INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

UNIVERSITY OF WOLLONGONG AUSTRALIA

Innovative Project

Context Aware IoT System

Damaris, Mendes Ferreira, mendesfe@etud.insa-toulouse.fr

Dorian, Moulinié, moulinie@etud.insa-toulouse.fr

Julien, Chouvet, chouvet.julien@gmail.com

Juan, Yu, j_yu@etud.insa-toulouse.fr

2018 - 2019

Tutors: Nicolas Seydoux and Nicolas Verstaevel

Abstract

The emerging sector of the new technologies, the Internet of Things (IoT), is bringing a completely new way to see our world by adding two adjectives to our daily life: connected and smart. Our project follows this philosophy. The goal is to incorporate these technologies in a university setting in order to build an intelligent digital campus to increase the efficiency and improve experiences for students and staff.

The system is based on an application capable of driving productivity with location-based intelligence on a smart-campus. We used a hardware element composed of a LoPy and a PyTrack to collect the Wi-Fi and Bluetooth signal strength information at a specific location on the campus. We compare this with the Wi-Fi and Bluetooth information collected by the smartphone application. Through an analysis of these signal strengths, we locate students and staff on campus. Thus, we can make data-driven decisions to create relevant experiences and services on the university campus. Moreover, in the future we can use the application to offer IoT services on campus by connecting students to multiple smart devices. This project is a first step in creating a smart-campus that can provide context-aware services in a fully integrated intelligent university.

Summary

List of Abbreviations

BSSID	Basic Service Set Identifier
RSSI	Received Signal Strength Indication
GPS/GNSS	Global Positioning System
IoT	Internet of things
LPWAN	Low-power wide-area network
LAN	Local Area Network
RF	Radio Frequency
MAC	Medium Access Control
AP	Access Point
dBm	decibels/milliwatt
IDE	Integrated Development Environment
UI	User Interface
JSON	JavaScript Object Notation
HTTP	Hypertext Transfer Protocol
KB	Knowledge base
RDF	Resource Description Framework
SOSA	Sensor, Observation, Sample, and Actuator

1. Introduction

Providing better student experiences is one of the main challenges in any educational institution. Building an intelligent digital campus can make a school or university more attractive and, moreover, can improve experiences for students and staff. The idea of a smart campus is to offer context-aware services such as, for example, facilities management and personalized experiences. Using location intelligence is a way to drive productivity creating relevant experiences and optimizing services while on campus.

The main idea of this project is to study a new way to locate users on the campus and secure access to data and applications by using device connectivity as a context. As the students and staff move around the campus with their smartphones we can determine user location by measuring the received signal strength (RSS) of all the networks nearby. In doing so, we can optimize services by the bias of a smartphone application by linking these services to the user and their location.

The most common means to determine geolocation today is through Global Navigation Satellite System (GPS or GNSS) which can allow a smart-campus system to locate users on campus. The problem is that usually students and staff are inside buildings, in a classroom for example, which require an indoor positioning system and GPS is not suitable for establishing indoor locations. As this technology is based on satellite signals, the microwaves that are sent to the smartphone are attenuated and scattered by roofs, walls and other objects. The resulting location is typically not accurate enough to use in an indoor location system.

When a device or a user is at a specific location, its connectivity can be used as a specific footprint. Whenever a device is in a specific place on the smart campus, the connectivity footprint is extracted and compared to the whole set of rooms footprints on the campus to localise the user. If the user's footprint matches with a room's footprint, we can offer access to specific room services through an application. We can also think of future applications by comparing footprints between users, so when two user devices share a similar footprint, they can exchange information for example.

For instance, when someone enters the university restaurant with our application installed on their smartphone, he will receive the day's menu. Or, when someone is listening to a presentation in a room, he will have access to information about the presentation. As soon as the device leaves the room, access to the room's information will be denied but new information will be displayed depending on a new footprint.

To implement such a concept, we developed an electronic device to collect the connectivity footprints on campus. These footprints will be stored on a server to be compared with user footprints all around the university campus. We also developed an Android smartphone application as a way to interact with users and offer context-aware services. We used Wi-Fi and Bluetooth technologies in this project as they are widely used all over the world, but this concept could be extended to any other signal-based technology.

2. Organization

2.1 Team

We are four students in INSA 5ISS who are learning about IoT. The table below explains a little bit about each of the group members:

Members	Juan	Damaris	Dorian	Julien
Orientation	IR	IR	IR	AE
Strengths	Web Semantics	Programming and algorithm development	Web Tech Technology intelligence	Electronics Technology intelligence
Weakness	Electronics	Electronics	Electronics	Software
Previous work	- Web site - Web server	- Mobile Game App - Development of an JavaScript and C++ application	- Web App Mobile - API Rest	- Arduino + BLE + Android - Connected sensors

We divided the workload so that every team member had a main area of focus, while keeping up with the general overview of the project. To divide the work, we considered the strengths, weaknesses, previous work and preferences of each member.

This way we managed to progress on the project fairly quickly once the difficulties with the project definitions and specifications were overcome. However, the most laborious part of the project ended up being the final tests and adjustments.

The coordination of the work was reinforced through weekly meetings of our team and occasional meetings with the tutor to have feedback on our progress. Communication between team members was maintained through the meetings and a dedicated message channel. Everyone actively interacted and asked questions if needed. It was also beneficial to have the team members in leading roles in their respective areas, which meant that there was no ambiguity in the decisions made.

2.2 Planning and methodology

Initially we made plans to schedule the workflow of the project by using a Gantt chart for exemple, nevertheless, we promptly moved on to a more Agile style of development. The focus of this approach was to prioritise the most essential components and iterate from there towards the final vision of the project. The team adapted better to this methodology because

trying to stick to the timeline of a Gantt chart is very difficult, mainly when the Agile lifecycle is integrated in the project. When we plan the entire project at once in the beginning, we can not predict all the difficulties that we are going to be faced to, that is why a weekly planning was a better solution to our organization.

We used Trello[5] to define the main tasks each month to know what we needed to accomplish in order to finish our work on time. Figure 1 shows the schedule we first established.

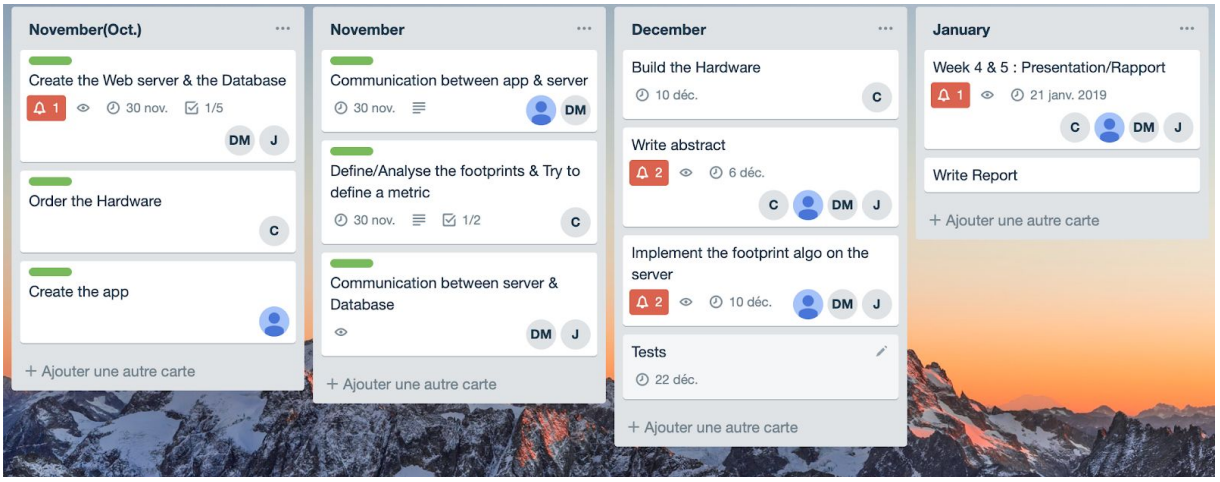


Figure 1. Trello planning.

Due to delays in the hardware delivery and Christmas vacations in December, we postponed some tasks until the beginning of the year. We left the January schedule free in the case of delays to make final tests and adjustments.

Weekly deadlines were set to divide the workload evenly relative to the timeline and fairly to the team members. For all meetings, we wrote a debriefing so we would have everything documented in a Google Drive (file storage and synchronization service on Cloud) that was shared with the team and the tutors. Another reason for this kind of documentation is that we worked remotely with the University of Wollongong from Australia. That made project supervision easier for both sides.

To implement an Agile iterative approach to plan and guide our project, we chose to work with the Scrum framework that is based on sprints. Every week we had a group meeting to keep our planning up to date and to discuss the work that we had done so far. Figure 2 shows how the Scrum methodology works .

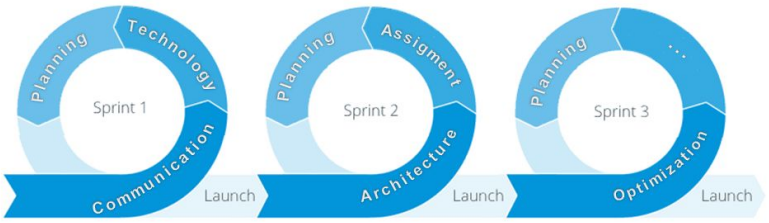


Figure 2. Scrum framework.

3. Expected Results

A live demonstration with a minimum viable product showing that the user gain access to specific services when entering the room with the smartphone. We plan to do that using all available networks in the room as connection information (Bluetooth and Wifi) as a way to localize the user. We expect to have a hardware which is able to collect the connectivity footprints and GPS coordinates to study the relationship between those data.

As this study is mainly a proof of concept, we are going to perform several tests with different algorithms to make the footprint calculus and different methods to collect the footprint. Moreover, we want to compare the efficiency of our system with the GPS. All this studies are going to be documented and the results are going to be available with free licence. All this data are going to be available on a GitHub repository[9] until the day of the final presentation of the project (25 January 2018). The repository will be divided in 5 main parts:

- Smartphone with the applications project.
- Hardware with the code used to collect the footprint.
- Server with the algorithms used for footprint analysis and comparison.
- Knowledge base with the source to our ontology.
- Studies with all the documentation of the project.

4. Materials and Methods

4.1 Footprints

The localisation system we developed is based on what we call « footprints ». In a given time, at a given place, a device (a smartphone for example) has a certain footprint. Based on this footprint and on a map of footprints, we are able to localise a device.

In this section, we will see what this footprint is made of and how we calculate it for two networks that we choose: Wi-Fi and Bluetooth. It is not limited to these signals in particular and can be extended to any kind of wireless signal as LPWAN (LoRa, Sigfox, etc) for example. We choose to focus in this two technologies because they are massively present in our testing environment.

4.1.1 Wireless Signals

Bluetooth and Wi-Fi are both methods that provide wireless communication. Bluetooth is a wireless technology standard that is used to exchange data over short distances (less than 10m), usually between personal mobile devices. Wi-Fi is also a wireless standard, but rather than being designed to communicate between devices, it serves mainly to wirelessly connect devices to the internet or Ethernet networks such as a corporate Local Area Network (LAN). Its range is farther than Bluetooth, as a Wi-Fi signal can be accessed up to 300 feet away. This means that a Wi-Fi-enabled device, such as a PC or smartphone, connects to the internet wirelessly when in a Wi-Fi "hotspot," or area in which a Wi-Fi signal may be accessed.

Both of these two technologies use radio waves to exchange data and operate in the Radio Frequency (RF) spectrum. Wi-Fi and Bluetooth networks can be characterized by name, frequency, and channel, but in this work we are more interested by the BSSID (Basic Service Set Identifier) and RSSI (Received Signal Strength Indicator) of these wireless networks.

The BSSID is the MAC (Medium Access Control) address of the AP (Access Point) and in Independent BSS or ad hoc networks. It is used to identify the source of the signal. The RSSI is the *Received Signal Strength Indicator*, and is useful for determining if the source of the signal is close or not. The easiest and most consistent way to express signal strength is with dBm, which stands for decibels relative to a milliwatt. Since RSSI is handled differently by most WiFi adapters, it is usually converted to dBm to make it consistent and human-readable. In figure 3, we can see the meaning of the RSSI in terms of signal strength. The RSSI value usually varies from -30dBm to -100dBm. When the RSSI tends to a higher value (close to

-30dBm), the signal strength is excellent. When the RSSI tends to a lower value (less than -70dBm), the signal strength is poor.

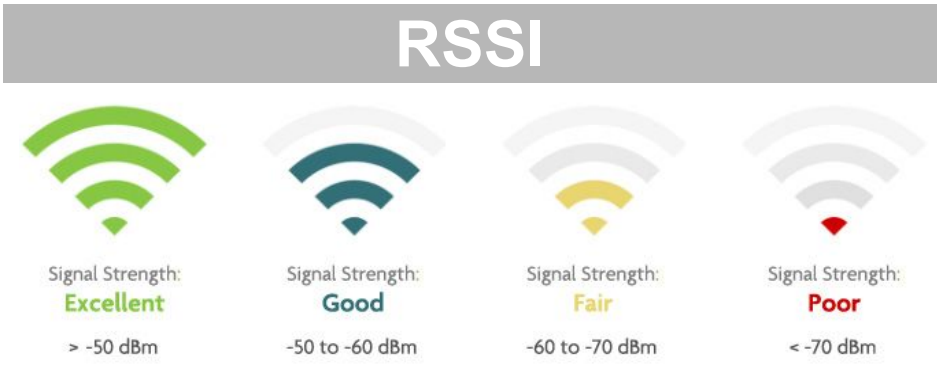


Figure 3. Received Signal Strength Indicator[8].

4.1.2 Footprint for the Wi-Fi

Nowadays, there are a lot of different Wi-Fi networks in buildings or public places. Therefore, using this technology might be the best solution because it can be deployed in almost every place where there are Wi-Fi access points.

Wi-Fi is a long-range network and can pass through walls fairly easily compared to other technologies like Bluetooth for example. However, in a building with a lot of Wi-Fi access points, such as the one we used for our experiments, the large number of networks causes interferences between the signals.

Thus, what we first notice is that, for the same room in the same building, if we look at the Wi-Fi networks and their intensity in different places in the room, the variation in the networks received and their intensity do not depend on the distance between these points and that there is no relation between the difference of RSSI between 2 points and the distance between these points. (You can find a study showing these observations in the folder "*Wifi_Signal_Analysis/RSSI_and_Distance*" of the Git repository of the project). This is due to the interferences between the signals and the absorption and reflexions that occur with the physical environment (walls, chairs, tables, human bodies, etc). Therefore, we cannot predict the relative distance between 2 points by only knowing the networks, and their respective intensity, they receive.

This is why we decided to implement a method that is based on footprints. The idea is simple:

- First, we have to calculate the footprint for each room of the building in which we want to implement the localisation system. In this way, we obtain a map of the building.
- Then, when a user with their smartphone is in a room, we calculate its footprint and we compare it to the footprints from our previously established map. By determining the footprint which most closely matches the user's footprint, we can either localise the user or know which other devices (sensors, actuators) he is closed to, in order to give him access to their services.

Hereafter we present how footprints are generated. They are determined differently for the mapping of the building and for the user.

4.1.2.1 Room footprints

A) Static and dynamic methods

There is two different methods to determine the footprint of the rooms: a static method and a dynamic method.

The main difference between them is that for the static method we use only one hardware to collect, only once, in each room the available networks, and their respective RSSI. Thus, we obtain a static map of the building. However, for the dynamic mapping, we need as much hardware as rooms we want to map because we put one hardware by room. These hardware are going to send periodically the networks they received and so create a dynamic map of the building.

Both of these methods have their own advantages, like for example:

- Static method:
  - Less expensive because we need only one hardware.
  - Need less resources for the server because with the dynamic method, each hardware sends periodically a request, containing all the received networks, to the server. Thus, with a big number of hardware the server can be saturated.
- Dynamic method:
  - Take into account the dynamism of the networks signals.
  - Is able to adapt the footprints of the rooms if case an access point is removed or added.

B) Observations

Before creating an algorithm to determine the footprints, we made several tests to study the characteristics of the available Wi-Fi signals on our testing environment. We notices two things:

1. There are some Wi-Fi access points we collect that are temporarily present. They correspond, for example, to mobile phones which are used as Wi-Fi access points. For the static method, these temporary access points are parasites for our mapping because if our footprints map takes them into account, once they disappear, the mapping is no longer valid. To avoid this phenomenon, to create our original mapping, we decided to perform scans, to collect the Wi-Fi networks available in a room, at night, in order to avoid collecting parasite Wi-Fi access point signals. However, for the dynamic method, these temporary access points are important because if they are seen by both a user and a sensor, this mean that they are close enough to exchange information.
2. The Wi-Fi signals received in a room and their respective RSSI vary a lot over the time, due to interferences. For example, if we do 2 scans in the same place at intervals a few minutes apart, we obtain a different number of networks available and very different RSSI. Therefore, to ensure the accuracy of our representative map of the rooms, we decided to perform scans over a two-hour time period, at 5 minute intervals; this represents 24 scans per room. This has just to be done for the static method, because one of the advantage of the dynamic method is that by doing periodical scans it can follow the variation of the signals, on condition that the period is short enough.

You can find a study showing these observations in the folder "*Wifi_Signal_Analysis/Wifi_Analysis_Over_Time*" of the Git repository of the project.

C) Algorithm to determine the footprint of a room

To map a room, we need to collect the information about all networks that we can detect. We search more specifically for the BSSID and RSSI of the networks. Then, once we have collected all this information on the received networks and their respective RSSI, we have to build the footprint of the room.

Let's take the example of the static method. We state that the result of the 24 scans made overnight is a list of x networks. We are going to see how many times each network appears

in the list of the *x* networks. The aim here is to select 5 different networks which will be the basis for the footprint of the room. The following algorithm allows us to find these networks:

1. If we have exactly 5 networks that appeared 24 times, these networks compose the footprint of the room.

2. If we have more than 5 networks that appeared 24 times, we are going to compare the mean of the RSSI, over the 24 scans, for each of these networks. The 5 networks that will compose the footprint of the room are the ones with the highest mean of the RSSI.

3. If we have (5 – *y*) networks that appeared 24 times, we will look at networks that appear 23 out of 24 times and take the *y* network(s) with the best mean RSSI, over the 23 scans, to complete the footprint of the room. But if we have less than *y* networks that appear 23 out of 24 times, we will look at networks that appear 22 out of 24 times, take the one(s) with the best mean RSSI, over the 22 scans, and complete the footprint of the room, and so on.

You can find in Annexe 1 a diagram representing this algorithm.

Note that we decided to compose the footprint with 5 networks because we wanted to see if this threshold is enough to have a good precision. The precision of the location could be improved by extending the number of networks that compose the footprint, but we need to have enough Wi-Fi access points in all the areas in which we want to deploy this system.

4.1.2.2 User footprints

For the same reason that we perform several scans for the mapping of the room, we do 3 scans to collect the Wi-Fi networks available in a room with the user's smartphone. The only difference is that we save each networks only one time. Thus, we obtain a list of *z* different networks. This list of networks represents the footprint of the user. Then, a comparison between this footprint and room's footprint is performed on the server to determine the location of the user (cf the server part).

4.1.3 Footprint for the Bluetooth

It is possible to localise users with Bluetooth using the same method as we did with Wi-Fi to localise a user in a building. However, there are fewer Bluetooth access points than for the Wi-Fi, and most of them come from mobile phones or from smart devices (smart watches for

example). Therefore, it might be difficult to obtain a good mapping of the room and thus to have accurate results.

Another simple method can be used to locate users with Bluetooth. As this technology is a short-range network, if a user is able to detect the Bluetooth signal, with an RSSI lower than a certain value, of a smart sensor (temperature, humidity), it means that they might be close to each other and thus in the same room.

Due to a lack of time, we had to focus on only one method to try to localise precisely a user in a building. Thus, we choose to develop the Wi-Fi method because it appeared to be the most interesting and scalable one. In part 5.6 we discuss about adding the Bluetooth method to the system we developed.

4.2 Architecture

Our system is composed of four main parts: hardware, application, knowledge base and server. We will present each one later in detail, but here is a brief introduction to the architecture of our system (see figure 4).

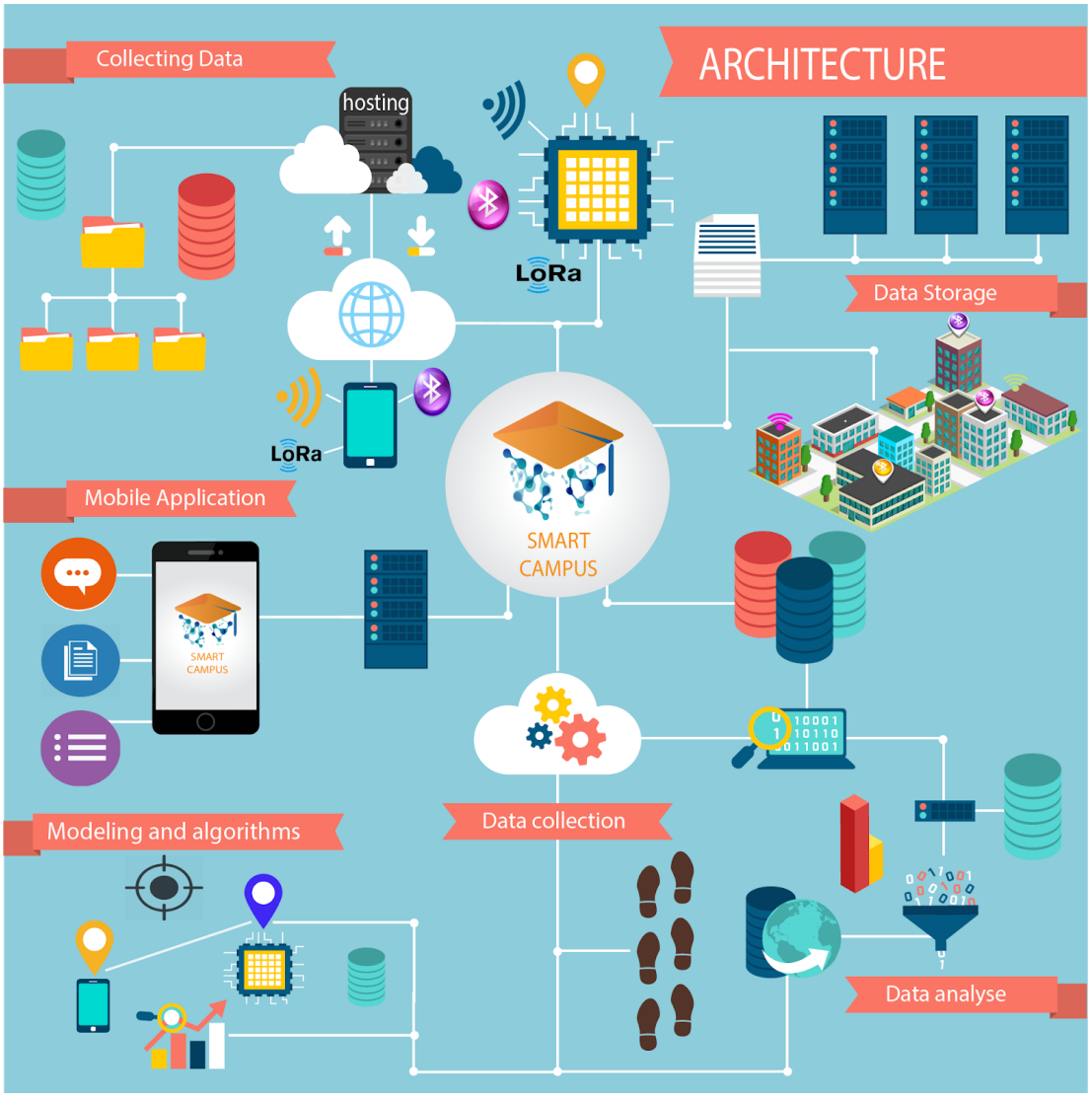


Figure 4. Architecture of the system.

We divided the architecture in two flows, one going from the hardware to the server and another going between the server and the application. Sections 3.2.1 and 3.2.2 explains how this two flows works separately.

4.2.1 Hardware function

First, we collected the networks data (RSSI + BSSID) across the building in each room, using the hardware. Then, we transmit this data to the server. In the static method, we transmit this

data just once with the corresponding room. For the dynamic method, the hardware collects the data and sends to the server periodically, also with the corresponding room. Doing so, the footprint of the room is actualized every minute. Afterwards, with this information, we run analyses using models and algorithms to generate the footprint of each room that is stored in the knowledge base. This information will be used later to compare with the smartphone footprint of the user.

4.2.2 Smartphone function

With the installation of the application on their smartphones, users can detect the available signals in their environment (list of Wi-Fi and Bluetooth). Once the application acquires the signals, it will send data to the server. As the footprints corresponding to each room are already stored in the knowledge base, the server can compare the data received from the application to the footprints in the knowledge base, and localise the user in the corresponding room. Once we have the corresponding room, the server sends to the application the information about the services that are available in that room at that time. Figure 5 shows the communication schema of the architecture.

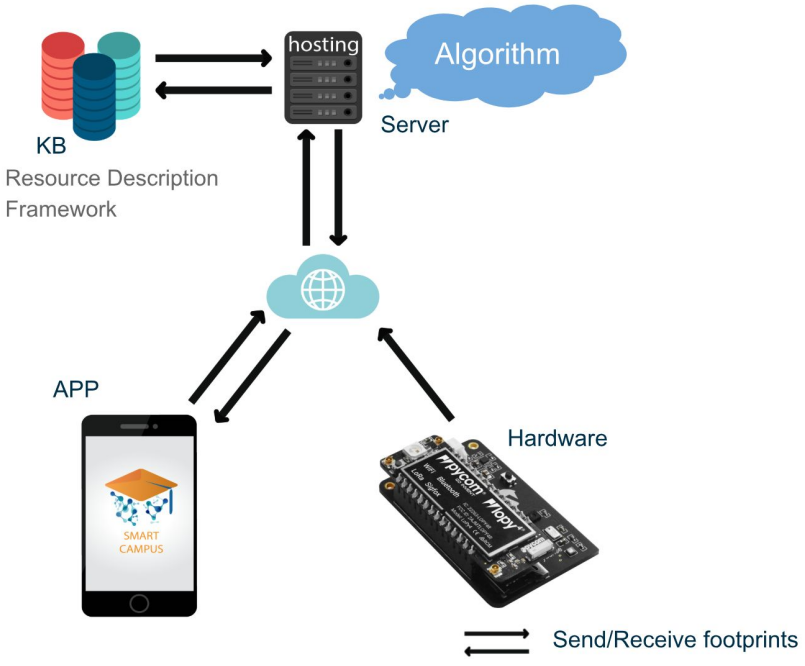


Figure 5. Communication skeleton.

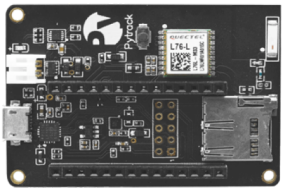
We only store the data collected by the hardware, and we do not store users data. Therefore , users do not need to worry about disclosure of personal information. When the smartphone application is closed, there is no data that is either transmitted to or stored by the server .

4.3 Hardware

4.3.1 Presentation of the hardware

To perform the mapping of the rooms, we used 2 complementary devices from *Pycom*:

Lopy 4: which embed Wi-Fi, Bluetooth, LoRa and Sigfox modules. It allows us to perform Wi-Fi and Bluetooth scans to collect the networks signals.



Pytrack: served as an expansion board for the *Lopy 4* and also embed a *GPS* and *GLONASS* module allowing us to compare the precision of our system to the *GPS/GLONASS* technologies.

Both of these devices are programmable in *microPython*. To develop, transfer and test the codes, we used the *Atom* IDE and to the *Pymakr* plugging.

4.3.2 Mapping

As explained previously in the footprint part, we used the Lopy-4 and Pytrack to collect the Wi-Fi networks in order to define the footprints of the different rooms of a building.

We can image two different scenarios to map the room:

- Map the rooms of a building by moving the hardware in each room and collecting the available networks.
- Put hardware in each room of the building and map the rooms periodically.

The second scenario needs as many hardware devices as there are rooms to map, but it allows us to obtain a dynamic mapping and thus is more efficient. In fact, when a smartphone sends to the server the networks it has received, we can compare them with a real time footprint of the room. Moreover, a Wi-Fi access point can be removed or stop working, so if we have one piece of hardware in each room, doing periodic scans to collect the available networks, we can take into account the modifications that occur in the building. However, this method can also increase the computation time of the server and so the response time (cf part 5.4)

4.4 Application

For this project, we choose smartphones as the medium to locate users on the campus and to offer context-aware services. Smartphones offer integrated Wi-Fi and Bluetooth service, with these being the two target technologies of our project. Another particularity of this kind of device is that users normally keep their smartphones close during the day, which is fundamental to a more accurate location service. We focused more specifically on Android. Developed by Google, this Unix based mobile operating system has acquired about 74% of the market share according to StatCounter Global Stats[6].

Once we had the target device, we chose to develop the application from scratch with Android Studio, the official integrated development environment (IDE) for Android operating system. It has a big community of Android developers and many tutorials are available on the internet. These two elements were fundamental during the project considering the amount of information and code examples that we found on the web.

Figure 6. Smartphone application.

4.4.1 Application Architecture

We based our application on the Android architecture components that help to structure the project in a way that is robust and maintainable. It is a simple, flexible, and practical approach that solves some common problems when developing User Interface (UI) applications.

We adapted the Android proposed architecture to our needs with the purpose of making the code expandable and functional. Nevertheless, separating the components into classes with clearly defined functions made further design more straight-forward.

Figure 7 shows the diagram of the architecture of the application. It is composed by a subset of the components, namely UIController, ViewModel and Communication. This diagram shows a basic form of this architecture.

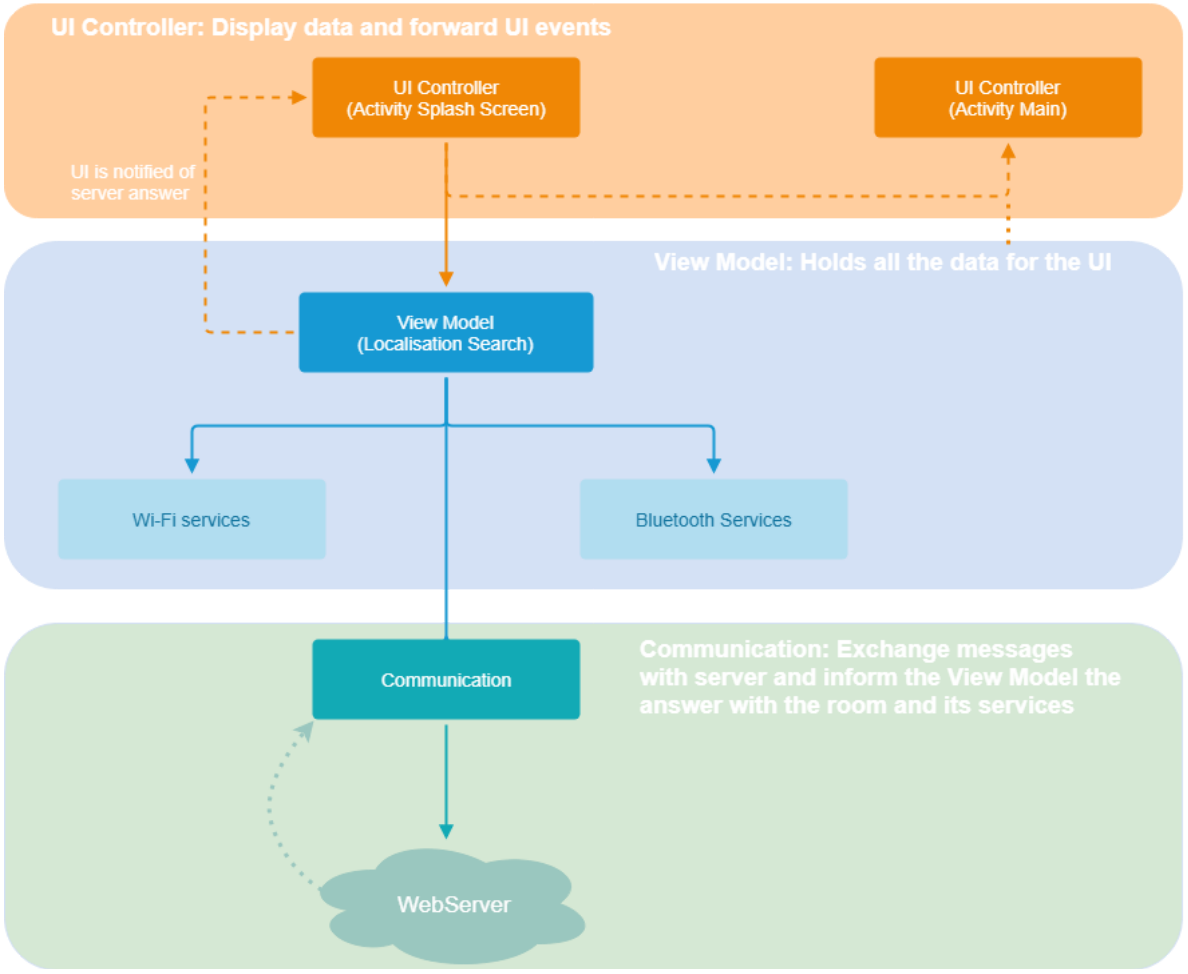


Figure 7. Diagram of the application architecture.

The UI Controller is the component that provides the user interface by rendering the data from the model in a particular format and arranges the interaction of the application with the user.

The ViewModel component contains all the information about the problem domain and controls the logical structure of data (data manager). It is also responsible for activating the Wi-Fi and Bluetooth scans, so we can have all the information about our network. This component does not depend on the user interface and can directly respond to information requests or instructions to change its information for example.

The Communication is the component that connects the application to the server. It is responsible for the communication between them by receiving and transmitting data. The communication component receives the Wi-Fi and Bluetooth footprints as input coming from the ViewModel. It performs the appropriate interactions with the server which, in turn, sends an actualization to the UI Controller using the View Model as an intermediary.

In doing so, we reached a low coupling of the code and this made the application development faster. In addition, it made the application more flexible to implement new features in future iterations because of the separation of responsibilities.

4.4.2 Operation

The Smart Campus application provides a very simple interface designed by the group with the software Adobe Illustrator[7]. The first screen that appears when the user opens the application is a Splash Screen that is shown in figure 6. It consists of a window that covers the entire screen containing the logo of the project and the name of the application. This launch screen appears while the location system is running. When the application receives the answer from the server with the user location and all the information about the room, the user is taken to a more functional screen where the context-aware services are available.

While the Splash Screen is displayed, the application will start the Wi-Fi services and the Bluetooth services. If the Wi-Fi or the Bluetooth is not activated at that moment, the application will exhibit a permission request (see figure 8) to activate and use these services.

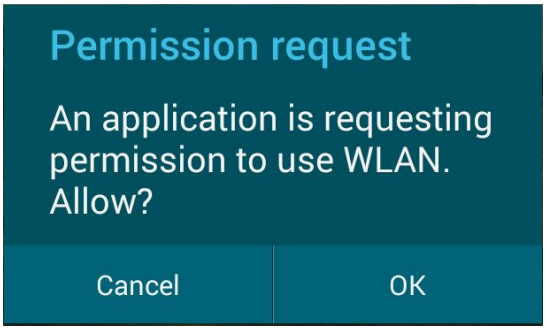


Figure 8. Permission request to access Wi-Fi information.

With access to Wi-Fi and Bluetooth services, the application makes 2 scans to collect as many signals as possible in the room. During the scan, the smartphone searches for all available Wi-Fi and Bluetooth signals to identify its BSSID and RSSI. These two pieces of information about all Wi-Fi and Bluetooth signals that were captured during the 2 scans are temporarily stored in different HashMaps that do not contain duplicate records., for example, if a Wi-Fi signal was detected in the 2 scans, that means its BSSID appeared twice with different RSSI in each measurement, so only the last RSSI will be kept and the other one will be ignored.

After collecting the smartphone footprint with 2 scans, the smartphone sends this information to the server as is shown in figure 9. The data-interchange format which was chosen was JSON because it is lightweight. The client-server communication is implemented with HTTP protocol. The application uses the HTTP POST request to send the signal information and

waits for an HTTP response with the room information. The server is responsible for analysing the footprint and comparing it with all rooms footprints to localise the user on the campus. It is also the server that has all the room information, for example, the timetable of that specific room that day.

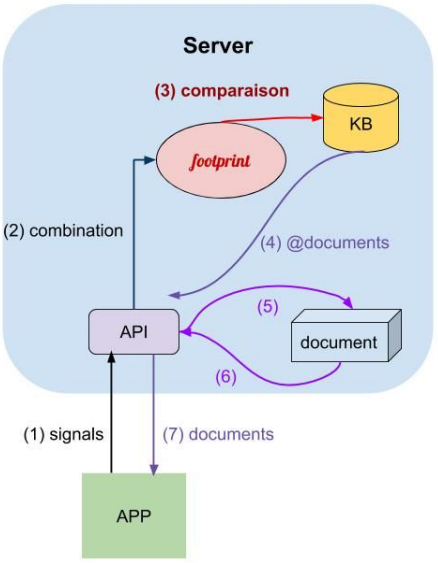


Figure 9. Communication between server and application.

The application receives the room information in JSON format. The data is decoded and displayed in a second screen with all the services that are available for that room. This part will be explained in the "Use case" section.

The main problem in this part of the project is the the battery consumption trade-off. Perform Bluetooth and Wi-Fi scan consume a lot of battery of the smartphone. That is why after Android 8.0 (API level 26), restrictions regarding permissions and the allowed frequency of Wi-Fi scans were introduced.

Android 8.0 and Android 8.1:

Each background app can scan one time in a 30-minute period.

Android 9 and later:

Each foreground app can scan four times in a 2-minute period. All background applications combined can scan one time in a 30-minute period.

This restrictions imply in a limited number of scans that we can make each 2 minutes (1 scan every 30 seconds). Once the user have access to a room services, it might have some delay to actualize the services after leaving the room.

4.5 Knowledge base

We used a knowledge base (KB) to store data. It is a kind of database using Semantic Web technology. Semantic Web data is represented using a technology standard named Resource Description Framework (RDF). RDF is a graph (web-like) structure that links data elements together in a self-describing way (cf Semantic Web technologies).

4.5.1 Why we chose Knowledge base

Firstly, it simplifies the readability of the data. Traditionally, the database is relational which means the data is stored in tables, and the columns are used as the indicator of the data -- what it represents. The advantage of RDF data is that details about it are always immediately available. To understand the data, the developers no longer need explicit knowledge of what data means.

Secondly, it builds connections for different databases with the same theme. On the Semantic Web, vocabularies are used to classify the terms that can be used in a particular application, characterize possible relationships, and define possible constraints using these terms. While ontologies are vocabularies for more complex and possibly quite formal collections of terms which can be used to describe a database.

In addition, we can make deductions in KB. Generally, it means wecan add a new kind of definition based on the data stored in the KB. This is often the reasoner which does the job, because it can look through the relations between the data and complete the KB. We defined the objects stored in the KB into different classes (cf 2.3 Ontology).

For example, we designated all the rooms of GEI as and only as a class of 'Place' in our KB. If someone enters the room 'GEI_105', we can make a deduction and mark the room 'GEI_105' as a class of 'Place With People' at the same time as 'Place' (see figure 10).

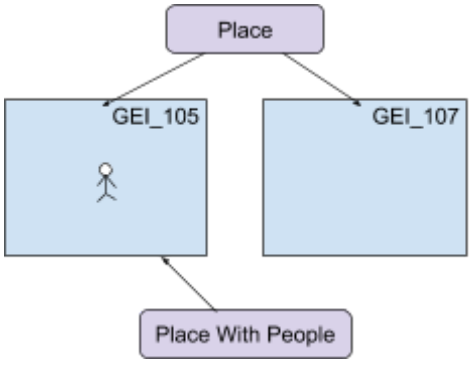


Figure 10. Example of "Place With People".

The KB stores the last location of the user. When a user moves to a new location, the signal environment is sent to the server which compares the new location to the location which was last stored in the KB. If the server determines the location has changed, the system adapts services to the new location. The KB allows us to determine which class to assign to the location, for example 'Place With People'. Once this class is known, specific services can be attributed to rooms. For example, we can turn on lights or open windows with the help of some actuators.

4.6 Server

The server is on a Raspberry Pi 3 and it is composed of an Apache server for php algorithms and a Fuseki Server for KB.

The server has 2 main roles:

- First, it gets the list of networks provided by the hardware and it computes a footprint that is related to a room/hardware
- Second, it compares the list of networks provided by a smartphone, with the hardware's footprint to determine if the user is in a mapped room in order to give access to services.

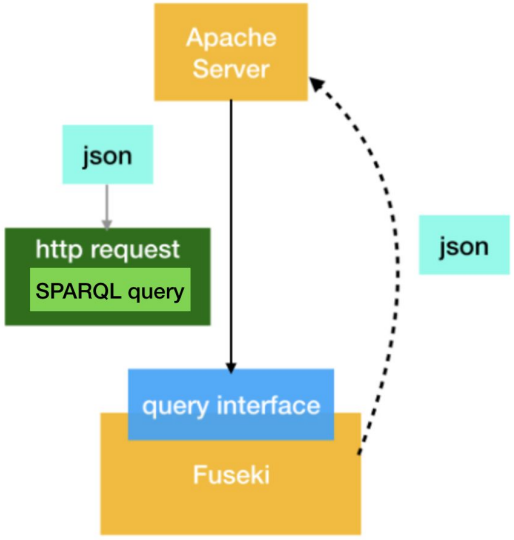


Figure 11. Communication between server and KB.

The Apache Server communicates with the Fuseki Server thanks to its interface which receives a simple http request to query the KB using SPARQL query. We are using JSON format to transmit data between server parts, which are one of the simplest formats to encode/decode.

4.7 Ontology

Since we use the Semantic Web technology, we need to create an ontology to apply it in our knowledge base (KB). Because the ontology can define the contents of the KB and their relationships.

First, we made a list of the subjects needed to be stored in the KB. Then, we searched for an existing ontology which could be used. Finally, we tried to improve it so that our work can be useful for other researchers.

4.7.1 Main Content

Here are the classes of information we need to store in the KB:

- **Hardware**

  id of the hardware

  related footprint

- **Footprint**

  related hardware

  related location (Room,etc)

  information of the signals around (Wi-Fi, Bluetooth)

  the time when the footprint was formed

- **Place**

  related footprints

- **Observed Signal** (Wi-Fi, Bluetooth)

  related hardware

  RSSI (signal force), BSSID (id of the signal)

- **Time**

  the time when the footprint is made

The following are the subjects which can support our scenarios:

- **Document**

  related location(s)

  the documents sent to users in certain locations

4.7.2 Connect to existing ontology

In order to create this ontology, we began by searching for the existing ontology for IoT. The following ontologies are what we found on the Internet:

SOSA (**Sensor, Observation, Sample, and Actuator**):

A lightweight but self-contained core ontology for IoT sensor activities. In our project, we only need to use the observation part to store the observed signal data. Here is a schematic which can explain the components of the observation in SOSA ontology. Combining it with our own ontology, we only need to take the class 'Observation', 'Sensor', 'FeatureOfInterest' and 'Result' (see figure 12).

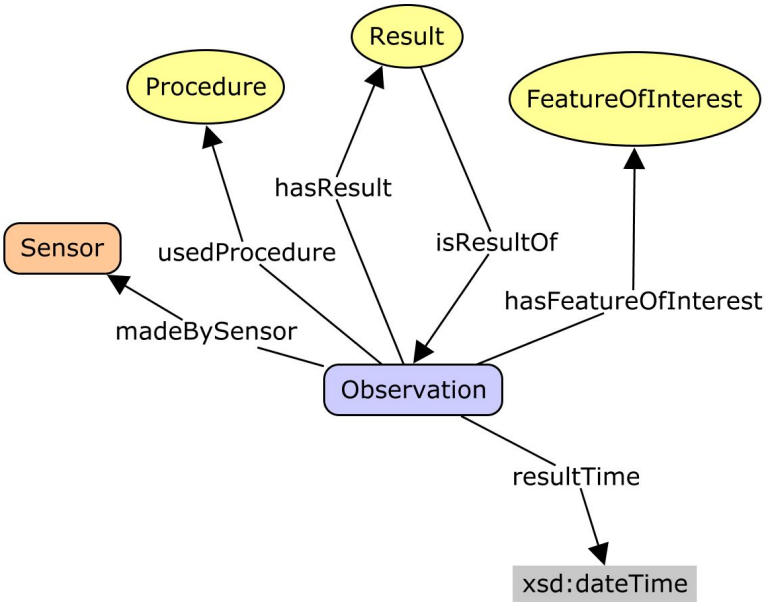


Figure 12. Observation part in SOSA ontology.

Observation: the term of action which observes the signals

Sensor: the sensor observes the signals

Feature Of Interest: observed signals

Result: the bssid and the force of the signals

As we need to store Wi-Fi and Bluetooth signals, we tried unsuccessfully to find some existing ontologies. Therefore, we decided to create an additional ontology which can describe the signals observed.

For the location on the campus, we found the 'Teaching Core Vocabulary Specification' ontology (teach). It is an ontology specifically for describing teaching situations in schools. However, we only used its 'Room' class to specify the location for our footprint.

4.7.3 Created ontologies

In this section we are going to explain the ontologies that we created for this project, along with the usages and the components of them. We have created 2 ontologies: wlan_v1[10] (ontology module) and iss_v1[11] (main ontology). The ontology module simplifies the main ontology. The wlan_v1 is only for describing wlan signals and the iss_v1 is the ontology we used on the KB. We deployed wlan_v1 and SOSA ontology in the iss_v1. More detailed descriptions of wlan_v1 and iss_v1 are as follows:

Ontology module

wlan_v1

Description: The first version of the ontology for describing observed wlan signals.

Usage: An additional ontology which can describe the wlan signals observed, including the strength of the signal (RSSI), unique id of the signal (BSSID) and the given name of the signal (SSID).

Main ontology

iss_v1

Description: The first version of the ontology for context aware IoT system. Only have the observation part of the sosa.

Usage:

1) Storage of the footprints measured by the hardware related to the location in GEI building.
2) Storage of the time when the footprints were formed (make sure the footprints are up to date).

Ontology imported: wlan, sosa

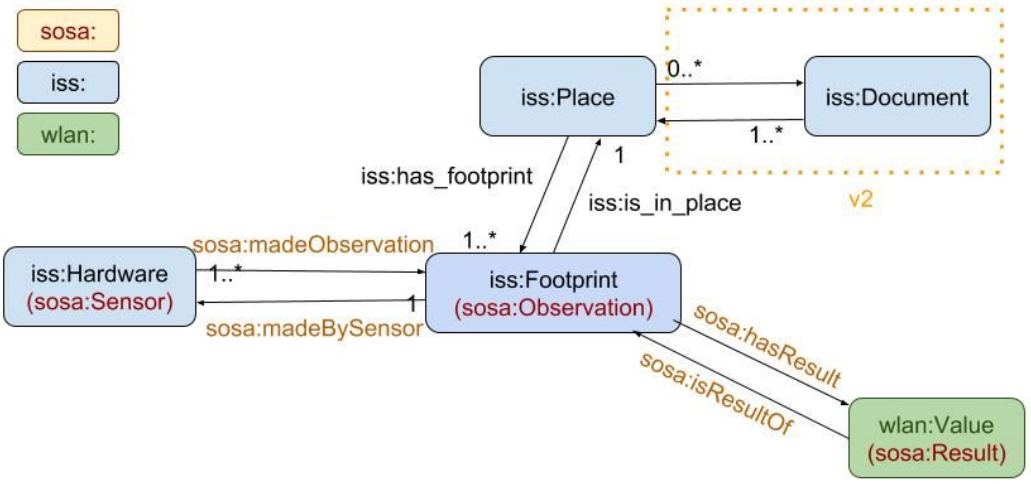


Figure 13. Ontology iss.

iss_v2

Description: The second version of the ontology for context aware IoT system. Not only have the observation part of the sosa, but also the actuator which can react to newly formed deduction.

Usage:

1) Storage of the footprints measured by the hardware related to the location in GEI building.
2) Storage of the time when the footprints were formed (make sure the footprints are up to date).
3) Storage of the rules which can deduct new rules in the KB.

Imported Ontology: wlan, sosa

Scenario:

1. Turn on the lights when someone enters the room.
2. Give information on conferences in the room.

5. Results and discussion

5.1 Comparison between GPS and our system

Here we present the study we have done to determine if GPS or the system we developed is better for indoor localisation.

The tests were done in our department (GEI), in 5 different rooms:

- Room 15 on the ground floor.
- Room 105 on the first floor.
- Rooms 109 & 111 on the first floor. These 2 rooms are just separated by a wall and a door.
- Room 213 on the second floor.

For each room, we determined the footprint, as described in the previous parts. Then, we collected the GPS coordinates (longitude and latitude) 3 times for each room and we used our smartphone app to determine our location, using our system, 10 times for each room.

5.1.1 Localisation using the GPS

On the picture below, we can see the real location of the 5 rooms of the GEI used for the test.

Figure 14. Real position of the GEI rooms.

To determine the GPS coordinates in each room, we used the GPS module on the Pytrack. During the tests, we noticed that it is impossible to obtain the GPS coordinates in the room if

we do not put the Pytrack outside the windows to allow it to make a fix, which means to acquire a satellite signal and determine the latitude and longitude. Once the GPS has made its first fix, while we did not power the Pytrack off, we were able to get GPS coordinates without putting the hardware outside the window.

The pictures below show the results of the 3 fixes done in each room.

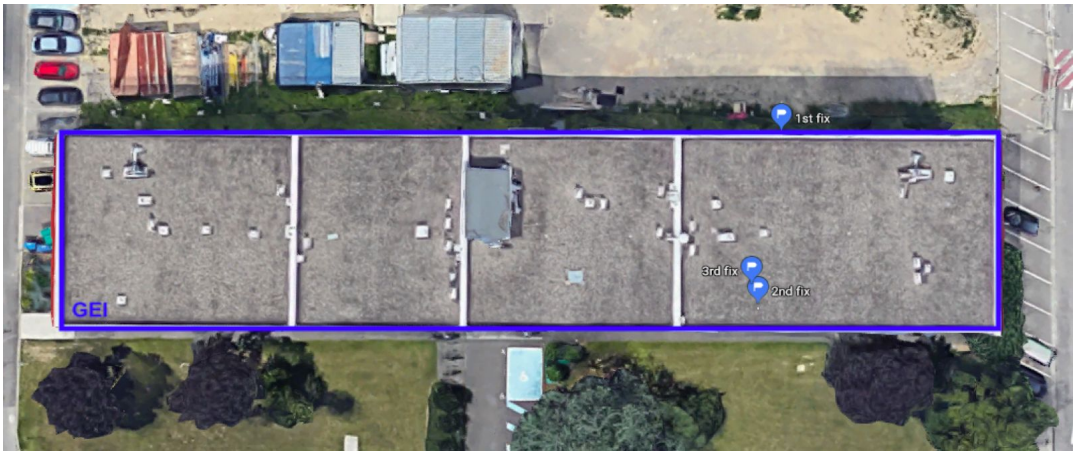


Figure 15. Result of the GPS fix for room 105.

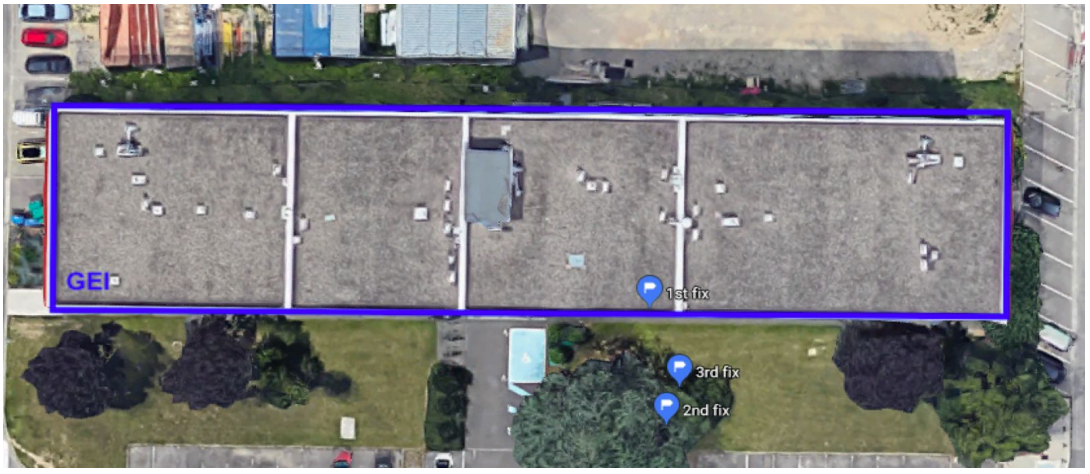


Figure 16. Results of the 3 GPS fix for room 109.

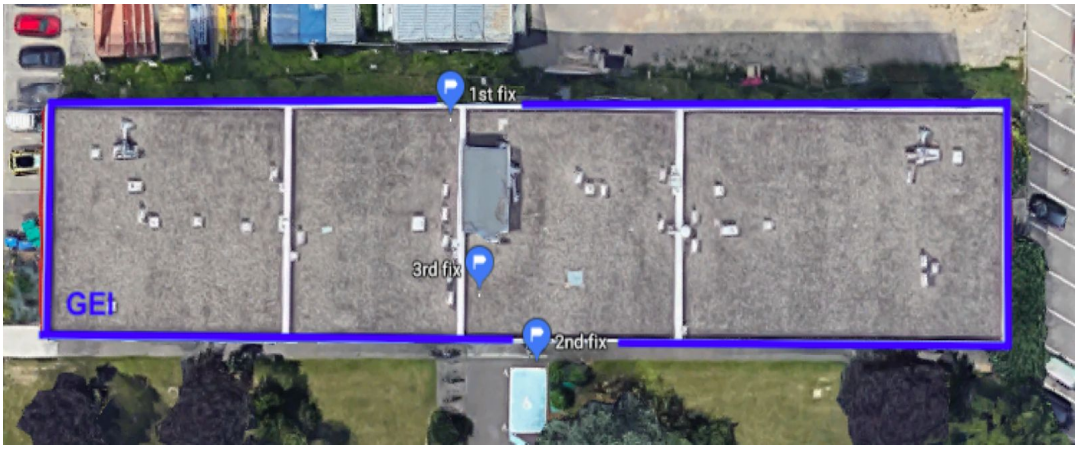


Figure 17. Results of the 3 GPS fix for room 111.

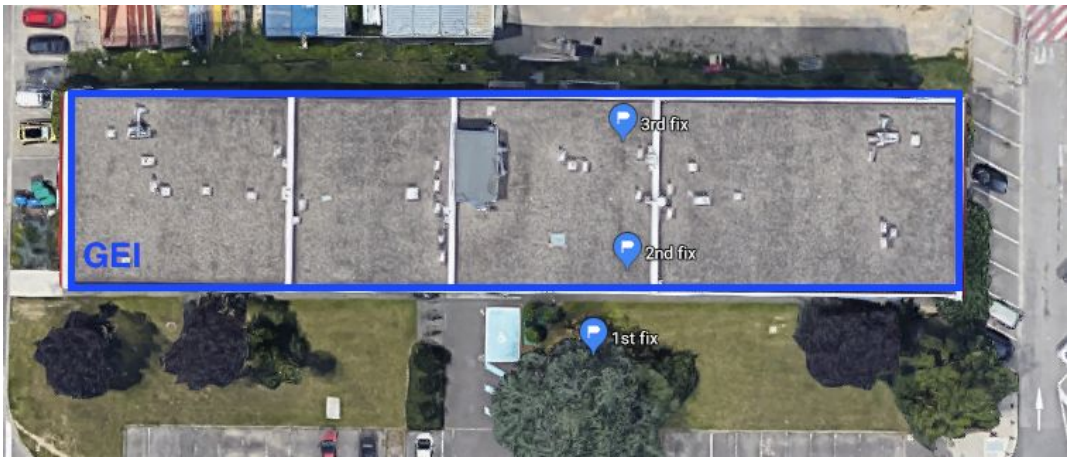


Figure 18. Results of the 3 GPS fix for room 213.

Note that for room 15, the results were very incorrect (and surprising) that we went twice in the room to collect the data and we took 8 fixes.

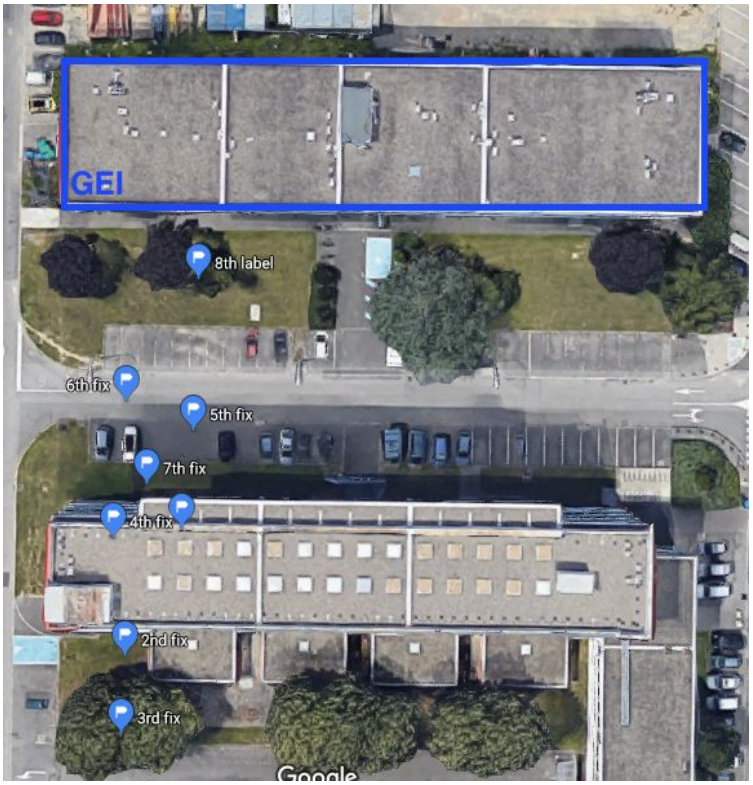


Figure 19. Results of the 3 GPS fix for room 15.

From these results, we can conclude that:

- For a given room, the GPS coordinates and thus the location, vary a lot between the different fixes.
- Most of the time, only one out of three fixes provides correct information about the location of the room. Moreover, the other 2 fixes give completely erroneous locations, which can be situated outside of the building.
- We do not have information on the floor location in the building. In fact, the libraries available to get GPS information from the Pytrack GPS module do not allow us to have information about the altitude.

5.1.2 Localisation using our system

In order to verify the efficiency of our system, we tested different methods by using our smartphone application in the 5 different rooms in GEI. In each room, we launched the application 10 times in different points of the classroom. The results for the static and dynamic methods are going to be explained in sections 5.1.2.1 and 5.1.2.2 respectively.

5.1.2.1 Static Methodology

A. Limited number of networks in the footprint

For this first experiment we defined the footprint by using 5 networks for each room. We first chose this number because at the beginning the calculation of the footprint and the comparison between the footprints of the rooms and the ones of the users' smartphones was done manually, and also because this number appears to be sufficient enough to have an accurate localisation. The chart below shows the results for each room with this methodology:

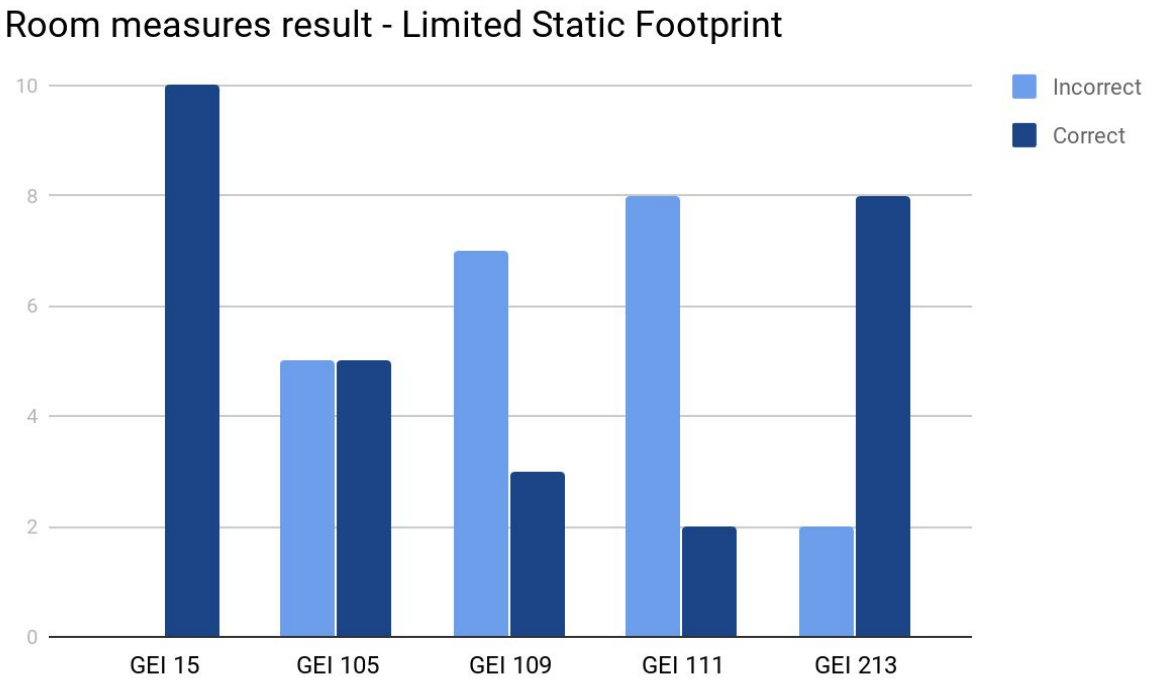


Figure 20. Room measures result - Limited static footprint.

As we see in the chart, the results are related to the distance between one room and another. For example, for the room GEI 15, the algorithm works perfectly because this room is far away from the others. For the rooms GEI 109 and GEI 111, the algorithm has a high percentage of errors (in the GitHub repository[9], you can find more detailed data). This is due to the physical proximity between them and also because the room GEI 213 is just above.

As a conclusion, we see that a system based on a previous map of the building with limited number of networks in the footprint does not work as we expected. Network systems are dynamic and interferences occur all the time. The smartphone footprint varies with the network changes and interferences during the day. For this reason, it is normal that the smartphone captures different RSSIs for the same network in two measurements that are taken at different moments (even within seconds of each other). Also, it is normal that we do not

capture a network for a fraction of time. In many tests we observed that some of the networks were not detectable and after they appeared some time later. This is due to wireless interference that results in a intermittent signal or strong variations of RSSI of the network. The usual source for Wi-Fi interference in this kind of situation is the number of wireless devices operating on the network. As the GEI has a large passage of people during the day, we have a large quantity of devices emitting Bluetooth and Wi-Fi signals producing noise that typically interferes with the function of an adjacent device.

To improve our results, we thought about two solutions: adding more networks to the room footprint or making a dynamic footprint. The results for these two solutions are going to be explained in the sequence.

B. Unlimited number of networks in the footprint

With the automation of the calculation of the footprints and of the comparison, we made more tests to determine the precision of our system and we saw that it is not accurate enough.

Therefore, one solution to improve the precision is to increase the number of networks which compose the footprints of the rooms. In fact, by increasing this number, we increase the number of points of reference for the comparison and thus the accuracy of the localisation.

However, the results were even worse. As the rooms are really close, we detect the same Wi-Fi networks everywhere. That is why the score is the same for rooms that are close to each other. As the score is the same, the algorithm can not differentiate one room from another. To work around this problem, we would have to find the most similar room footprint by comparing each network RSSI of the smartphone footprint with the RSSI of the room footprint. We did not had time to implement this method, but we would probably have more problems with interference as explained in the last section.

5.1.2.2 Dynamic Methodology

In the mapping section (3.3.2), we talked about the dynamic footprint. This method consists in putting hardware in each room of the building and mapping the rooms periodically too have a dynamic footprint that varies during the day with the network conditions. In this case, we use the real time footprint of the room, so interferences and intermittent signals do not interfere in the result. In addition, this help us to improve our method by taking advantage of mobile points that are in the room at that specific time and also Bluetooth short-range wireless signals. For example, if a mobile phone is in the room with the Bluetooth connection on, the hardware could actualize the footprint on the server making this new footprint more reliable by taking in account this new network point. When the hardware detects a Bluetooth signal, it means that the device is very close, so any device that also detects Bluetooth, is also close to

the hardware. Therefore, this approach is a way to improve our system even though this could compromise the scalability of the project because of the high cost of putting equipment in each room of the campus. During our tests that we describe in this section, we did not considered the Bluetooth signals, so the actual results could be improved as we describe in the section 6.4.

A. Limited number of networks in the footprint

For this first test, as in the limited static footprint, we defined the footprint by using 5 networks for each room. We run the tests only in target rooms (GEI 111, GEI 109 and GEI 213) that are close to each other. The chart below shows the results for each room with this methodology:

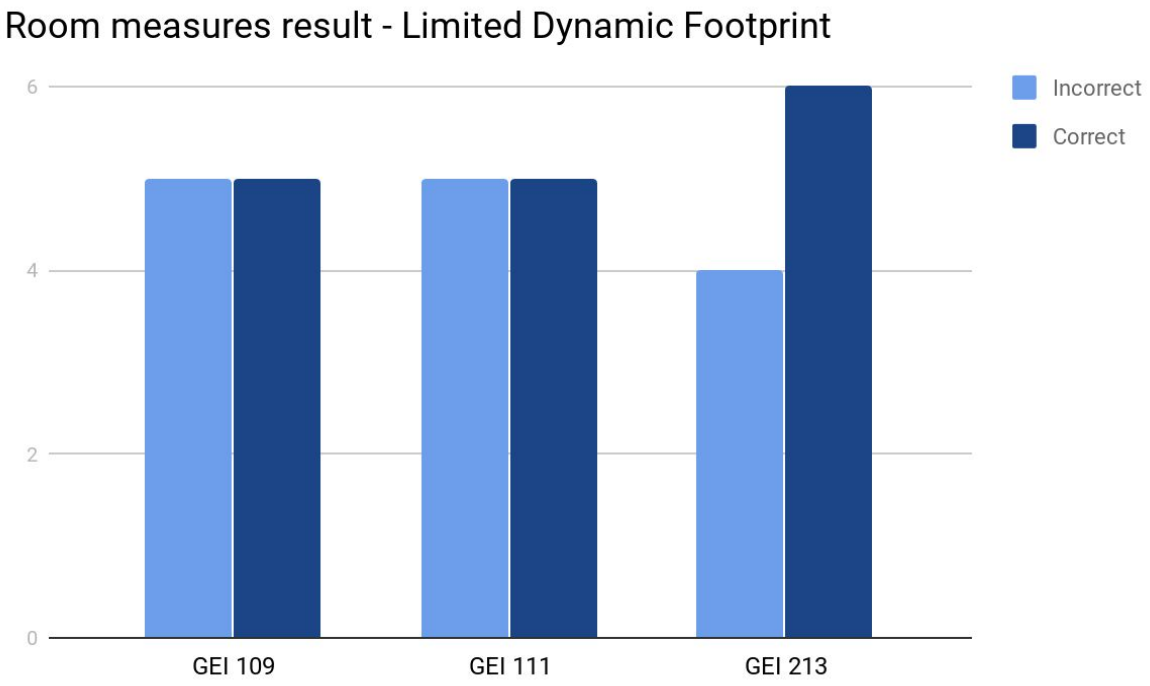


Figure 21. Room measures result - Limited dynamic footprint.

As we see in the chart, the results improved with this methodology by reducing the influence of networks interferences in the precision of the system. Nevertheless, the results were far away from a satisfactory precision. By limiting the number of networks in the footprint, we lost some precious information about the environment and that resulted in a loss of precision.

Therefore, we performed experiments with no limit to the number of networks. The results are in the sequence.

B. Unlimited number of networks in the footprint

In order to have a better precision, we implemented the dynamic method with unlimited number of networks in the footprint. We run the tests only in target rooms (GEI 111, GEI 109 and GEI 213) that are close to each other. The chart below shows the results for each room with this methodology:

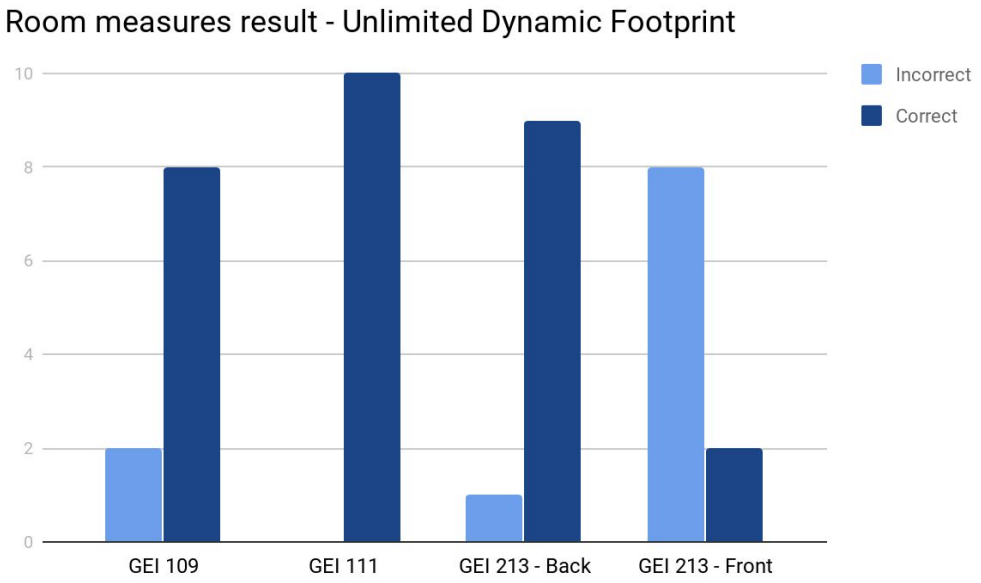


Figure 22. Room measures result - Unlimited dynamic footprint.

As we see in the chart, the results improved in a really satisfactory way by using this methodology. The only bad result was in the GEI 213. To understand why, we need to consider the room structure. Figure 23 shows a representation of this specifique classroom.

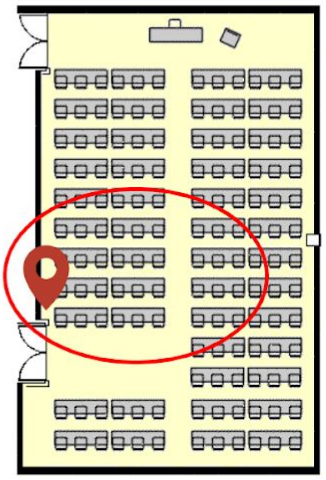


Figure 23. Representation of the room GEI 213.

This room is bigger than the other rooms that we tested. In our test, we placed the hardware in the back of the room and in the left side because of the power plug. That is why the results were much better in the back than in the front of the classroom. The only bad result that we had in the back of this room was when we performed the measure in the right side, far away from the hardware. As we can see in the figure 23, the range of the hardware is limited (red circle). Therefore, it is better to place the hardware in the middle of the room trying to cover both sides equally. The red circle left side is smaller compared to its right side because of the wall interference. That is good thing for our system, because a student in the hallway will not be localized as being inside of the room.

As a conclusion, we see that in big rooms, we need to place at least 2 hardwares in strategic points in order to have a better cover of the area. Besides that, the results shows that the system is accurate enough using this methodology. The precision is not 100% yet, but that can be improved by adding the Bluetooth in the calculus or implementing more efficient algorithms to the comparison of the footprints.

However, the most you increase the number of networks in the footprint, the most the time of treatment to make the comparison between the footprints will be important. In this test, we observed that the time of computation doubled, going from 0,5 second to, at least 1 second. This time is acceptable, but we have to take into account that during theses tests we had only 20 networks for the rooms footprints and only a single smartphone requesting the server. But in a smart campus with more networks available and hundreds of users using the application at the same time, the response time can be rapidly increase.

5.2 Use case

To make a live demonstration of our system we thought of a specific situation in which our application could potentially be used. We implemented a simulation of the timetable of the day of our final presentation. The result is shown in the figure 21.

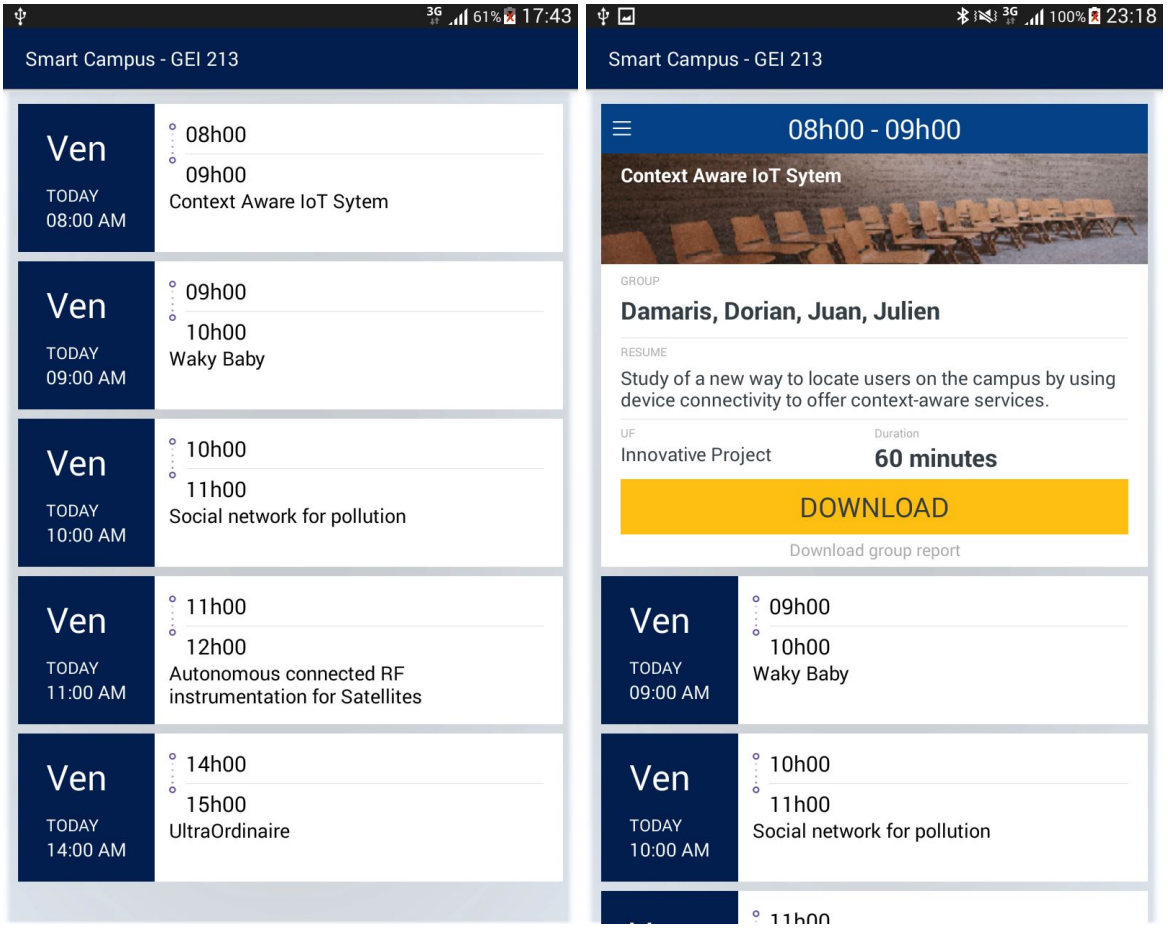


Figure 24. Use case application.

For this example, we used the room “GEI 213”. When the user enters in this room, he gains access to the timetable of the room that, in this case, is composed of 5 presentations. The timetable is composed of one card for each presentation. The card shows the start time, end time of the presentation and also the name of the group. When a user opens the card of a specific presentation, more information is displayed: the name of the team members, a resume of the project, the discipline (UF) and the duration of the presentation. In addition to this, the user can also download the group report for example. Each card has the specific information about each group. This information is sent by the server that has the stored data.

As soon as the user leaves the room, access to the room’s timetable will be denied but new information will be displayed depending on the new footprint of the device.

6. Future works and improvements

6.1 Other Signals

In our system, we only use the Wi-Fi signals data to compose the room footprint. It is still possible to add Bluetooth, LoRa or any other wireless signal without changing the architecture of the project. However, this could have some impacts on the server algorithms that might be reformulated depending on the signal that we want to add to compensate for distorting factors between the different technologies. For example, we could make weighting in the reliability of the signal. Bluetooth signals, that is a short-distance wireless signal, could have a higher importance in the footprint.

6.2 Use cases

For future work, we could implement more use cases in order to offer more services to create relevant experiences and optimize services on campus. Firstly, we can think of specialized services for different types of users: teachers, students and manager. Teachers, for example, could have access to student lists and could post files and information on a specific class in a specific room. Students could have access to a chat where they could talk about the class with colleagues that are in the same room. The manager could control lights, doors and windows in a scenario with a completely connected campus. We could also think of a map service to help users that do not know the campus to find buildings and move around more easily through the university.

6.3 Security

For the security of our system, we thought about separating the users by groups and giving them different logins. If we want to provide a more specific service to the target users, we need to collect private data on the users. For example, we could connect our application with the current INSA login system used for the university e-mail. However, in our current system, we do not want to collect personal data to avoid user information leakes. To apply this idea, we need to ask the users if they are willing to give us their personal information. These kinds of approaches could compromise the application popularity as users could think that we are storing their location (which is not true) and their information.

6.4 Taking Bluetooth devices into consideration

To improve the accuracy of our localisation system, we can take other wireless into consideration. One the most available in our environment, after the Wi-Fi, is the Bluetooth.

Adding this network to our system can be done easily. In fact, the comparison between the footprints can be done in the same way for the Bluetooth as it is done for the Wi-Fi.

However, as most of the Bluetooth devices comes from mobile phones or from smart devices (smart watches for example), the implementation of this technology makes sense only if we consider a dynamic mapping of the rooms.

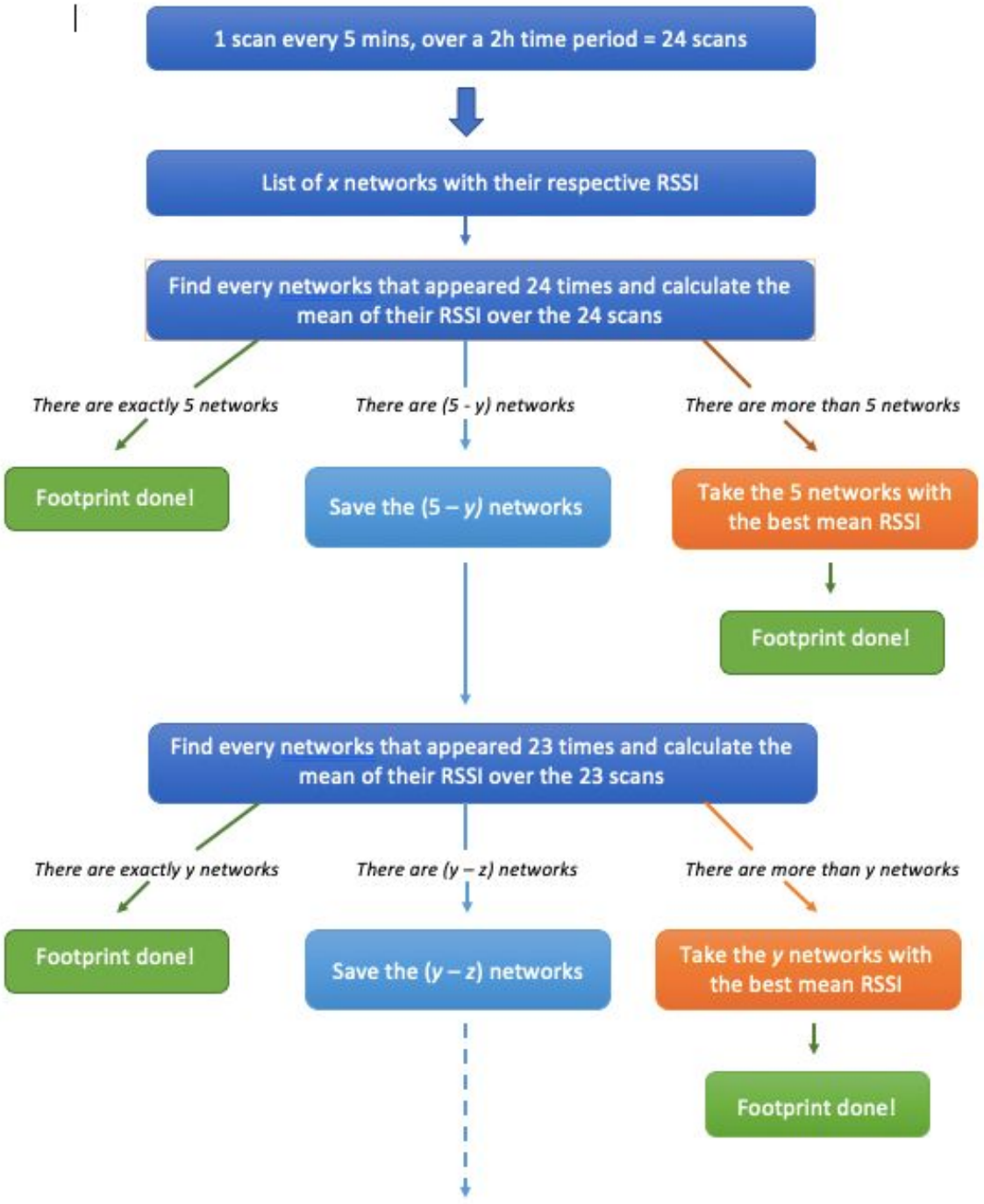
Therefore, as the Bluetooth is a short range network, this can improve the precision of the localisation because it adds information about the close environment of the devices and thus can help to determine the good room when the Wi-Fi method encounters its limits.

7. Conclusion

The context aware IoT system can be a relative good solution to the localisation indoor. Not only we can apply it inside a school or university, but also it can be used for a city. Since the system locate users and secure access to data and applications by using device connectivity as a context, the services can be provided to the nearby users. Wi-Fi and Bluetooth technologies are used as parameters, and this concept could be extended to any other wireless signal-based technology. To do a static mapping, we only need one hardware to collect the footprints. It is suitable for a relative stable environment. For a changing environment, we recommend a dynamic mapping with hardwares positioned in each room. If the system is full implemented, the users can experience indoor localisation and connected services based on it. This project is a first step in creating a smart-campus that can provide context-aware services in a fully integrated intelligent university.

Annexes

- **Annexe 1: Diagram representing the algorithm to find the footprint of a room.**



- **Annexe 2: Semantic Web technologies**

Resource Description Framework(RDF)

RDF is a standard model for data interchange on the Web. It extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (a "triple", see figure 17). This simple format can not only represent the data, but also the relations between them. This gives us the chance to create more complicate databases than traditional ones.

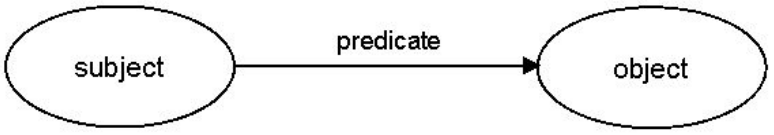


Figure 25. Triplet.

Ontology

An ontology encompasses a representation, formal naming, and definition of the categories, properties, and relations between the concepts, data, and entities that substantiate one, many, or all domains.

Every field creates ontologies to limit complexity and organize information into data and knowledge. As new ontologies are made, their use hopefully improves problem solving within that domain.

Knowledge base

The original use of the term knowledge-base was to describe one of the two subsystems of a knowledge-based system. A knowledge-based system consists of a knowledge-base that represents facts about the world and an inference engine that can reason about those facts and use rules and other forms of logic to deduce new facts or highlight inconsistencies.

On the Semantic Web, the inference engine of the knowledge base is the perfect tool to improve the relations and the properties. It makes it easier to manage the data on the Internet.

Bibliography

[1] PROV-O: The PROV Ontology. [Online]. Link: https://www.w3.org/TR/prov-o/

[2] Semantic Sensor Network Ontology. [Online]. Link: https://www.w3.org/TR/vocab-ssn/

[3] The SSN ontology. [Online]. Link: http://www.w3.org/ns/ssn/

[4] The SOSA ontology. [Online]. Link: http://www.w3.org/ns/sosa/

[5] Trello Timeline. [Online]. Link:

https://trello.com/invite/b/KXAvtUjJ/96a8ed9e2621b3ddeabfd8bb35b62f5b/timeline

[6] Adobe Illustrator CC. [Online]. Link: www.adobe.com/Illustrator

[7] StatsCounter Global Stats. [Online]. Link: http://gs.statcounter.com/

[8] NetSpotApp: What is RSSI level. [Online]. Link:
https://www.netspotapp.com/what-is-rssi-level.html

[9] GitHub repository. [Online]. Link:
https://github.com/NSeydoux/ContextAwareIoTSystem

[10] wlan_v1 Ontology. [Online]. Link:

https://w3id.org/laas-iot/wlan

[11] iss_v1 Ontology. [Online]. Link:

https://w3id.org/laas-iot/iss_v1

Illustrations