribuicao a- limite inferior, b - limite superior
a <- -10
b <- 10

# realizacao do teste
ks.test(yn,"punif", a, b)

# colors = c("red", "yellow", "green", "violet", "orange", "blue", "pink", "cyan")

colors = c("orange")

hist(yn,right=FALSE, col=colors, main="yn title", xlab="yn label")
plot(density(yn))

shapiro.test(yn);

qqnorm(yn);qqline(yn, col = 1)
```

D

Local System - BLE

D.1 Start_BLE_RaspPI.js

```
1
2
3  //Using the bleno module
4  var bleno = require('bleno');
5  var logger = require('nodejslogger');
6  logger.init({"file":"log_NodeJS.log", "mode":"DIE"});
7
8
9
10 var servicePhdUUUID = ['1720'];
11
12 var characteristicPhdUUUID = ['1721'];
13
14 var WSENSOR_UUID_TEMP_CHAR = ['1725']; //Actual Temperature - Write (8 bits)
15 var WSENSOR_UUID_PRES_CHAR = ['1726']; //Presences Number - Write (16 bits)
16 var WSENSOR_UUID_PING_CHAR = ['1727']; //Ping - Write (8 bits)
17
18
19
20 var idLocalSystem = 153
21
22 var idLocationRoom = 1
23 var idPrefTemperature = 1
24 var idPrefHumidity = 2
25
26
27
28
29
30 //Once bleno starts, begin advertising our BLE address
```

```
31 bleno.on('stateChange', function(state) {
32 console.log('State change: ' + state);
33 logger.info('State change: ' + state);
34 if (state === 'poweredOn') {
35 bleno.startAdvertising('MyDevice',servicePhdUUUID);
36 } else {
37 bleno.stopAdvertising();
38 }
39 });
40
41 //Notify the console that we've accepted a connection
42 bleno.on('accept', function(clientAddress) {
43 console.log("Accepted connection from address: " + clientAddress);
44 logger.info('Accepted connection from address: ' + clientAddress);
45 });
46
47 //Notify the console that we have disconnected from a client
48 bleno.on('disconnect', function(clientAddress) {
49 console.log("Disconnected from address: " + clientAddress);
50 logger.info('Disconnected from address: ' + clientAddress);
51 });
52
53 //When we begin advertising, create a new service and characteristic
54 bleno.on('advertisingStart', function(error) {
55 if (error) {
56 console.log("Advertising start error:" + error);
57 logger.error('Advertising start error:' + error);
58 } else {
59 console.log("Advertising start success");
60 logger.info('Advertising start success');
61 bleno.setServices([
62
63 // Define a new service
64 new bleno.PrimaryService({
65 uuid : '1720',
66 characteristics : [
67
68 // Define a new characteristic within that service
69 new bleno.Characteristic({
70 value : null,
71 uuid : '1721',
72 properties : ['notify', 'read', 'write'],
73
74 // If the client subscribes, we send out a message every 1 second
75 onSubscribe : function(maxValueSize, updateValueCallback) {
```

```
76  console.log("Device subscribed");
77  logger.info('Device subscribed');
78  this.intervalId = setInterval(function() {
79  console.log("Sending: Hi!");
80  logger.info('Sending: Hi!');
81  updateValueCallback(new Buffer("Hi!"));
82  }, 1000);
83  },
84
85  // If the client unsubscribes, we stop broadcasting the message
86  onUnsubscribe : function() {
87  console.log("Device unsubscribed");
88  logger.info('Device unsubscribed');
89  clearInterval(this.intervalId);
90  },
91
92  // Send a message back to the client with the characteristic's value
93  onReadRequest : function(offset, callback) {
94  console.log("Read request received");
95  logger.info('Read request received');
96  callback(this.RESULT_SUCCESS, new Buffer("Echo: " +
97  (this.value ? this.value.toString("utf-8") : "")));
98  },
99
100 // Accept a new value for the characterstic's value
101 onWriteRequest : function(data, offset, withoutResponse, callback) {
102 this.value = data;
103 console.log('Write request: value = ' + this.value);
104 logger.info('Write request: value = ' + this.value);
105 console.log('Write request: value Size = ' + data.length);
106 logger.info('Write request: value Size = ' + data.length);
107
108
109 console.log('Write request UUID');
110 logger.info('Write request UUID');
111 var exec = require('child_process').exec;
112 var child = exec('curl -X POST  http://pfo.dynip.sapo.pt:8080/WS/Rest/LocalSystemWS/
        ↪ RegisterUserAtLocalSystem/' + data + '/' + idLocalSystem,
113 function (error, stdout, stderr){
114 console.log('Output -> ' + stdout);
115 logger.info('Output -> ' + stdout);
116 if(error !== null){
117 console.log("Error -> "+error);
118 logger.error('Error -> '+error);
119 }
```

```
120 });
121
122 module.exports = child;
123 callback(this.RESULT_SUCCESS);
124 }
125
126 }),
127
128 new bleno.Characteristic({
129 value : null,
130 uuid : '1725',
131 properties : ['write'],
132
133 // Accept a new value for the characterstic's value
134 onWriteRequest : function(data, offset, withoutResponse, callback) {
135 this.value = data;
136 this.convertedData = this.value.toString('hex');
137 console.log('Write request: value = ' + this.convertedData);
138 logger.info('Write request: value = ' + this.convertedData);
139 console.log('Write request: value Size = ' + data.length);
140 logger.info('Write request: value Size = ' + data.length);
141
142 console.log('Write request Temperature');
143 logger.info('Write request Temperature');
144 var exec = require('child_process').exec;
145 var child = exec('curl -X POST  --user pfo:Pfo214 http://pfo.dynip.sapo.pt:8080/WS/
    ↪ Rest/LocalSystemWS/insertPreference_Live_Measurement/' + idLocalSystem + '/'+
    ↪ idLocationRoom + '/'+ idPrefTemperature + '/' + this.convertedData,
146 function (error, stdout, stderr){
147 console.log('Output -> ' + stdout);
148 logger.info('Output -> ' + stdout);
149 if(error !== null){
150 console.log("Error -> "+error);
151 logger.error('Error -> '+error);
152 }
153 });
154
155
156
157 module.exports = child;
158 callback(this.RESULT_SUCCESS);
159 }
160
161 }),
162
```

```
163 new bleno.Characteristic({
164 value : null,
165 uuid : '1726',
166 properties : ['write'],
167
168 // Accept a new value for the characterstic's value
169 onWriteRequest : function(data, offset, withoutResponse, callback) {
170 this.value = data;
171 this.convertedData = this.value.toString('hex');
172 console.log('Write request: value = ' + this.convertedData);
173 logger.info('Write request: value = ' + this.convertedData);
174 console.log('Write request: value Size = ' + data.length);
175 logger.info('Write request: value Size = ' + data.length);
176
177 console.log('Write request Presence');
178 logger.info('Write request Presence');
179 var exec = require('child_process').exec;
180 var child = exec('curl -X POST  --user pfo:Pfo214 http://pfo.dynip.sapo.pt:8080/WS/
    ↪ Rest/LocalSystemWS/insertPreference_Live_Measurement/' + idLocalSystem + '/'+
    ↪ idLocationRoom + '/'+ idPrefHumidity + '/' + this.convertedData,
181 function (error, stdout, stderr){
182 console.log('Output -> ' + stdout);
183 logger.info('Output -> ' + stdout);
184 if(error !== null){
185 console.log("Error -> "+error);
186 logger.error('Error -> '+error);
187 }
188 });
189
190 module.exports = child;
191 callback(this.RESULT_SUCCESS);
192 }
193
194 }),
195 new bleno.Characteristic({
196 value : null,
197 uuid : '1727',
198 properties : ['write'],
199
200 // Accept a new value for the characterstic's value
201 onWriteRequest : function(data, offset, withoutResponse, callback) {
202 this.value = data;
203 console.log('Write request: value = ' + this.value.toString('hex'));
204 logger.info('Write request: value = ' + this.value.toString('hex'));
205 console.log('Write request: value Size = ' + data.length);
```

```
206 logger.info('Write request: value Size = ' + data.length);
207
208 console.log('Write request Ping');
209 logger.info('Write request Ping');
210
211 callback(this.RESULT_SUCCESS);
212 }
213 })
214
215 ]
216 })
217 ]);
218 }
219 });
```

E

Data Layer - Web Services Server

E.1 Access.java

```
1
2  package com.wsphd.dao;
3
4  import java.net.CookieHandler;
5  import java.sql.Connection;
6
7  import java.sql.PreparedStatement;
8  import java.sql.ResultSet;
9  import java.sql.SQLException;
10 import java.sql.Time;
11 import java.sql.Timestamp;
12 import java.text.DecimalFormat;
13 import java.text.SimpleDateFormat;
14
15 import java.util.ArrayList;
16 import java.util.Calendar;
17 import java.util.List;
18 import java.util.Random;
19
20 import com.wsphd.dto.History_Pref_LocalSystem;
21 import com.wsphd.dto.History_Pref_User;
22 import com.wsphd.dto.LocalSystem;
23 import com.wsphd.dto.Local_System_Present_Preferences;
24 import com.wsphd.dto.Preferences_Card;
25 import com.wsphd.dto.User;
26 import com.wsphd.dto.Utilization_Preferences;
27
28 public class Access {
29
30 public ArrayList<User> getUsers(Connection con) throws SQLException {
```

```
31
32 ArrayList<User> userList = new ArrayList<>();
33
34 PreparedStatement stmt = con.prepareStatement("SELECT * FROM User_Information ORDER BY
       ↪ idUSer");
35 ResultSet rs = stmt.executeQuery();
36
37 try {
38 while (rs.next()) {
39 User userObj = new User();
40
41 userObj.setIdUser(rs.getInt("idUser"));
42
43 userObj.setUser_UUID(rs.getString("User_UUID"));
44
45 userList.add(userObj);
46
47 }
48
49 } catch (SQLException e)
50
51 {
52 e.printStackTrace();
53 } finally {
54 if (rs != null) {
55 rs.close();
56 }
57 if (stmt != null) {
58 stmt.close();
59 }
60 if (con != null) {
61 con.close();
62 }
63 }
64
65 return userList;
66
67 }
68
69 private List<Integer> getAll_IDUsers(Connection con) throws SQLException {
70
71 List<Integer> array_IDUsers = new ArrayList<>();
72
73 PreparedStatement stmt = con.prepareStatement("SELECT idUSer FROM User_Information
       ↪ ORDER BY idUSer");
```

```
74 ResultSet rs = stmt.executeQuery();
75
76 try {
77 while (rs.next()) {
78 array_IDUsers.add(rs.getInt("idUser"));
79 }
80
81 } catch (SQLException e)
82
83 {
84 e.printStackTrace();
85 } finally {
86 if (rs != null) {
87 rs.close();
88 }
89 if (stmt != null) {
90 stmt.close();
91 }
92 }
93
94 return array_IDUsers;
95
96 }
97
98 public String getUserUUID(Connection con, String idUser) throws SQLException {
99
100 String userUUID = null;
101
102
103
104 PreparedStatement stmt = con.prepareStatement("SELECT User_UUID FROM User_Information
    ↪ WHERE idUser = (?)");
105
106
107 stmt.setString(1, idUser);
108
109
110 ResultSet rs = stmt.executeQuery();
111
112 try {
113 while (rs.next()) {
114
115 userUUID = (rs.getString("User_UUID"));
116
117 }
```

```
118
119 } catch (SQLException e)
120
121 {
122 e.printStackTrace();
123 } finally {
124 if (rs != null) {
125 rs.close();
126 }
127 if (stmt != null) {
128 stmt.close();
129 }
130 if (con != null) {
131 con.close();
132 }
133 }
134
135 return userUUID;
136
137 }
138
139
140 public String validateUserUUID(Connection con, String useruuidToValidate) throws
        ↪ SQLException {
141
142 System.out.printf("UUID to Validate: %s \n", useruuidToValidate);
143
144 String userUUID = null;
145
146 PreparedStatement stmt = con.prepareStatement("SELECT User_UUID FROM User_Information
        ↪ WHERE User_UUID = (?)");
147
148 stmt.setString(1, useruuidToValidate);
149
150 ResultSet rs = stmt.executeQuery();
151 try {
152 while (rs.next()) {
153
154 userUUID = (rs.getString("User_UUID"));
155
156 System.out.printf("UUID Returned: %s \n", userUUID);
157 }
158
159 } catch (SQLException e)
160
```

```
161 {
162 e.printStackTrace();
163 return null;
164 } finally {
165 if (rs != null) {
166 rs.close();
167 }
168 if (stmt != null) {
169 stmt.close();
170 }
171 if (con != null) {
172 con.close();
173 }
174 }
175
176 return userUUID;
177
178 }
179
180
181 public ArrayList<LocalSystem> getLocalSystems(Connection con) throws SQLException {
182
183 ArrayList<LocalSystem> localSystemsList = new ArrayList<>();
184
185 PreparedStatement stmt = con.prepareStatement("SELECT * FROM Local_System_Information
    ↪ ORDER BY idLocal_System");
186 ResultSet rs = stmt.executeQuery();
187
188 try {
189 while (rs.next()) {
190 LocalSystem localSystemObj = new LocalSystem();
191
192 localSystemObj.setidLocal_System(rs.getInt("idLocal_System"));
193
194 localSystemObj.setDesc_LocalSystem(rs.getString("Desc_LocalSystem"));
195
196 localSystemObj.setGPS_Latitude(rs.getString("GPS_Latitude"));
197
198 localSystemObj.setGPS_Longitude(rs.getString("GPS_Longitude"));
199
200
201 localSystemsList.add(localSystemObj);
202 }
203
204 } catch (SQLException e)
```

```
205
206 {
207 e.printStackTrace();
208 } finally {
209 if (rs != null) {
210 rs.close();
211 }
212 if (stmt != null) {
213 stmt.close();
214 }
215 if (con != null) {
216 con.close();
217 }
218 }
219 return localSystemsList;
220
221 }
222
223
224 private List<Integer> getAllLocalSystemsIDS(Connection con) throws SQLException {
225
226 List<Integer> array_localSystemsIDS = new ArrayList<>();
227
228
229 PreparedStatement stmt = con.prepareStatement("SELECT idLocal_System FROM
    ↪ Local_System_Information ORDER BY idLocal_System");
230 ResultSet rs = stmt.executeQuery();
231
232 try {
233 while (rs.next()) {
234
235 array_localSystemsIDS.add(rs.getInt("idLocal_System"));
236
237 }
238
239 } catch (SQLException e)
240
241 {
242 e.printStackTrace();
243 } finally {
244 if (rs != null) {
245 rs.close();
246 }
247 if (stmt != null) {
248 stmt.close();
```

```
249 }
250
251 }
252 return array_localSystemsIDS;
253
254 }
255
256 private List<Integer> getAll_PreferenceIDS(Connection con) throws SQLException {
257
258 List<Integer> array_Preference_IDS = new ArrayList<>();
259
260 PreparedStatement stmt = con.prepareStatement("SELECT idPreference FROM
    ↪ Preferences_IDS ORDER BY idPreference");
261 ResultSet rs = stmt.executeQuery();
262
263 try {
264 while (rs.next()) {
265
266 array_Preference_IDS.add(rs.getInt("idPreference"));
267
268 }
269
270 } catch (SQLException e)
271
272 {
273 e.printStackTrace();
274 } finally {
275 if (rs != null) {
276 rs.close();
277 }
278 if (stmt != null) {
279 stmt.close();
280 }
281
282 }
283 return array_Preference_IDS;
284
285 }
286
287
288 private List<Integer> getAll_PreferenceRank_IDS(Connection con) throws SQLException {
289
290 List<Integer> array_PreferenceRank_IDS = new ArrayList<>();
291
```

```
292 PreparedStatement stmt = con.prepareStatement("SELECT idPreference_Rank FROM
        ↪ Preference_Rank ORDER BY idPreference_Rank");
293 ResultSet rs = stmt.executeQuery();
294
295 try {
296 while (rs.next()) {
297
298 array_PreferenceRank_IDS.add(rs.getInt("idPreference_Rank"));
299
300 }
301
302 } catch (SQLException e)
303
304 {
305 e.printStackTrace();
306 } finally {
307 if (rs != null) {
308 rs.close();
309 }
310 if (stmt != null) {
311 stmt.close();
312 }
313
314 }
315 return array_PreferenceRank_IDS;
316
317 }
318
319
320 public ArrayList<History_Pref_LocalSystem> getHistoryLocalSystem(Connection con)
        ↪ throws SQLException {
321
322 ArrayList<History_Pref_LocalSystem> history_Local_SystemList = new ArrayList<>();
323
324 PreparedStatement stmt = con.prepareStatement("SELECT * FROM History_Pref_Local_System
        ↪  ORDER BY ID_SystemFk");
325 ResultSet rs = stmt.executeQuery();
326
327 try {
328 while (rs.next()) {
329
330 History_Pref_LocalSystem history_Local_SystemObj = new History_Pref_LocalSystem();
331
332 history_Local_SystemObj.setID_SystemFk(rs.getInt("ID_SystemFk"));
333 history_Local_SystemObj.setTime_Period(rs.getTime("Time_Period"));
```

```
334 history_Local_SystemObj.setID_PreferencesFk(rs.getInt("ID_PreferencesFk"));
335 history_Local_SystemObj.setPreference_Value(rs.getString("Preference_Value"));
336
337 history_Local_SystemList.add(history_Local_SystemObj);
338
339 }
340
341 } catch (SQLException e)
342
343 {
344 e.printStackTrace();
345 } finally {
346 if (rs != null) {
347 rs.close();
348 }
349 if (stmt != null) {
350 stmt.close();
351 }
352 if (con != null) {
353 con.close();
354 }
355 }
356
357 return history_Local_SystemList;
358 }
359
360 public Boolean insertLocalSystem(Connection con, int idLocal_System, String
    ↪ Desc_LocalSystem, String GPS_Latitude, String GPS_Longitude) throws
    ↪ SQLException {
361
362 PreparedStatement stmt = con.prepareStatement("INSERT INTO Local_System_Information (
    ↪ idLocal_System, Desc_LocalSystem, GPS_Latitude, GPS_Longitude) VALUES (?, ?, ?,
    ↪  ?) "
363 + "ON DUPLICATE KEY UPDATE Desc_LocalSystem = VALUES(Desc_LocalSystem), GPS_Latitude =
    ↪  VALUES(GPS_Latitude), GPS_Longitude = VALUES(GPS_Longitude)");
364
365 stmt.setInt(1, idLocal_System);
366
367 stmt.setString(2, Desc_LocalSystem);
368
369 stmt.setString(3, GPS_Latitude);
370 stmt.setString(4, GPS_Longitude);
371
372 stmt.executeUpdate();
373
```

```
374 return true;
375
376 }
377
378
379 public ArrayList<Utilization_Preferences> getUtilization_Preferences(Connection con)
        ↪ throws SQLException {
380
381 ArrayList<Utilization_Preferences> utilization_Preferences_List = new ArrayList<>();
382
383 PreparedStatement stmt = con.prepareStatement("SELECT * FROM Utilization_Preferences
        ↪ ORDER BY User_ID_FK;");
384 ResultSet rs = stmt.executeQuery();
385
386 try {
387 while (rs.next()) {
388
389 Utilization_Preferences utilization_PreferencesObj = new Utilization_Preferences();
390
391 utilization_PreferencesObj.setLocal_ID_FK(rs.getInt("Local_ID_FK"));
392 utilization_PreferencesObj.setUser_ID_FK(rs.getInt("User_ID_FK"));
393 utilization_PreferencesObj.setID_Prefs_FK(rs.getInt("ID_Prefs_FK"));
394 utilization_PreferencesObj.setPreference_Value(rs.getString("Preference_Value"));
395
396 utilization_Preferences_List.add(utilization_PreferencesObj);
397
398 }
399
400 } catch (SQLException e)
401
402 {
403 e.printStackTrace();
404 } finally {
405 if (rs != null) {
406 rs.close();
407 }
408 if (stmt != null) {
409 stmt.close();
410 }
411 if (con != null) {
412 con.close();
413 }
414 }
415
416 return utilization_Preferences_List;
```

```
417 }
418
419 public ArrayList<Utilization_Preferences> getUtilization_PreferencesforUUID(Connection
        ↪  con, String userUUID) throws SQLException {
420
421 PreparedStatement stmt = con.prepareStatement("SELECT * FROM Utilization_Preferences
        ↪ WHERE User_ID_FK = (?)");
422
423 stmt.setString(1, userUUID);
424
425 ResultSet rs = stmt.executeQuery();
426
427 ArrayList<Utilization_Preferences> utilization_Preferences_List = new ArrayList<>();
428
429 try {
430 while (rs.next()) {
431
432 Utilization_Preferences utilization_PreferencesObj = new Utilization_Preferences();
433
434 utilization_PreferencesObj.setLocal_ID_FK(rs.getInt("Local_ID_FK"));
435 utilization_PreferencesObj.setUser_ID_FK(rs.getInt("User_ID_FK"));
436 utilization_PreferencesObj.setID_Prefs_FK(rs.getInt("ID_Prefs_FK"));
437 utilization_PreferencesObj.setPreference_Value(rs.getString("Preference_Value"));
438
439 utilization_Preferences_List.add(utilization_PreferencesObj);
440
441 }
442
443 } catch (SQLException e)
444
445 {
446 e.printStackTrace();
447 } finally {
448 if (rs != null) {
449 rs.close();
450 }
451 if (stmt != null) {
452 stmt.close();
453 }
454 if (con != null) {
455 con.close();
456 }
457 }
458 return utilization_Preferences_List;
459 }
```

```
460
461 public String createUser(Connection con, String UUID) throws SQLException {
462
463 System.out.printf("UUID Received to Create: %s \n", UUID);
464 String userUUID = null;
465
466
467 PreparedStatement stmt = con.prepareStatement("INSERT INTO User_Information (User_UUID
        ↪ ) VALUE (?)");
468
469 stmt.setString(1, UUID);
470
471 try {
472 stmt.executeUpdate();
473 {
474
475 userUUID = UUID;
476
477 System.out.printf("UUID Created: %s \n", userUUID);
478
479
480 }
481
482 } catch (SQLException e)
483
484 {
485 e.printStackTrace();
486 return null;
487 } finally {
488 if (stmt != null) {
489 stmt.close();
490 }
491 if (con != null) {
492 con.close();
493 }
494 }
495
496 return userUUID;
497 }
498
499 public Boolean insertHistoryPrefLocalSystem(Connection con, int ID_SystemFk, Time
        ↪ Time_Period, int ID_PreferencesFk, String Preference_Value) throws SQLException
        ↪  {
500
501
```

```
502 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_Local_System (
        ↪ ID_SystemFk,Time_Period,ID_PreferencesFk,Preference_Value) VALUES (?,?,?,?)");
503
504 stmt.setInt(1, ID_SystemFk);
505
506 stmt.setTime(2, Time_Period);
507
508 stmt.setInt(3, ID_PreferencesFk);
509 stmt.setString(4, Preference_Value);
510
511 stmt.executeUpdate();
512
513 return true;
514
515 }
516
517
518 public Boolean updatePrefCardforidUser(Connection con, int idUser, int IDPref, String
        ↪ PrefValue) throws SQLException {
519 System.out.printf("idUser Received to Update Pref. Card: %d \n", idUser);
520 System.out.printf("IDPref Received to Update Pref. Card: %d \n", IDPref);
521 System.out.printf("PrefValue Received to Update Pref. Card: %s \n", PrefValue);
522
523 PreparedStatement stmt = con.prepareStatement("UPDATE Preferences_Card SET
        ↪ Preference_Value = (?) WHERE ID_UserFk = (?) AND ID_PrefsFk = (?)");
524
525 stmt.setString(1, PrefValue);
526 stmt.setInt(2, idUser);
527 stmt.setInt(3, IDPref);
528
529 try {
530 stmt.executeUpdate();
531 {
532
533 System.out.printf("PrefValue %s for IFPref %d for idUser %d Updated! \n", PrefValue,
        ↪ IDPref, idUser);
534 return true;
535 }
536
537 } catch (SQLException e)
538
539 {
540 e.printStackTrace();
541 return false;
542 } finally {
```

```
543 if (stmt != null) {
544 stmt.close();
545 }
546 if (con != null) {
547 con.close();
548 }
549 }
550 }
551
552 public int getIdUserforUUID(Connection con, String UUID) throws SQLException {
553 System.out.printf("UUID Received to get userID: %s \n", UUID);
554
555 int idUser = 0;
556
557
558 PreparedStatement stmt = con.prepareStatement("SELECT idUser FROM User_Information
    ↪ WHERE User_UUID = (?)");
559
560 stmt.setString(1, UUID);
561
562
563 ResultSet rs = stmt.executeQuery();
564 try {
565 while (rs.next()) {
566 idUser = (rs.getInt("idUser"));
567 System.out.printf("idUser Returned: %d for UUID %s \n", idUser, UUID);
568
569 }
570
571 } catch (SQLException e)
572
573 {
574 e.printStackTrace();
575 idUser = -1;
576 } finally {
577 if (rs != null) {
578 rs.close();
579 }
580 if (stmt != null) {
581 stmt.close();
582 }
583 if (con != null) {
584 con.close();
585 }
586 }
```

```
587 System.out.printf("idUser Returned: %d for UUID %s \n", idUser, UUID);
588 return idUser;
589
590 }
591
592
593 public Boolean createUserPrefCardforidUser(Connection con, int idUser) throws
        ↪ SQLException {
594
595 Boolean boolResponse = null;
596
597 System.out.printf("idUser Received to create Pref. Card: %d \n", idUser);
598
599 PreparedStatement stmt = con.prepareStatement("INSERT INTO Preferences_Card (ID_UserFk
        ↪ ,ID_PrefsFk,Preference_Value, Preference_Rank) VALUES (?,?,?,?)");
600
601 Preferences_Card preferences_CardObj = new Preferences_Card();
602
603
604 List<Integer> preference_IDS_List = this.getAll_PreferenceIDS(con);
605
606 List<Integer> preferenceRank_IDS_List = this.getAll_PreferenceRank_IDS(con);
607
608
609 try {
610
611 for (int i = 1; i <= preference_IDS_List.size(); i++) {
612 if (i == 1 | i == 2) {
613 stmt.setInt(1, idUser);
614 stmt.setInt(2, i);
615 stmt.setString(3, preferences_CardObj.getRandomValuesforPrefCard(i));
616 stmt.setInt(4, 1);
617
618 stmt.executeUpdate();
619 } else {
620 for (int r = 1; r <= preferenceRank_IDS_List.size(); r++) {
621 stmt.setInt(1, idUser);
622 stmt.setInt(2, i);
623 stmt.setString(3, preferences_CardObj.getRandomValuesforPrefCard(i));
624 stmt.setInt(4, r);
625
626 stmt.executeUpdate();
627 }
628 }
629 }
```

```
630 System.out.printf("User PrefCard Created for idUser %s \n", idUser);
631 boolResponse = true;
632
633 } catch (SQLException e)
634
635 {
636 e.printStackTrace();
637 boolResponse = false;
638 } finally {
639 if (stmt != null) {
640 stmt.close();
641 }
642 if (con != null) {
643 con.close();
644 }
645 }
646
647 return boolResponse;
648 }
649
650 public ArrayList<Preferences_Card> getPrefCardforidUser(Connection con, int idUser)
        ↪ throws SQLException {
651
652 System.out.printf("idUser Received to get Pref. Card: %d \n", idUser);
653
654
655 ArrayList<Preferences_Card> prefCardList = new ArrayList<>();
656
657 PreparedStatement stmt = con.prepareStatement("SELECT ID_PrefsFk, Preference_Value
        ↪ FROM Preferences_Card WHERE ID_UserFk = (?) AND Preference_Rank= 1");
658
659
660 stmt.setInt(1, idUser);
661
662
663 ResultSet rs = stmt.executeQuery();
664
665 try {
666
667 while (rs.next()) {
668 System.out.printf("Get PrefCard for idUser %s \n", idUser);
669
670
671 Preferences_Card prefCardObj = new Preferences_Card();
672
```

```
673 prefCardObj.setID_PrefsFk(rs.getInt("ID_PrefsFk"));
674 prefCardObj.setPreference_Value(rs.getString("Preference_Value"));
675
676 prefCardList.add(prefCardObj);
677
678 }
679
680 } catch (SQLException e)
681
682 {
683 e.printStackTrace();
684 } finally {
685 if (rs != null) {
686 rs.close();
687 }
688 if (stmt != null) {
689 stmt.close();
690 }
691 if (con != null) {
692 }
693 }
694 System.out.printf("PrefCard  List %s \n", prefCardList);
695
696 return prefCardList;
697 }
698
699 public ArrayList<History_Pref_User> getAllHistoryPrefUsers(Connection con) throws
    ↪ SQLException {
700
701 ArrayList<History_Pref_User> history_Pref_UserList = new ArrayList<>();
702
703 PreparedStatement stmt = con.prepareStatement("SELECT * FROM History_Pref_User ORDER
    ↪ BY ID_User_FK");
704 ResultSet rs = stmt.executeQuery();
705
706 try {
707 while (rs.next()) {
708 History_Pref_User history_Pref_UserObj = new History_Pref_User();
709
710 history_Pref_UserObj.setID_System_FK(rs.getInt("ID_System_FK"));
711 history_Pref_UserObj.setTime(rs.getTimestamp("Time"));
712 history_Pref_UserObj.setTime_Period(rs.getTime("Time_Period"));
713 history_Pref_UserObj.setID_User_FK(rs.getInt("ID_User_FK"));
714 history_Pref_UserObj.setID_Prefs_FK(rs.getInt("ID_Prefs_FK"));
715 history_Pref_UserObj.setPreference_Value(rs.getString("Preference_Value"));
```

```
716 history_Pref_UserObj.setUserUpdateValue(rs.getInt("UserUpdateValue"));
717
718 history_Pref_UserList.add(history_Pref_UserObj);
719
720 }
721
722 } catch (SQLException e)
723
724 {
725 e.printStackTrace();
726 } finally {
727 if (rs != null) {
728 rs.close();
729 }
730 if (stmt != null) {
731 stmt.close();
732 }
733 if (con != null) {
734 con.close();
735 }
736 }
737
738 return history_Pref_UserList;
739
740 }
741
742 public Boolean initializeLocal_System_Present_Preferences(Connection con, int
    ↪ idLocalSystem) throws SQLException {
743 Boolean boolResponse = null;
744
745 System.out.printf("idLocalSystem Received to
    ↪ initializeLocal_System_Present_Preferences: %d \n", idLocalSystem);
746
747 Calendar calendar = Calendar.getInstance();
748
749 Time hour = Time.valueOf("00:00:00");
750 calendar.setTime(hour);
751
752
753 PreparedStatement stmt = con.prepareStatement("INSERT INTO
    ↪ Local_System_Present_Preferences (LocalSystemID_FK,Time_Period,IdPreference_FK,
    ↪ Preference_Value) "
754 + "VALUES (?,?,?,?) ON DUPLICATE KEY UPDATE Time_Period = VALUES(Time_Period),
    ↪ IdPreference_FK = VALUES(IdPreference_FK), Preference_Value = VALUES(
    ↪ Preference_Value)");
```

```
755
756 try {
757
758 for (int i = 1; i <= 48; i++) {
759 calendar.add(Calendar.MINUTE, 30);
760
761 SimpleDateFormat sdf = new SimpleDateFormat("HH:mm:ss");
762
763
764 Time hourToGetPeriod = Time.valueOf((sdf.format(calendar.getTime())));
765
766 System.out.println(hourToGetPeriod);
767
768 System.out.println(sdf.format(calendar.getTime()));
769
770
771 Preferences_Card preferences_CardObj = new Preferences_Card();
772
773 List<Integer> preference_IDS_List = this.getAll_PreferenceIDS(con);
774
775 for (int j = 1; j <= preference_IDS_List.size(); j++) {
776 stmt.setInt(1, idLocalSystem);
777 stmt.setTime(2, hourToGetPeriod);
778 stmt.setInt(3, j);
779 stmt.setString(4, preferences_CardObj.getRandomValuesforPrefCard(j));
780
781 stmt.executeUpdate();
782 }
783
784 }
785 System.out.printf("initializeLocal_System_Present_Preferences for idLocalSystem %s \n"
    ↪ , idLocalSystem);
786 boolResponse = true;
787 } catch (SQLException e)
788
789 {
790 e.printStackTrace();
791 boolResponse = false;
792 } finally {
793 if (stmt != null) {
794 stmt.close();
795 }
796 if (con != null) {
797 con.close();
798 }
```

```
799 }
800
801 return boolResponse;
802 }
803
804 public Boolean insertHistoryForidUserAndLocalSystem(Connection con, int idUser, int
        ↪ idLocalSystem, Time Time_Period) throws SQLException {
805
806 Boolean boolResponse = null;
807
808 System.out.printf("idUser %d LocalSytemID %d and TimePeriod %s Received to create
        ↪ HistoryPrefUser! \n", idUser, idLocalSystem, Time_Period);
809
810 ArrayList<History_Pref_User> arrayOfPreferencesOfUser =
        ↪ getHistoryPrefUserINLocalSystemAndTimePeriod(con, idUser, idLocalSystem,
        ↪ Time_Period);
811
812 if (arrayOfPreferencesOfUser.isEmpty()) {
813 ArrayList<Preferences_Card> arrayPreferencesCardOfUser = getPrefCardforidUser(con,
        ↪ idUser);
814 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_User (
        ↪ ID_User_FK,ID_System_FK,Time_Period,ID_Prefs_FK,Preference_Value) VALUES
        ↪ (?,?,?,?,?)");
815
816 try {
817 for (Preferences_Card dataPreferencesCard : arrayPreferencesCardOfUser) {
818
819 stmt.setInt(1, idUser);
820 stmt.setInt(2, idLocalSystem);
821 stmt.setTime(3, Time_Period);
822 stmt.setInt(4, dataPreferencesCard.getID_PrefsFk());
823 stmt.setString(5, dataPreferencesCard.getPreference_Value());
824
825 stmt.executeUpdate();
826
827 System.out.printf("History User Created for idUser %d at LocalSystemID %d and
        ↪ TimePeriod %s With PrefID %d and PrefValue %s ! \n", idUser, idLocalSystem,
        ↪ Time_Period, dataPreferencesCard.getID_PrefsFk(), dataPreferencesCard.
        ↪ getPreference_Value());
828 }
829
830 boolResponse = true;
831 } catch (SQLException e)
832
833 {
```

```
834 e.printStackTrace();
835 boolResponse = false;
836 } finally {
837 if (stmt != null) {
838 stmt.close();
839 }
840 if (con != null) {
841 con.close();
842 }
843 }
844 } else {
845
846 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_User (
        ↪ ID_User_FK,ID_System_FK,Time_Period,ID_Prefs_FK,Preference_Value) VALUES
        ↪ (?,?,?,?,?)");
847
848
849 try {
850 for (History_Pref_User dataPreferences : arrayOfPreferencesOfUser) {
851
852 stmt.setInt(1, idUser);
853 stmt.setInt(2, idLocalSystem);
854 stmt.setTime(3, Time_Period);
855 stmt.setInt(4, dataPreferences.getID_Prefs_FK());
856 stmt.setString(5, dataPreferences.getPreference_Value());
857
858 stmt.executeUpdate();
859
860 System.out.printf("History User Created for idUser %d at LocalSystemID %d and
        ↪ TimePeriod %s With PrefID %d and PrefValue %s ! \n", idUser, idLocalSystem,
        ↪ Time_Period, dataPreferences.getID_Prefs_FK(), dataPreferences.
        ↪ getPreference_Value());
861
862
863 }
864
865 boolResponse = true;
866 } catch (SQLException e)
867
868 {
869 e.printStackTrace();
870 boolResponse = false;
871 } finally {
872 if (stmt != null) {
873 stmt.close();
```

```
874 }
875 if (con != null) {
876 con.close();
877 }
878 }
879 }
880 return boolResponse;
881 }
882
883
884 public Boolean insertHistoryPrefUserWithIdUserAndTime(Connection con, int idUser, int
        ↪ idLocalSystem, Timestamp TimeToInsert, Time TimePeriod) throws SQLException {
885
886 Boolean boolResponse = null;
887
888 System.out.printf("idUser %d LocalSystemID %d TimePeriod %s Timestamp %s Received to
        ↪ create HistoryPrefUserWithIdUserAndTime! \n", idUser, idLocalSystem, TimePeriod
        ↪ , TimeToInsert);
889
890 ArrayList<History_Pref_User> arrayOfPreferencesOfUser =
        ↪ getHistoryPrefUserINLocalSystemAndTimePeriod(con, idUser, idLocalSystem,
        ↪ TimePeriod);
891
892 System.out.print(arrayOfPreferencesOfUser);
893
894 int originalPrefValue;
895 String changedPrefValue;
896
897 if (arrayOfPreferencesOfUser.isEmpty()) {
898 ArrayList<Preferences_Card> arrayPreferencesCardOfUser = getPrefCardforidUser(con,
        ↪ idUser);
899
900
901 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_User (
        ↪ ID_User_FK, ID_System_FK, Time, Time_Period, ID_Prefs_FK,Preference_Value)
        ↪ VALUES (?,?,?,?,?,?)");
902
903 System.out.printf("Teste \n");
904
905
906 Random random0 = new Random();
907 Random random1 = new Random();
908 Random random2 = new Random();
909 Random random3 = new Random();
910
```

```
911
912 int randomPercentageMin = 1;
913 int randomPercentageMax = 100;
914
915 int randomIdPrefMin = 1;
916 int randomIdPrefMax = 2;
917
918
919 int randomIdPreference = random0.nextInt((randomIdPrefMax - randomIdPrefMin) + 1) +
        ↪ randomIdPrefMin;
920
921
922 int delayPrefValueMinA = -5;
923 int delayPrefValueMaxA = 5;
924
925 int delayPrefValueMinB = -3;
926 int delayPrefValueMaxB = 3;
927
928
929 int randomPrefValueA = random1.nextInt((delayPrefValueMaxA - delayPrefValueMinA) + 1)
        ↪ + delayPrefValueMinA;
930 int randomPrefValueB = random2.nextInt((delayPrefValueMaxB - delayPrefValueMinB) + 1)
        ↪ + delayPrefValueMinB;
931
932
933 int randomPercentage = random3.nextInt((randomPercentageMax - randomPercentageMin) +
        ↪ 1) + randomPercentageMin;
934
935
936 try {
937 for (Preferences_Card dataPreferencesCard : arrayPreferencesCardOfUser) {
938 if (dataPreferencesCard.getID_PrefsFk() == randomIdPreference) {
939
940 if (randomPercentage <= 5) {
941 System.out.printf("5 -> " + randomPercentage);
942
943 originalPrefValue = Integer.valueOf(dataPreferencesCard.getPreference_Value());
944
945 changedPrefValue = String.valueOf(originalPrefValue + randomPrefValueA);
946
947 stmt.setInt(1, idUser);
948 stmt.setInt(2, idLocalSystem);
949 stmt.setTimestamp(3, TimeToInsert);
950 stmt.setTime(4, TimePeriod);
951 stmt.setInt(5, dataPreferencesCard.getID_PrefsFk());
```

```
952 stmt.setString(6, changedPrefValue);
953
954 insertWeekHistory(con,idUser,idLocalSystem,TimeToInsert, TimePeriod,
        ↪ dataPreferencesCard.getID_PrefsFk(), changedPrefValue);
955
956
957 } else if (randomPercentage > 5 && randomPercentage <= 20) {
958
959 System.out.printf("20 -> " + randomPercentage);
960
961 originalPrefValue = Integer.valueOf(dataPreferencesCard.getPreference_Value());
962
963
964 changedPrefValue = String.valueOf(originalPrefValue + randomPrefValueB);
965
966 stmt.setInt(1, idUser);
967 stmt.setInt(2, idLocalSystem);
968 stmt.setTimestamp(3, TimeToInsert);
969 stmt.setTime(4, TimePeriod);
970 stmt.setInt(5, dataPreferencesCard.getID_PrefsFk());
971 stmt.setString(6, changedPrefValue);
972
973 } else if (randomPercentage > 20) {
974
975 System.out.printf("80 -> " + randomPercentage);
976
977 stmt.setInt(1, idUser);
978 stmt.setInt(2, idLocalSystem);
979 stmt.setTimestamp(3, TimeToInsert);
980 stmt.setTime(4, TimePeriod);
981 stmt.setInt(5, dataPreferencesCard.getID_PrefsFk());
982 stmt.setString(6, dataPreferencesCard.getPreference_Value());
983 }
984 } else {
985 stmt.setInt(1, idUser);
986 stmt.setInt(2, idLocalSystem);
987 stmt.setTimestamp(3, TimeToInsert);
988 stmt.setTime(4, TimePeriod);
989 stmt.setInt(5, dataPreferencesCard.getID_PrefsFk());
990 stmt.setString(6, dataPreferencesCard.getPreference_Value());
991 }
992
993
994 stmt.executeUpdate();
995
```

```
996 System.out.printf("History User Created for idUser %d at LocalSystemID %d and
        ↪ TimePeriod %s With PrefID %d and PrefValue %s ! \n", idUser, idLocalSystem,
        ↪ TimePeriod, dataPreferencesCard.getID_PrefsFk(), dataPreferencesCard.
        ↪ getPreference_Value());
997 }
998
999 boolResponse = true;
1000 } catch (SQLException e)
1001
1002 {
1003 e.printStackTrace();
1004 boolResponse = false;
1005 } finally {
1006 if (stmt != null) {
1007 stmt.close();
1008 }
1009 if (con != null) {
1010 con.close();
1011 }
1012 }
1013 } else {
1014 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_User (
        ↪ ID_User_FK, ID_System_FK, Time, Time_Period, ID_Prefs_FK,Preference_Value)
        ↪ VALUES (?,?,?,?,?,?)");
1015
1016 System.out.printf("Teste \n");
1017
1018
1019 Random random0 = new Random();
1020 Random random1 = new Random();
1021 Random random2 = new Random();
1022 Random random3 = new Random();
1023
1024
1025 int randomPercentageMin = 1;
1026 int randomPercentageMax = 100;
1027
1028 int randomIdPrefMin = 1;
1029 int randomIdPrefMax = 2;
1030
1031
1032 int randomIdPreference = random0.nextInt((randomIdPrefMax - randomIdPrefMin) + 1) +
        ↪ randomIdPrefMin;
1033
1034 int delayPrefValueMinA = -5;
```

```
1035 int delayPrefValueMaxA = 5;
1036
1037 int delayPrefValueMinB = -3;
1038 int delayPrefValueMaxB = 3;
1039
1040
1041 double randomPrefValueA = delayPrefValueMinA + random1.nextDouble() *
         ↪ delayPrefValueMaxA;
1042 double randomPrefValueB = delayPrefValueMinB + random2.nextDouble() *
         ↪ delayPrefValueMaxB;
1043
1044 int randomPercentage = random3.nextInt((randomPercentageMax - randomPercentageMin) +
         ↪ 1) + randomPercentageMin;
1045
1046 DecimalFormat df = new DecimalFormat("0.00");
1047
1048 try {
1049 for (History_Pref_User dataPreferences : arrayOfPreferencesOfUser) {
1050 if (dataPreferences.getID_Prefs_FK() == randomIdPreference) {
1051
1052 if (randomPercentage <= 5) {
1053 System.out.printf("5 -> " + randomPercentage);
1054
1055 originalPrefValue = Integer.valueOf(dataPreferences.getPreference_Value());
1056
1057 changedPrefValue = String.valueOf(df.format(originalPrefValue + originalPrefValue *
         ↪ randomPrefValueA));
1058
1059 stmt.setInt(1, idUser);
1060 stmt.setInt(2, idLocalSystem);
1061 stmt.setTimestamp(3, TimeToInsert);
1062 stmt.setTime(4, TimePeriod);
1063 stmt.setInt(5, dataPreferences.getID_Prefs_FK());
1064 stmt.setString(6, changedPrefValue);
1065
1066
1067 } else if (randomPercentage > 5 && randomPercentage <= 20) {
1068
1069 System.out.printf("20 -> " + randomPercentage);
1070
1071 originalPrefValue = Integer.valueOf(dataPreferences.getPreference_Value());
1072
1073 changedPrefValue = String.valueOf(df.format(originalPrefValue + originalPrefValue *
         ↪ randomPrefValueB));
1074
```

```
1075 stmt.setInt(1, idUser);
1076 stmt.setInt(2, idLocalSystem);
1077 stmt.setTimestamp(3, TimeToInsert);
1078 stmt.setTime(4, TimePeriod);
1079 stmt.setInt(5, dataPreferences.getID_Prefs_FK());
1080 stmt.setString(6, changedPrefValue);
1081
1082 } else if (randomPercentage > 20) {
1083
1084 System.out.printf("80 -> " + randomPercentage);
1085
1086 stmt.setInt(1, idUser);
1087 stmt.setInt(2, idLocalSystem);
1088 stmt.setTimestamp(3, TimeToInsert);
1089 stmt.setTime(4, TimePeriod);
1090 stmt.setInt(5, dataPreferences.getID_Prefs_FK());
1091 stmt.setString(6, dataPreferences.getPreference_Value());
1092 }
1093 } else {
1094 stmt.setInt(1, idUser);
1095 stmt.setInt(2, idLocalSystem);
1096 stmt.setTimestamp(3, TimeToInsert);
1097 stmt.setTime(4, TimePeriod);
1098 stmt.setInt(5, dataPreferences.getID_Prefs_FK());
1099 stmt.setString(6, dataPreferences.getPreference_Value());
1100 }
1101
1102
1103 stmt.executeUpdate();
1104
1105 System.out.printf("History User Created for idUser %d at LocalSystemID %d and
      ↪ TimePeriod %s With PrefID %d and PrefValue %s ! \n", idUser, idLocalSystem,
      ↪ TimePeriod, dataPreferences.getID_Prefs_FK(), dataPreferences.
      ↪ getPreference_Value());
1106 }
1107
1108 boolResponse = true;
1109 } catch (SQLException e)
1110
1111 {
1112 e.printStackTrace();
1113 boolResponse = false;
1114 } finally {
1115 if (stmt != null) {
1116 stmt.close();
```

```
1117 }
1118 if (con != null) {
1119 con.close();
1120 }
1121 }
1122 }
1123
1124 return boolResponse;
1125 }
1126
1127
1128 private ArrayList<History_Pref_User> getHistoryPrefUserINLocalSystemAndTimePeriod(
     ↪ Connection con, int idUser, int idLocalSystem, Time TimePeriod) throws
     ↪ SQLException {
1129
1130 ArrayList<History_Pref_User> history_Pref_UserList = new ArrayList<>();
1131
1132
1133 PreparedStatement stmt = con.prepareStatement("SELECT ID_Prefs_FK, Preference_Value
     ↪ FROM Utilization_Preferences WHERE User_ID_FK= (?) AND Local_ID_FK=(?) AND
     ↪ Time_Period =(?)");
1134
1135 stmt.setInt(1, idUser);
1136 stmt.setInt(2, idLocalSystem);
1137 stmt.setTime(3, TimePeriod);
1138
1139 ResultSet rs = stmt.executeQuery();
1140
1141
1142 try {
1143 while (rs.next()) {
1144 History_Pref_User history_Pref_UserObj = new History_Pref_User();
1145
1146 history_Pref_UserObj.setID_Prefs_FK(rs.getInt("ID_Prefs_FK"));
1147 history_Pref_UserObj.setPreference_Value(rs.getString("Preference_Value"));
1148
1149 history_Pref_UserList.add(history_Pref_UserObj);
1150
1151 }
1152
1153 } catch (SQLException e)
1154
1155 {
1156 e.printStackTrace();
1157 } finally {
```

```
1158 if (rs != null) {
1159 rs.close();
1160 }
1161 if (stmt != null) {
1162 stmt.close();
1163 }
1164 
1165 }
1166 
1167 System.out.printf("Estou aqui history_Pref_UserList %s \n", history_Pref_UserList);
1168 
1169 return history_Pref_UserList;
1170 
1171 }
1172 
1173 public ArrayList<Local_System_Present_Preferences> getLocal_System_Present_Preferences
     ↪ (Connection con, int idLocalSystem, Time TimePeriod) throws SQLException {
1174 
1175 ArrayList<Local_System_Present_Preferences> localSystem_Present_Pref_List = new
     ↪ ArrayList<>();
1176 
1177 
1178 PreparedStatement stmt = con.prepareStatement("SELECT IdPreference_FK,
     ↪ Preference_Value FROM Local_System_Present_Preferences WHERE LocalSystemID_FK =
     ↪  (?) AND Time_Period=(?)");
1179 
1180 
1181 stmt.setInt(1, idLocalSystem);
1182 stmt.setTime(2, TimePeriod);
1183 
1184 ResultSet rs = stmt.executeQuery();
1185 
1186 
1187 try {
1188 while (rs.next()) {
1189 Local_System_Present_Preferences localSystem_Present_Pref_Obj = new
     ↪ Local_System_Present_Preferences();
1190 
1191 localSystem_Present_Pref_Obj.setIdPreference_FK(rs.getInt("IdPreference_FK"));
1192 localSystem_Present_Pref_Obj.setPreference_Value(rs.getString("Preference_Value"));
1193 
1194 localSystem_Present_Pref_List.add(localSystem_Present_Pref_Obj);
1195 
1196 }
1197 
```

```
1198 } catch (SQLException e)
1199
1200 {
1201 e.printStackTrace();
1202 } finally {
1203 if (rs != null) {
1204 rs.close();
1205 }
1206 if (stmt != null) {
1207 stmt.close();
1208 }
1209
1210 }
1211
1212 System.out.printf("Estou aqui localSystem_Present_Pref_List %s \n",
     ↪ localSystem_Present_Pref_List);
1213
1214 return localSystem_Present_Pref_List;
1215
1216 }
1217
1218 public Boolean updateLocalSystemPresentPreferences(Connection con, int idLocalSystem,
     ↪ int idUser, int idPreference, String PreferenceValue) throws SQLException {
1219 int UserUpdateValue = 1;
1220
1221 Boolean boolResponse = null;
1222
1223 System.out.printf("idLocalSystem %d For User %d  Received to
     ↪ updateLocalSystemPresentPreferences \n", idLocalSystem, idUser);
1224
1225
1226 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_User (
     ↪ ID_User_FK,ID_System_FK,ID_Prefs_FK,Preference_Value, UserUpdateValue) VALUES
     ↪ (?,?,?,?,?)");
1227
1228
1229 stmt.setInt(1, idUser);
1230 stmt.setInt(2, idLocalSystem);
1231 stmt.setInt(3, idPreference);
1232 stmt.setString(4, PreferenceValue);
1233 stmt.setInt(5, UserUpdateValue);
1234
1235 try {
1236 stmt.executeUpdate();
1237
```

```
1238 {
1239 boolResponse = true;
1240 System.out.printf("UPDATED IDPref %d with PrefValue %s at LocalSystemID %d for idUser
         ↪ %d \n", idPreference, PreferenceValue, idLocalSystem, idUser);
1241
1242 }
1243
1244 } catch (SQLException e)
1245
1246 {
1247 e.printStackTrace();
1248 boolResponse = false;
1249 } finally {
1250
1251 if (stmt != null) {
1252 stmt.close();
1253 }
1254 if (con != null) {
1255 con.close();
1256 }
1257 }
1258
1259 return boolResponse;
1260 }
1261
1262 public int getLastLocalSystemIDForUser(Connection con, int idUser) throws SQLException
         ↪  {
1263
1264 int idLocalSystem = -1;
1265
1266 System.out.printf("idUser Received to getLastLocalSystemIDForUser: %d \n", idUser);
1267
1268 PreparedStatement stmt = con.prepareStatement("SELECT ID_System_FK FROM
         ↪ History_Pref_User WHERE ID_User_FK = (?) ORDER BY Time DESC LIMIT 1");
1269
1270
1271 stmt.setInt(1, idUser);
1272
1273 ResultSet rs = stmt.executeQuery();
1274
1275 try {
1276 while (rs.next()) {
1277
1278 idLocalSystem = (rs.getInt("ID_System_FK"));
1279 }
```

```
1280
1281
1282 } catch (SQLException e)
1283
1284 {
1285 e.printStackTrace();
1286 idLocalSystem = -1;
1287 } finally {
1288 if (rs != null) {
1289 rs.close();
1290 }
1291 if (stmt != null) {
1292 stmt.close();
1293 }
1294 if (con != null)
1295 con.close();
1296 }
1297
1298 System.out.printf("Last idLocalSystem is %d for idUser %d \n", idLocalSystem, idUser);
1299 return idLocalSystem;
1300 }
1301
1302 public Boolean initializeUtilization_PreferencesForAllSystem(Connection con) throws
        ↪ SQLException {
1303 Boolean boolResponse = null;
1304
1305 System.out.printf("initializeUtilization_Preferences A \n");
1306
1307 Calendar calendar = Calendar.getInstance();
1308
1309 Time hour = Time.valueOf("00:00:00");
1310 calendar.setTime(hour);
1311
1312 int contador_Inserts = 0;
1313 long tStart;
1314 long tEnd;
1315
1316
1317
1318 PreparedStatement stmt = con.prepareStatement("INSERT INTO Utilization_Preferences (
        ↪ User_ID_FK,Local_ID_FK,Time_Period,ID_Prefs_FK,Preference_Value) "
1319 + "VALUES (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?)
        ↪ , (?,?,?,?,?) "
1320
```

```
1321 + "ON DUPLICATE KEY UPDATE User_ID_FK=VALUES(User_ID_FK), Local_ID_FK=VALUES(
         ↪ Local_ID_FK), Time_Period = VALUES(Time_Period), ID_Prefs_FK = VALUES(
         ↪ ID_Prefs_FK), Preference_Value = VALUES(Preference_Value)");
1322
1323
1324 tStart = System.currentTimeMillis();
1325 try {
1326
1327 for (int i = 1; i <= 48; i++) {
1328 calendar.add(Calendar.MINUTE, 30);
1329
1330 SimpleDateFormat sdf = new SimpleDateFormat("HH:mm:ss");
1331
1332
1333 Time hourToGetPeriod = Time.valueOf((sdf.format(calendar.getTime())));
1334
1335 System.out.println(hourToGetPeriod);
1336
1337 System.out.println(sdf.format(calendar.getTime()));
1338
1339
1340 List<Integer> userList = this.getAll_IDUsers(con);
1341
1342
1343 List<Integer> localSystemList = this.getAllLocalSystemsIDS(con);
1344
1345
1346 for (int l = 0; l < localSystemList.size(); l++) {
1347 int idLocalSystem = localSystemList.get(l);
1348
1349
1350 for (int u = 0; u < userList.size(); u++) {
1351 int idUser = userList.get(u)
1352
1353
1354 Preferences_Card preferences_CardObj = new Preferences_Card();
1355
1356 List<Integer> preference_IDS_List = this.getAll_PreferenceIDS(con);
1357
1358
1359 int s = 1;
1360 for (int j = 1; j <= preference_IDS_List.size(); j++) {
1361 System.out.printf("initializeUtilization_Preferences With UserID %d at LocalSystem %d
         ↪ For Time %s With PrefID %d and PrefValue %s \n", idUser, idLocalSystem,
         ↪ hourToGetPeriod, j, preferences_CardObj.getRandomValuesforPrefCard(j));
```

```
1362
1363
1364 stmt.setInt(s++, idUser);
1365 stmt.setInt(s++, idLocalSystem);
1366 stmt.setTime(s++, hourToGetPeriod);
1367 stmt.setInt(s++, j);
1368 stmt.setString(s++, preferences_CardObj.getRandomValuesforPrefCard(j));
1369
1370 }
1371
1372 stmt.executeUpdate();
1373 contador_Inserts++;
1374
1375 }
1376 }
1377
1378
1379 }
1380
1381 tEnd = System.currentTimeMillis();
1382 long tDelta = tEnd - tStart;
1383 double elapsedMinutes = (tDelta / 1000.0) / 60;
1384 System.out.printf(" Method initializeUtilization_Preferences Done With %d Inserts In %
     ↪ f Minutes!", contador_Inserts, elapsedMinutes);
1385
1386 boolResponse = true;
1387 } catch (SQLException e)
1388
1389 {
1390 e.printStackTrace();
1391 boolResponse = false;
1392 } finally {
1393 if (stmt != null) {
1394 stmt.close();
1395 }
1396 if (con != null) {
1397 con.close();
1398 }
1399 }
1400
1401 return boolResponse;
1402 }
1403
1404
```

```
1405 public Boolean initializeUtilization_PreferencesForNewUser(Connection con, int idUser)
         ↪  throws SQLException {
1406 Boolean boolResponse = null;
1407
1408 System.out.printf("initializeUtilization_Preferences A \n");
1409
1410 Calendar calendar = Calendar.getInstance();
1411
1412 Time hour = Time.valueOf("00:00:00");
1413 calendar.setTime(hour);
1414
1415 int contador_Inserts = 0;
1416 long tStart;
1417 long tEnd;
1418
1419
1420
1421 PreparedStatement stmt = con.prepareStatement("INSERT INTO Utilization_Preferences (
         ↪ User_ID_FK,Local_ID_FK,Time_Period,ID_Prefs_FK,Preference_Value) "
1422 + "VALUES (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?), (?,?,?,?,?)
         ↪ , (?,?,?,?,?) "
1423
1424 + "ON DUPLICATE KEY UPDATE User_ID_FK=VALUES(User_ID_FK), Local_ID_FK=VALUES(
         ↪ Local_ID_FK), Time_Period = VALUES(Time_Period), ID_Prefs_FK = VALUES(
         ↪ ID_Prefs_FK), Preference_Value = VALUES(Preference_Value)");
1425
1426
1427 tStart = System.currentTimeMillis();
1428 try {
1429
1430 for (int i = 1; i <= 48; i++) {
1431 calendar.add(Calendar.MINUTE, 30);
1432
1433 SimpleDateFormat sdf = new SimpleDateFormat("HH:mm:ss");
1434
1435
1436 Time hourToGetPeriod = Time.valueOf((sdf.format(calendar.getTime())));
1437
1438 System.out.println(hourToGetPeriod);
1439
1440
1441 List<Integer> localSystemList = this.getAllLocalSystemsIDS(con);
1442
1443 List<Integer> preference_IDS_List = this.getAll_PreferenceIDS(con);
1444
```

```
1445
1446 for (int l = 0; l < localSystemList.size(); l++) {
1447 int idLocalSystem = localSystemList.get(l);
1448
1449 Preferences_Card preferences_CardObj = new Preferences_Card();
1450
1451 int s = 1;
1452 for (int j = 1; j <= preference_IDS_List.size(); j++) {
1453 System.out.printf("initializeUtilization_Preferences With UserID %d at LocalSystem %d
    ↪ For Time %s With PrefID %d and PrefValue %s \n", idUser, idLocalSystem,
    ↪ hourToGetPeriod, j, preferences_CardObj.getRandomValuesforPrefCard(j));
1454
1455 stmt.setInt(s++, idUser);
1456 stmt.setInt(s++, idLocalSystem);
1457 stmt.setTime(s++, hourToGetPeriod);
1458 stmt.setInt(s++, j);
1459 stmt.setString(s++, preferences_CardObj.getRandomValuesforPrefCard(j));
1460 }
1461
1462 stmt.executeUpdate();
1463 contador_Inserts++;
1464
1465 }
1466
1467 }
1468
1469 tEnd = System.currentTimeMillis();
1470 long tDelta = tEnd - tStart;
1471 double elapsedMinutes = (tDelta / 1000.0) / 60;
1472 System.out.printf(" Method initializeUtilization_Preferences Done With %d Inserts In %
    ↪ f Minutes! \n", contador_Inserts, elapsedMinutes);
1473
1474 boolResponse = true;
1475 } catch (SQLException e)
1476
1477 {
1478 e.printStackTrace();
1479 boolResponse = false;
1480 } finally {
1481 if (stmt != null) {
1482 stmt.close();
1483 }
1484 if (con != null) {
1485 con.close();
1486 }
```

```
1487 }
1488
1489 return boolResponse;
1490 }
1491
1492 public String RegisterUserAtLocalSystem(Connection con, String useruuidToValidate)
     ↪ throws SQLException {
1493
1494 System.out.printf("UUID to Validate Register At Local System: %s \n",
     ↪ useruuidToValidate);
1495
1496 String userUUID = null;
1497
1498
1499 PreparedStatement stmt = con.prepareStatement("SELECT User_UUID FROM User_Information
     ↪ WHERE User_UUID = (?)");
1500
1501 stmt.setString(1, useruuidToValidate);
1502
1503 ResultSet rs = stmt.executeQuery();
1504 try {
1505 while (rs.next()) {
1506
1507 userUUID = (rs.getString("User_UUID"));
1508
1509 System.out.printf("UUID Returned: %s \n", userUUID);
1510 }
1511
1512 } catch (SQLException e)
1513
1514 {
1515 e.printStackTrace();
1516 return null;
1517 } finally {
1518 if (rs != null) {
1519 rs.close();
1520 }
1521 if (stmt != null) {
1522 stmt.close();
1523 }
1524 if (con != null) {
1525 con.close();
1526 }
1527 }
1528
```

```
1529 return userUUID;
1530
1531 }
1532
1533
1534 public Boolean insertPreference_Live_Measurement(Connection con, int ID_LocalSystem,
         ↪ int ID_Location_Measured, Time Time_Period, int ID_Prefer_Measured, String
         ↪ Pref_Value_Measured) throws SQLException {
1535 Boolean boolResponse;
1536
1537 PreparedStatement stmt = con.prepareStatement("INSERT INTO
         ↪ Preferences_Live_Measurement (ID_LocalSystem,ID_Location_Measured,Time_Period,
         ↪ ID_Prefer_Measured,Pref_Value_Measured) VALUES (?,?,?,?,?)");
1538
1539
1540 try {
1541
1542 stmt.setInt(1, ID_LocalSystem);
1543 stmt.setInt(2, ID_Location_Measured);
1544 stmt.setTime(3, Time_Period);
1545 stmt.setInt(4, ID_Prefer_Measured);
1546 stmt.setString(5, Pref_Value_Measured);
1547
1548 stmt.executeUpdate();
1549
1550 boolResponse = true;
1551
1552 System.out.printf("Preferences_Live_Measurement Inserted! LocalSystem %s Location %d
         ↪ TimePeriod %s Preference %d Pref_Value %s \n", ID_LocalSystem,
         ↪ ID_Location_Measured, Time_Period, ID_Prefer_Measured, Pref_Value_Measured);
1553
1554 } catch (SQLException e)
1555
1556 {
1557 e.printStackTrace();
1558 boolResponse = false;
1559 } finally {
1560
1561 if (stmt != null) {
1562 stmt.close();
1563 }
1564 if (con != null)
1565 con.close();
1566 }
1567
```

```
1568 return boolResponse;
1569 }
1570
1571 public Timestamp addDaysTotimeStamp(Timestamp timeStampToIncrement, int
         ↪ daysToIncrement) {
1572
1573 Calendar cal = Calendar.getInstance();
1574 cal.setTimeInMillis(timeStampToIncrement.getTime());
1575 cal.add(Calendar.DAY_OF_MONTH, daysToIncrement);
1576 timeStampToIncrement = new Timestamp(cal.getTime().getTime());
1577
1578 return timeStampToIncrement;
1579 }
1580
1581 public Boolean insertWeekHistory(Connection con, int idUser, int idLocalSystem,
         ↪ Timestamp TimeToInsert, Time TimePeriod, int ID_Prefs_FK, String
         ↪ changedPrefValue) throws SQLException {
1582
1583
1584 Boolean boolResponse = null;
1585
1586 System.out.printf("idUser %d LocalSystemID %d TimePeriod %s Timestamp %s Received to
         ↪ create HistoryPrefUserWithIdUserAndTime! \n", idUser, idLocalSystem, TimePeriod
         ↪ , TimeToInsert);
1587
1588 PreparedStatement stmt = con.prepareStatement("INSERT INTO History_Pref_User (
         ↪ ID_User_FK, ID_System_FK, Time, Time_Period, ID_Prefs_FK, Preference_Value)
         ↪ VALUES (?,?,?,?,?,?)");
1589
1590 int nrDays = 7;
1591
1592 for (int i = 0; i <= 364; i += nrDays) {
1593 stmt.setInt(1, idUser);
1594 stmt.setInt(2, idLocalSystem);
1595 stmt.setTimestamp(3, addDaysTotimeStamp(TimeToInsert, i));
1596 stmt.setTime(4, TimePeriod);
1597 stmt.setInt(5, ID_Prefs_FK);
1598 stmt.setString(6, changedPrefValue);
1599
1600 stmt.executeUpdate();
1601
1602 }
1603 return boolResponse;
1604 }
1605 }
```

E.2 AccessManager.java

```
1
2  package com.wsphd.model;
3
4
5  import java.sql.Connection;
6
7  import java.sql.Time;
8  import java.sql.Timestamp;
9  import java.util.ArrayList;
10
11
12
13 import com.wsphd.dao.Access;
14 import com.wsphd.dao.Database;
15 import com.wsphd.dto.History_Pref_LocalSystem;
16 import com.wsphd.dto.History_Pref_User;
17 import com.wsphd.dto.LocalSystem;
18 import com.wsphd.dto.Local_System_Present_Preferences;
19 import com.wsphd.dto.Preferences_Card;
20 import com.wsphd.dto.User;
21 import com.wsphd.dto.Utilization_Preferences;
22
23 public class AccessManager {
24
25
26
27 public ArrayList<User> getUsers() throws Exception
28 {
29 ArrayList<User> userList;
30 Database db = new Database();
31 Connection con = db.getConnection();
32 Access access = new Access();
33 userList = access.getUsers(con);
34 return userList;
35
36 }
37
38 public ArrayList<LocalSystem> getLocalSystems() throws Exception
39 {
40 ArrayList<LocalSystem> localSystemsList;
41 Database db = new Database();
42 Connection con = db.getConnection();
43 Access access = new Access();
```

```
44 localSystemsList = access.getLocalSystems(con);
45 return localSystemsList;
46 }
47
48 public Boolean insertLocalSystem(int idLocal_System, String Desc_LocalSystem, String
       ↪ GPS_Latitude, String GPS_Longitude) throws Exception
49 {
50
51 Database db = new Database();
52 Connection con = db.getConnection();
53 Access access = new Access();
54
55 access.insertLocalSystem(con, idLocal_System, Desc_LocalSystem, GPS_Latitude,
       ↪ GPS_Longitude);
56
57 return true;
58 }
59
60 public ArrayList<History_Pref_LocalSystem> getHistoryLocalSystem() throws Exception
61 {
62 ArrayList<History_Pref_LocalSystem> historyLocalSystemList;
63 Database db = new Database();
64 Connection con = db.getConnection();
65 Access access = new Access();
66 historyLocalSystemList = access.getHistoryLocalSystem(con);
67 return historyLocalSystemList;
68
69 }
70
71
72
73 public ArrayList<Utilization_Preferences> getUtilization_Preferences() throws
       ↪ Exception
74 {
75 ArrayList<Utilization_Preferences> utilization_PreferencesList;
76 Database db = new Database();
77 Connection con = db.getConnection();
78 Access access = new Access();
79 utilization_PreferencesList = access.getUtilization_Preferences(con);
80 return utilization_PreferencesList;
81 }
82
83 public ArrayList<Utilization_Preferences> getUtilization_PreferencesforUUID(String
       ↪ userUUID) throws Exception
84 {
```

```
85 ArrayList<Utilization_Preferences> utilization_PreferencesList;
86 Database db = new Database();
87 Connection con = db.getConnection();
88 Access access = new Access();
89 utilization_PreferencesList = access.getUtilization_PreferencesforUUID(con, userUUID);
90 return utilization_PreferencesList;
91 }
92
93
94 public String getUserUUID(String idUser) throws Exception
95 {
96
97 String userUUID;
98
99 Database db = new Database();
100 Connection con = db.getConnection();
101 Access access = new Access();
102 userUUID = access.getUserUUID(con, idUser);
103 return userUUID;
104
105 }
106
107
108 public String validateUserUUID(String useruuidToValidate) throws Exception
109 {
110
111 String userUUID;
112
113
114 Database db = new Database();
115 Connection con = db.getConnection();
116 Access access = new Access();
117
118 userUUID = access.validateUserUUID(con, useruuidToValidate);
119
120 return userUUID;
121
122 }
123
124
125
126
127 public String createUser(String UUID) throws Exception
128 {
129
```

```
130 String userUUID;
131
132 Database db = new Database();
133 Connection con = db.getConnection();
134 Access access = new Access();
135
136 userUUID = access.createUser(con, UUID);
137
138 return userUUID;
139
140 }
141
142 public Boolean insertHistoryPrefLocalSystem(int ID_SystemFk, Time Period_Time, int
        ↪ ID_PreferencesFk, String Preference_Value) throws Exception
143 {
144
145 Database db = new Database();
146 Connection con = db.getConnection();
147 Access access = new Access();
148
149 access.insertHistoryPrefLocalSystem(con, ID_SystemFk, Period_Time, ID_PreferencesFk,
        ↪ Preference_Value);
150
151 return true;
152 }
153
154
155 public Boolean updatePrefCardforidUser(int idUser, int IDPref, String PrefValue)
        ↪ throws Exception
156 {
157
158 Boolean boolResponse;
159
160 Database db = new Database();
161 Connection con = db.getConnection();
162 Access access = new Access();
163
164 boolResponse = access.updatePrefCardforidUser(con, idUser, IDPref, PrefValue);
165
166 return boolResponse;
167
168 }
169
170 public int getIdUserforUUID(String UUID) throws Exception
171 {
```

```
172
173 int idUser;
174
175 Database db = new Database();
176 Connection con = db.getConnection();
177 Access access = new Access();
178
179 idUser = access.getIdUserforUUID(con, UUID);
180
181 return idUser;
182
183 }
184
185 public Boolean createUserPrefCardforidUser(int idUser) throws Exception
186 {
187
188 Boolean boolResponse;
189
190 Database db = new Database();
191 Connection con = db.getConnection();
192 Access access = new Access();
193
194 boolResponse = access.createUserPrefCardforidUser(con, idUser);
195
196 return boolResponse;
197
198 }
199
200 public ArrayList<Preferences_Card> getPrefCardforidUser(int idUser) throws Exception
201 {
202 ArrayList<Preferences_Card> prefCardList;
203
204 Database db = new Database();
205 Connection con = db.getConnection();
206 Access access = new Access();
207 prefCardList = access.getPrefCardforidUser(con, idUser);
208
209 return prefCardList;
210
211 }
212
213 public ArrayList<History_Pref_User> getAllHistoryPrefUsers() throws Exception
214 {
215 ArrayList<History_Pref_User> historyPrefUsersList;
216 Database db = new Database();
```

```
217 Connection con = db.getConnection();
218 Access access = new Access();
219 historyPrefUsersList = access.getAllHistoryPrefUsers(con);
220 return historyPrefUsersList;
221 }
222
223 public Boolean initializeLocal_System_Present_Preferences(int idLocalSystem) throws
        ↪ Exception
224 {
225
226 Boolean boolResponse;
227
228 Database db = new Database();
229 Connection con = db.getConnection();
230 Access access = new Access();
231
232 boolResponse = access.initializeLocal_System_Present_Preferences(con, idLocalSystem);
233
234 return boolResponse;
235
236 }
237
238 public Boolean insertHistoryForidUserAndLocalSystem(int idUser, int idLocalSystem,
        ↪ Time TimePeriod) throws Exception
239 {
240 Boolean boolresponse;
241 Database db = new Database();
242 Connection con = db.getConnection();
243 Access access = new Access();
244 boolresponse = access.insertHistoryForidUserAndLocalSystem(con, idUser, idLocalSystem,
        ↪  TimePeriod);
245 return boolresponse;
246 }
247
248 public Boolean insertHistoryPrefUserWithIdUserAndTime(int idUser, int idLocalSystem,
        ↪ Timestamp TimeToInsert, Time TimePeriod) throws Exception
249 {
250 Boolean boolresponse;
251 Database db = new Database();
252 Connection con = db.getConnection();
253 Access access = new Access();
254 boolresponse = access.insertHistoryPrefUserWithIdUserAndTime(con, idUser,
        ↪ idLocalSystem, TimeToInsert, TimePeriod);
255 return boolresponse;
256 }
```

```
257
258 public int getLastLocalSystemIDForUser(int idUser) throws Exception
259 {
260 int idLocalSystem = -1;
261
262 Database db = new Database();
263 Connection con = db.getConnection();
264 Access access = new Access();
265 idLocalSystem = access.getLastLocalSystemIDForUser(con, idUser);
266 return idLocalSystem;
267 }
268
269 public ArrayList<Local_System_Present_Preferences> getLocal_System_Present_Preferences
    ↪ (int idLocalSystem, Time TimePeriod) throws Exception
270 {
271 ArrayList<Local_System_Present_Preferences> utilization_PreferencesAtLocalSystemList;
272 Database db = new Database();
273 Connection con = db.getConnection();
274 Access access = new Access();
275 utilization_PreferencesAtLocalSystemList = access.getLocal_System_Present_Preferences(
    ↪ con, idLocalSystem, TimePeriod);
276 return utilization_PreferencesAtLocalSystemList;
277 }
278
279 public Boolean updateLocalSystemPresentPreferences(int idLocalSystem, int idUser, int
    ↪ idPreference, String PreferenceValue) throws Exception
280 {
281 Boolean boolresponse;
282 Database db = new Database();
283 Connection con = db.getConnection();
284 Access access = new Access();
285 boolresponse = access.updateLocalSystemPresentPreferences(con, idLocalSystem, idUser,
    ↪ idPreference, PreferenceValue);
286 return boolresponse;
287 }
288
289 public Boolean initializeUtilization_PreferencesForAllSystem() throws Exception
290 {
291
292 Boolean boolResponse;
293
294 Database db = new Database();
295 Connection con = db.getConnection();
296 Access access = new Access();
297
```

```
298 boolResponse = access.initializeUtilization_PreferencesForAllSystem(con);
299
300 return boolResponse;
301
302 }
303
304 public Boolean initializeUtilization_PreferencesForNewUser(int idUser) throws
        ↪ Exception
305 {
306
307 Boolean boolResponse;
308
309 Database db = new Database();
310 Connection con = db.getConnection();
311 Access access = new Access();
312
313 boolResponse = access.initializeUtilization_PreferencesForNewUser(con, idUser);
314
315 return boolResponse;
316
317 }
318
319 public Boolean insertPreference_Live_Measurement(int ID_LocalSystem, int
        ↪ ID_Location_Measured, Time Time_Period, int ID_Prefer_Measured, String
        ↪ Pref_Value_Measured) throws Exception
320 {
321 Boolean boolResponse;
322
323 Database db = new Database();
324 Connection con = db.getConnection();
325 Access access = new Access();
326
327 boolResponse = access.insertPreference_Live_Measurement(con, ID_LocalSystem,
        ↪ ID_Location_Measured, Time_Period, ID_Prefer_Measured, Pref_Value_Measured);
328
329 return boolResponse;
330 }
331
332 }
```

E.3 LocalSystemWS.java

```
1
2  package com.wsphd.webservices;
3
4
5  import java.sql.Time;
6  import java.time.LocalTime;
7  import java.util.ArrayList;
8
9  import javax.annotation.security.RolesAllowed;
10 import javax.ws.rs.GET;
11 import javax.ws.rs.POST;
12 import javax.ws.rs.Path;
13 import javax.ws.rs.PathParam;
14 import javax.ws.rs.Produces;
15
16 import com.google.gson.Gson;
17 import com.wsphd.dto.History_Pref_LocalSystem;
18 import com.wsphd.dto.LocalSystem;
19 import com.wsphd.dto.Local_System_Present_Preferences;
20 import com.wsphd.dto.Utilization_Preferences;
21 import com.wsphd.model.AccessManager;
22
23
24 @Path("/LocalSystemWS")
25
26 public class LocalSystemWS {
27
28 @GET
29 @Path("/getLocalSystems")
30 @Produces("application/json")
31
32 public String GetaLocalSystems()
33 {
34
35 String localSystemsList = null;
36
37 ArrayList<LocalSystem> LocalSystemsList;
38 try
39 {
40 LocalSystemsList = new AccessManager().getLocalSystems();
41 Gson gson = new Gson();
42 localSystemsList = gson.toJson(LocalSystemsList);
43 } catch (Exception e)
```

```
44
45 {
46 e.printStackTrace();
47 }
48 return localSystemsList;
49 }
50
51 @POST
52 @Path("/insertLocalSystem/{idLocal_System}/{Desc_LocalSystem}/{GPS_Latitude}/{
       ↪ GPS_Longitude}")
53 @Produces("application/json")
54
55 public Boolean insertLocalSystem(@PathParam("idLocal_System") int idLocal_System,
       ↪ @PathParam("Desc_LocalSystem") String Desc_LocalSystem, @PathParam("
       ↪ GPS_Latitude") String GPS_Latitude, @PathParam("GPS_Longitude") String
       ↪ GPS_Longitude)
56
57 {
58
59 Boolean boolResponse;
60 try
61 {
62 new AccessManager().insertLocalSystem(idLocal_System, Desc_LocalSystem, GPS_Latitude,
       ↪ GPS_Longitude);
63 boolResponse= true;
64 }
65
66 catch (Exception e)
67
68 {
69 e.printStackTrace();
70 boolResponse= false;
71 }
72
73
74 if (boolResponse)
75 {
76 try
77 {
78 new AccessManager().initializeLocal_System_Present_Preferences(idLocal_System);
79
80 boolResponse = true;
81
82 } catch (Exception e)
83
```

```
84  {
85  e.printStackTrace();
86  boolResponse = false;
87  }
88
89  }
90
91  else
92  {
93  boolResponse = false;
94  }
95  return boolResponse;
96  }
97
98
99  @GET
100 @Path("/getHistoryLocalSystems")
101 @Produces("application/json")
102
103 public String historyLocalSystems()
104 {
105 String historyLocalSystems = null;
106
107 ArrayList<History_Pref_LocalSystem> historyLocalSystemsList;
108 try
109 {
110 historyLocalSystemsList = new AccessManager().getHistoryLocalSystem();
111 Gson gson = new Gson();
112 historyLocalSystems = gson.toJson(historyLocalSystemsList);
113 } catch (Exception e)
114
115 {
116 e.printStackTrace();
117 }
118 return historyLocalSystems;
119 }
120
121 @POST
122 @Path("/insertHistoryPrefLocalSystem")
123 @Produces("application/json")
124
125
126 public Boolean insertHistoryPrefLocalSystem(int ID_SystemFk, Time Time_Period, int
        ↪ ID_PreferencesFk, String Preference_Value)
127
```

```
128 {
129
130
131 try
132 {
133 new AccessManager().insertHistoryPrefLocalSystem(ID_SystemFk, Time_Period,
        ↪ ID_PreferencesFk, Preference_Value);
134
135 } catch (Exception e)
136
137 {
138 e.printStackTrace();
139 }
140 return true;
141 }
142
143 @POST
144 @Path("/getUtilization_PreferencesforUUID")
145 @Produces("application/json")
146
147 public String Utilization_PreferencesforUUID(String userUUID)
148 {
149 String utilization_PreferencesforUUID = null;
150
151 ArrayList<Utilization_Preferences> utilization_PreferencesList;
152 try
153 {
154 utilization_PreferencesList = new AccessManager().getUtilization_PreferencesforUUID(
        ↪ userUUID);
155 Gson gson = new Gson();
156 utilization_PreferencesforUUID = gson.toJson(utilization_PreferencesList);
157 } catch (Exception e)
158
159 {
160 e.printStackTrace();
161 }
162 return utilization_PreferencesforUUID;
163 }
164
165 @GET
166 @Path("/getUtilization_Preferences")
167 @Produces("application/json")
168
169 public String Utilization_Preferences()
170 {
```

```
171 String utilization_Preferences = null;
172
173 ArrayList<Utilization_Preferences> utilization_PreferencesList;
174 try
175 {
176 utilization_PreferencesList = new AccessManager().getUtilization_Preferences();
177 Gson gson = new Gson();
178 utilization_Preferences = gson.toJson(utilization_PreferencesList);
179 } catch (Exception e)
180
181 {
182 e.printStackTrace();
183 }
184 return utilization_Preferences;
185 }
186
187 @POST
188 @Path("/getLocal_System_Present_Preferences/{UUID}")
189 @Produces("application/json")
190
191 public ArrayList<Local_System_Present_Preferences> getLocal_System_Present_Preferences
    ↪ (@PathParam("UUID") String UUID)
192 {
193
194
195 ArrayList<Local_System_Present_Preferences> utilization_PreferencesAtLocalSystemList =
    ↪  new ArrayList<>();
196
197 int idUser =-1;
198 int idLocalSystem = -1;
199
200 try
201 {
202 idUser = new AccessManager().getIdUserforUUID(UUID);
203 }
204
205 catch (Exception e)
206
207 {
208 e.printStackTrace();
209 idUser=-1;
210 }
211
212 if (idUser!=-1)
213 {
```

```
214 try
215 {
216 idLocalSystem = new AccessManager().getLastLocalSystemIDForUser(idUser);
217
218 }
219 catch (Exception e)
220 {
221 e.printStackTrace();
222 idLocalSystem = -1;
223 }
224 }
225 else
226 {
227 utilization_PreferencesAtLocalSystemList = null;
228 }
229 if (idLocalSystem!=-1)
230 {
231
232 try
233 {
234 LocalTime hourToGetPeriod = LocalTime.now();
235
236 Time timePeriod = Local_System_Present_Preferences.getTimePeriod(hourToGetPeriod);
237
238 System.out.println(timePeriod);
239
240 utilization_PreferencesAtLocalSystemList = new AccessManager().
    ↪ getLocal_System_Present_Preferences(idLocalSystem, timePeriod);
241
242 } catch (Exception e)
243
244 {
245 e.printStackTrace();
246 }
247 }
248 else
249 {
250 utilization_PreferencesAtLocalSystemList = null;
251 }
252
253 return utilization_PreferencesAtLocalSystemList;
254 }
255
256
257 @POST
```

```
258 @Path("/updateLocalSystemPresentPreferences/{UUID}/{idPreference}/{PreferenceValue}")
259 @Produces("application/json")
260
261 public Boolean updateLocalSystemPresentPreferences(@PathParam("UUID") String UUID,
        ↪ @PathParam("idPreference") int idPreference, @PathParam("PreferenceValue")
        ↪ String PreferenceValue)
262
263 {
264 int idUser =-1;
265 int idLocalSystem = -1;
266
267 Boolean boolResponse;
268 try
269 {
270 idUser = new AccessManager().getIdUserforUUID(UUID);
271 }
272
273 catch (Exception e)
274
275 {
276 e.printStackTrace();
277 idUser=-1;
278 }
279
280 if (idUser!=-1)
281 {
282 try
283 {
284 idLocalSystem = new AccessManager().getLastLocalSystemIDForUser(idUser);
285
286 }
287 catch (Exception e)
288 {
289 e.printStackTrace();
290 idLocalSystem = -1;
291 }
292 }
293 else
294 {
295 boolResponse = false;
296 }
297 if (idLocalSystem!=-1)
298 {
299 try
300 {
```

```
301
302 boolResponse = new AccessManager().updateLocalSystemPresentPreferences(idLocalSystem,
        ↪ idUser, idPreference, PreferenceValue);
303
304 }
305 catch (Exception e)
306 {
307 e.printStackTrace();
308 boolResponse = false;
309 }
310 }
311 else
312 {
313 boolResponse = false;
314 }
315 return boolResponse;
316 }
317
318
319
320 @POST
321 @Path("/RegisterUserAtLocalSystem/{UUID}/{idLocalSystem}")
322 @Produces("application/json")
323
324 public Boolean RegisterUserAtLocalSystem(@PathParam("UUID") String UUID,@PathParam("
        ↪ idLocalSystem") int idLocalSystem)
325 {
326
327 String UUID_Validated = "-1";
328
329 int idUser=-1;
330
331 Boolean boolResponse;
332 try
333 {
334 UUID_Validated = new AccessManager().validateUserUUID(UUID);
335 }
336
337 catch (Exception e)
338
339 {
340 e.printStackTrace();
341 UUID_Validated="-1";
342 }
343
```

```
344 if (!UUID_Validated.equals("-1"))
345 {
346
347 try
348 {
349 idUser = new AccessManager().getIdUserforUUID(UUID);
350
351 } catch (Exception e)
352
353 {
354 e.printStackTrace();
355 idUser=-1;
356 }
357 }
358 else
359 {
360 boolResponse=false;
361
362 }
363
364 if (idUser!=-1)
365 {
366 try
367 {
368 LocalTime hourToGetPeriod = LocalTime.now();
369
370 Time timePeriod = Local_System_Present_Preferences.getTimePeriod(hourToGetPeriod);
371
372 System.out.println(timePeriod);
373
374 new AccessManager().insertHistoryForidUserAndLocalSystem(idUser, idLocalSystem,
        ↪ timePeriod);
375
376 boolResponse =true;
377
378 } catch (Exception e)
379
380 {
381 e.printStackTrace();
382 boolResponse=false;
383 }
384
385 }
386
387 else
```

```
388 {
389 boolResponse= false;
390 }
391 return boolResponse;
392 }
393 
394 
395 @RolesAllowed("member")
396 @POST
397 @Path("/insertPreference_Live_Measurement/{ID_LocalSystem}/{ID_Location_Measured}/{
        ↪ ID_Prefer_Measured}/{Pref_Value_Measured}")
398 @Produces("application/json")
399 
400 public Boolean insertPreference_Live_Measurement(@PathParam("ID_LocalSystem") int
        ↪ ID_LocalSystem, @PathParam("ID_Location_Measured") int ID_Location_Measured,
        ↪ @PathParam("ID_Prefer_Measured") int ID_Prefer_Measured, @PathParam("
        ↪ Pref_Value_Measured") String Pref_Value_Measured)
401 
402 {
403 Boolean boolResponse;
404 
405 try
406 {
407 LocalTime hourToGetPeriod = LocalTime.now();
408 
409 Time Time_Period = Local_System_Present_Preferences.getTimePeriod(hourToGetPeriod);
410 
411 boolResponse = new AccessManager().insertPreference_Live_Measurement(ID_LocalSystem,
        ↪ ID_Location_Measured, Time_Period, ID_Prefer_Measured, Pref_Value_Measured);
412 
413 }
414 catch (Exception e)
415 {
416 e.printStackTrace();
417 boolResponse = false;
418 }
419 
420 return boolResponse;
421 }
422 }
```

E.4 PreferencesCardWS.java

```
1
2  package com.wsphd.webservices;
3
4
5
6  import java.util.ArrayList;
7
8  import javax.ws.rs.POST;
9  import javax.ws.rs.Path;
10 import javax.ws.rs.Produces;
11
12 import com.wsphd.dto.Preferences_Card;
13 import com.wsphd.model.AccessManager;
14
15
16
17 import javax.ws.rs.PathParam;
18
19 @Path("/preferencesCardWS")
20
21 public class PreferencesCardWS {
22
23 @POST
24 @Path("/getPrefCardforUUID/{UUID}")
25 @Produces("application/json")
26
27 public ArrayList<Preferences_Card> getPrefCardforidUser(@PathParam("UUID") String UUID
       ↪ )
28 {
29
30 System.out.printf("UUID Received to Get Pref. Card: %s \n", UUID);
31
32 int idUser;
33
34 ArrayList<Preferences_Card> prefCardList;
35
36 try
37 {
38 idUser = new AccessManager().getIdUserforUUID(UUID);
39
40 } catch (Exception e)
41
42 {
```

```
43 e.printStackTrace();
44 idUser=0;
45 }
46
47 if (idUser!=0)
48 {
49 try
50 {
51 prefCardList = new AccessManager().getPrefCardforidUser(idUser);
52
53
54 } catch (Exception e)
55
56 {
57 e.printStackTrace();
58 prefCardList=null;
59 }
60
61 }
62
63 else
64 {
65 prefCardList=null;
66 }
67 return prefCardList;
68 }
69
70
71 @POST
72
73
74 @Path("/updatePrefCardforUUID/{UUID}/{IDPref}/{PrefValue}")
75
76 @Produces("application/json")
77 public Boolean updatePrefCardforUUID(@PathParam("UUID") String UUID, @PathParam("
    ↪ IDPref") int IDPref, @PathParam("PrefValue") String PrefValue)
78
79 {
80
81 Boolean boolResponse = null;
82 System.out.printf("UUID Received to Update Pref. Card: %s \n", UUID);
83 System.out.printf("IDPref Received to Update Pref. Card: %d \n", IDPref);
84 System.out.printf("PrefValue Received to Update Pref. Card: %s \n", PrefValue);
85
86 int idUser=0;
```

```
87
88  try
89  {
90  idUser = new AccessManager().getIdUserforUUID(UUID);
91
92  } catch (Exception e)
93
94  {
95  e.printStackTrace();
96  idUser=0;
97  }
98
99  if (idUser!=0)
100 {
101 try
102 {
103 boolResponse = new AccessManager().updatePrefCardforidUser(idUser, IDPref, PrefValue);
104
105 boolResponse = true;
106
107 } catch (Exception e)
108
109 {
110 e.printStackTrace();
111 boolResponse=false;
112 }
113
114 }
115
116 else
117 {
118 boolResponse=false;
119 }
120 return boolResponse;
121 }
122 }
```

E.5 UserWS.java

```
1
2  package com.wsphd.webservices;
3
4  import java.sql.Time;
5  import java.sql.Timestamp;
6  import java.text.SimpleDateFormat;
7  import java.time.LocalTime;
8  import java.util.ArrayList;
9
10 import javax.annotation.security.RolesAllowed;
11 import javax.ws.rs.GET;
12 import javax.ws.rs.POST;
13 import javax.ws.rs.Path;
14 import javax.ws.rs.PathParam;
15 import javax.ws.rs.Produces;
16
17 import com.google.gson.Gson;
18 import com.wsphd.dto.History_Pref_User;
19 import com.wsphd.dto.Local_System_Present_Preferences;
20 import com.wsphd.dto.User;
21
22 import com.wsphd.model.AccessManager;
23
24
25 @Path("/userWS")
26
27 public class UserWS {
28
29 @RolesAllowed("member")
30 @GET
31 @Path("/users")
32 @Produces("application/json")
33
34 public String users()
35 {
36 String users = null;
37 ArrayList<User> userList;
38 try
39 {
40 userList = new AccessManager().getUsers();
41 Gson gson = new Gson();
42 users = gson.toJson(userList);
43 } catch (Exception e)
```

```
44
45 {
46 e.printStackTrace();
47 }
48 return users;
49 }
50
51 @POST
52 @Path("/getUserUUID/{idUser}")
53
54 @Produces("application/json")
55
56 public String userUUID(@PathParam("idUser") String idUser)
57 {
58 String userUUID = null;
59
60 try
61 {
62 userUUID = new AccessManager().getUserUUID(idUser);
63 Gson gson = new Gson();
64 userUUID = gson.toJson(userUUID);
65 } catch (Exception e)
66
67 {
68 e.printStackTrace();
69 }
70 return userUUID;
71 }
72
73
74 @POST
75 @Path("/validateUserUUID/{UUID}")
76 @Produces("application/json")
77
78 public String validateUserUUID(@PathParam("UUID") String UUID)
79 {
80
81 String userUUID;
82
83 try
84 {
85 userUUID = new AccessManager().validateUserUUID(UUID);
86
87 } catch (Exception e)
88
```

```
89 {
90 e.printStackTrace();
91
92 return null;
93 }
94 return userUUID;
95
96 }
97
98 @POST
99 @Path("/createUser/{UUID}")
100 @Produces("application/json")
101
102 public String createUser(@PathParam("UUID") String UUID)
103 {
104 String userUUID;
105
106 try
107 {
108 userUUID = new AccessManager().createUser(UUID);
109
110
111 } catch (Exception e)
112
113 {
114 e.printStackTrace();
115 return null;
116 }
117
118 int idUser;
119
120 try
121 {
122 idUser = new AccessManager().getIdUserforUUID(UUID);
123
124 } catch (Exception e)
125
126 {
127 e.printStackTrace();
128 idUser=0;
129 }
130
131 if (idUser!=0)
132 {
133 try
```

```
134 {
135 new AccessManager().createUserPrefCardforidUser(idUser);
136
137
138 } catch (Exception e)
139
140 {
141 e.printStackTrace();
142 return null;
143 }
144 }
145
146 else
147 {
148 return null;
149 }
150 return userUUID;
151 }
152
153 @POST
154 @Path("/insertHistoryPrefUser/{UUID}/{idLocal_System}")
155 @Produces("application/json")
156
157 public Boolean insertHistoryPrefUser(@PathParam("UUID") String UUID, @PathParam("
        ↪ idLocal_System") int idLocal_System)
158 {
159
160 Boolean boolResponse;
161
162 int idUser;
163
164 try
165 {
166 idUser = new AccessManager().getIdUserforUUID(UUID);
167
168 } catch (Exception e)
169
170 {
171 e.printStackTrace();
172 idUser=-1;
173 }
174
175 if (idUser!=-1)
176 {
177 try
```

```
178 {
179 LocalTime hourToGetPeriod = LocalTime.now();
180
181 Time timePeriod = Local_System_Present_Preferences.getTimePeriod(hourToGetPeriod);
182
183 System.out.println(timePeriod);
184
185 new AccessManager().insertHistoryForidUserAndLocalSystem(idUser, idLocal_System,
        ↪ timePeriod);
186
187 boolResponse =true;
188
189 } catch (Exception e)
190
191 {
192 e.printStackTrace();
193 boolResponse=false;
194 }
195 }
196
197 else
198 {
199 boolResponse= false;
200 }
201 return boolResponse;
202 }
203
204 @POST
205 @Path("/insertHistoryPrefUserWithIdUserAndTime/{idUser}/{idLocal_System}/{TimeToInsert
        ↪ }")
206 @Produces("application/json")
207
208 public Boolean insertHistoryPrefUserWithIdUserAndTime(@PathParam("idUser") int idUser,
        ↪  @PathParam("idLocal_System") int idLocal_System, @PathParam("TimeToInsert")
        ↪ Timestamp TimeToInsert)
209 {
210
211 Boolean boolResponse;
212
213 try
214 {
215
216 SimpleDateFormat timeFormat = new SimpleDateFormat("HH:mm:ss");
217 String StringTimeToInsert = timeFormat.format(TimeToInsert);
218 System.out.println(StringTimeToInsert);
```

```
219 LocalTime hourToGetPeriod = LocalTime.parse(StringTimeToInsert);
220 Time timePeriod = Local_System_Present_Preferences.getTimePeriod(hourToGetPeriod);
221
222 System.out.println(timePeriod);
223
224 new AccessManager().insertHistoryPrefUserWithIdUserAndTime(idUser, idLocal_System,
        ↪ TimeToInsert, timePeriod);
225
226 boolResponse =true;
227
228 } catch (Exception e)
229
230 {
231 e.printStackTrace();
232 boolResponse=false;
233 }
234 return boolResponse;
235 }
236
237 @GET
238 @Path("/getAllHistoryPrefUsers")
239 @Produces("application/json")
240
241 public String getAllHistoryPrefUsers()
242 {
243 String historyPrefUsers = null;
244
245 ArrayList<History_Pref_User> history_Pref_User_List;
246 try
247 {
248 history_Pref_User_List = new AccessManager().getAllHistoryPrefUsers();
249 Gson gson = new Gson();
250 historyPrefUsers = gson.toJson(history_Pref_User_List);
251 } catch (Exception e)
252
253 {
254 e.printStackTrace();
255 }
256 return historyPrefUsers;
257 }
258 }
```