VRIJE
UNIVERSITEIT
BRUSSEL

Graduation thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Applied Sciences and Engineering: Computer Science

USING AUGMENTED REALITY TO CONTROL DEVICE INTERACTION IN A WEB OF THINGS ENVIRONMENT

MAXIME LIBERT

Academic year 2019–2020

Promoter: Prof. Dr. Beat Signer
Advisor: Prof. Dr. Beat Signer
Faculty of Sciences and Bio-Engineering Sciences

VRIJE UNIVERSITEIT BRUSSEL

Afstudeer eindwerk ingediend in gedeeltelijke vervulling van de eisen voor het behalen van de graad Master of Science in Applied Sciences and Engineering: Computer Science

USING AUGMENTED REALITY TO CONTROL DEVICE INTERACTION IN A WEB OF THINGS ENVIRONMENT

MAXIME LIBERT

Academiejaar 2019–2020

Promoter: Prof. Dr. Beat Signer
Advisor: Prof. Dr. Beat Signer

Faculteit Wetenschappen en Bio-ingenieurswetenschappen

"Deze masterproef is (ten dele) tot stand gekomen in de periode dat het hoger onderwijs onderhevig was aan een lockdown en beschermende maatregelen ter voorkoming van de verspreiding van het COVID-19 virus. Het proces van opmaak, de verzameling van gegevens, de onderzoeksmethode en/of andere wetenschappelijke werkzaamheden die ermee gepaard gaan, zijn niet altijd op gebruikelijke wijze kunnen verlopen. De lezer dient met deze context rekening te houden bij het lezen van deze masterproef, en eventueel ook indien sommige conclusies zouden worden overgenomen".

"This master's thesis came about (in part) during the period in which higher education was subjected to a lockdown and protective measures to prevent the spread of the COVID-19 virus. The process of formatting, data collection, the research method and/or other scientific work the thesis involved could therefore not always be carried out in the usual manner. The reader should bear this context in mind when reading this Master's thesis, and also in the event that some conclusions are taken on board".

USING AUGMENTED REALITY TO CONTROL DEVICE INTERACTION IN A WEB OF THINGS ENVIRONMENT

Maxime Libert
Master of Science in Applied Sciences and Engineering: Computer Science

2019-2020

Abstract

Nowadays, the world of connected devices, also called the Internet of Things, is undergoing a great evolution. More and more "smart" objects are appearing and as many applications to control them are being created with them. The main issue with that is that this diversity of connected devices and the abundance of applications it creates is that it is confusing and cumbersome for all of us, even the most experienced ones. When most of our objects are connected, how will we remember and store all this information? In this master thesis, we propose a solution to this problem by presenting a detailed explanation of the implementation process of one of the first cross-platform IoT middleware. Based on the concepts of the W3C's Web of Things and the use of Augmented Reality, we have developed a web application that allows its end users to interact with any of their connected devices from only one place, our middleware. During our research, we were able put together a prototype that requires that requires very little computing power using only JavaScript and Python. By parsing the Thing Description document provided in the Web of Things standard, our system is able to generate any smart object's control and monitoring panel. The prototype we produced also proved to be highly appreciated by the end users who were kind enough to participate in our evaluation phase despite the ongoing health crisis. The architecture we have conceived makes it possible to transform any item that has a web browser and a camera into a universal remote control capable of interacting with any connected object on a local network. The concepts discussed in this thesis, the findings and conclusions observed therein provide great opportunities for the future of users' interactions with their connected devices.

Keywords: Cross-Platform Middleware, Web of Things, Augmented Reality, User Interactions, JavaScript, Python.

Acknowledgements

First of all, I would like to thank Prof. Dr. Beat Signer for his availability and support throughout my thesis. I would like to thank him especially for guiding me and pushing me to do my best. I am very fortunate to have had Prof. Signer as a promoter and advisor, his office door was always open whenever I had questions. I am very grateful to Prof. Signer because he always made sure that this thesis was my own work, but helped guide me in the right direction whenever he thought I needed it.

I would also like to thank my parents and my brother for always believing in me and for their support in all circumstances. A special thank goes to my girlfriend Charlotte for her trust, her encouragements even during the most difficult times and for being there by my side every day.

To conclude, a warm thank you to the professors and assistants I met during my years of study at ULB and VUB. They enhanced my passion for computer science. Many thanks also to all my friends with whom I spent my time at in the university: Alexandre, Benjamin, David, Pédro, Gregory, Quentin and Florian. They are the ones with whom I shared those years of study, life is simpler when you are a close-knit group that helps each other and have a good time together.

Last but not least, a big thank you to Nicolas my "sidekick" for more than ten years now. My brother from another mother with whom I share everything but especially with whom I have decompressed all these years. Our friendship is one of the most precious things to me.

I dedicate this thesis to Léon, my little star. He much, like this thesis, took nine months to build and was finally born in the middle of May 2020.

Contents

1 Introduction

The Internet, which originated from the ARPANET, has been around for a while now. The ARPANET was a network that connected in the beginning only four computers of different universities across America. Then, the ARPANET evolved, the number of connected computers exploded and was no longer limited to American universities. At that time, people started to use the term "Internet" to symbolise the international network.

One of the greatest inspirations in the creation of the Internet of today, was Vanevar Bush and his paper *As We May Think* [1] describing a fictional information system, "the memex", which allows its user to visualise information and create associative links between browsed documents. This instrument also allows users to create information, save it in their memex memory and share the contents of their memex with other people who have one. Later, Tim Berners-Lee proposed the concept of the World Wide Web (WWW) [2]. The idea was to use the hypertext technology similar to what Nelson imagined in 1965 [3] and create a system that allows the user to consult pages accessible on sites. As a result the first web browser was born. Then, after that, came the first search engines and with the time more and more applications working on the Internet.

As we saw previously, the Internet started to connect people to knowledge. Its users were only consumers of content and rarely creators, given the complexity required to create a web page at the time. Since the beginning of the 2000s, new interfaces have been developed to simplify navigation and add possibilities of interaction with the content of the web pages. These new interfaces introduced what is known as the Web 2.0: from this point on, the Internet user has become a content producer and can feed certain sites with text, images and many other types of media. Thanks to this, the user can also interact with others through forums, blogs, wikis or social networks. In short, Web 2.0 allows people to connect with each other and not only with knowledge. Moreover, the user is no longer a passive consumer of information.

Given the amount of data created and shared through the Web 2.0, researchers began to look for a way to help the machines understand the content of the sites and not just display the data. To this end, Web 3.0 was born nowadays, thanks to metadatas, machines can understand the context of websites and help the user with their tasks.

As we can see with time and improvements, the Internet has become a huge source of knowledge. It is now, as we know it, an immense space where people can access an infinite amount of information. It was built by people for people therefore until not long ago we could call it "The Internet of People". However, with the evolution of the Internet and the Web, more and more connected objects have appeared and are used to make life easier for people in their daily tasks. Those smart devices are grouped under the name of "Internet of Things" or IoT. The first known object of this group was a toaster and was invented in 1990 by John Romkey [4]. As one can see, the Internet of Things is connecting objects (things) and people together through the Internet, giving them the opportunity to interact and communicate the same way the first versions of the Internet did but now extended to a multitude of objects.

In parallel to the advent of the Internet, other sectors were also growing and developing. They all benefited from the evolution of computers and their connectivity, and naturally, the industry has grown around these very powerful tools. Gradually, all these objects and technologies have enabled the industry to take new directions and enabled two successive industrial revolutions (Figure 1.1).

We are now talking about Industry 4.0 and smart factories [5]. In addition to being very useful to the industry, IoT is also used to provide a more comfortable environment for our homes in what we call a "Smart Home" [6]. Furthermore IoT can help monitor and improve health conditions of home residents [7] and protect them [8]. In today's society, IoT is also used to solve serious issues such as global warming by reducing the carbon footprint of homes through an efficient management of their resources such as water and waste [9] but also in managing their energy consumption efficiently [10].

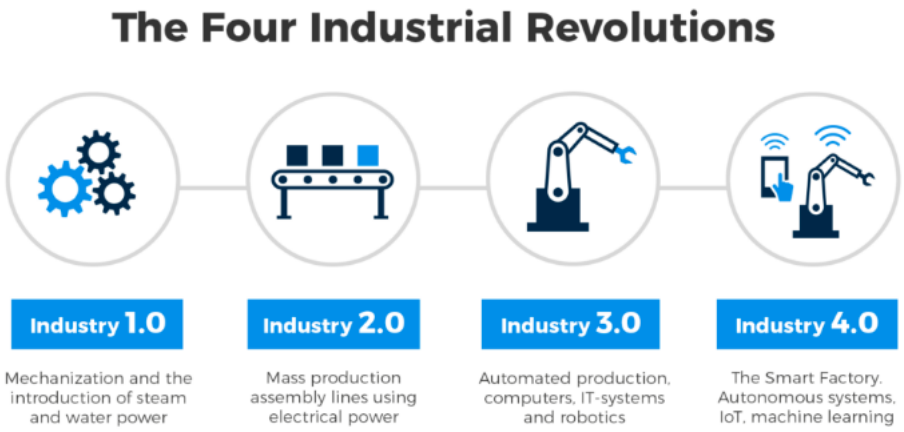


Figure 1.1: The four industrial revolutions

However, the Internet of Things does not connect all the objects to each other, it does not allow to exchange information between all the connected objects or even to create rules between all these objects. The problem lies in the fact that the producers of these objects prefer to create their own object ecosystems, thus prompt the user to buy only their own branded objects to allow more connectivity/functionalities. But above all the brands do that to get more money form users. Today the reality is that very few objects of different brands can communicate to each other. In fact, today's IoT could be described as an "Internet of Groups". To address this problem and improve interoperability between objects, some researchers have begun to define a concept called "Web of Things" or WoT[11, 12, 13, 14]. The latter would allow smart objects to be part of the World Wide Web and thus reuse the already well-known and successful protocols and standards of the current Web. The Web of Things acts as the glue that would allow what we called above "the Internet of Groups" to become the true "Internet of Things". Thanks to the WoT, it will no longer be necessary to use proprietary applications to interact with intelligent objects.

From now on, there is a central and common way to control all these objects, to create rules between them and to make them interact. One of the greatest strengths of the Web of Things is that it allows already existing objects to be part of its ecosystem. Indeed, there is no need to install a new operating system, upgrade anything, or embed extremely advanced technology. Only the fact of being connected to the Internet is necessary, which is already the case for all existing smart objects. The organisation in charge of standardising the Web of Things is the World Wide Web Consortium or W3C. It was founded by Tim Berners-Lee in 1994 and is an international community devoted to developing open web standards.

1.1 Problem Statement

As mentioned previously, it is troublesome to have to deal with an Internet of Groups rather than an Internet of Things. In the current state of affairs, we are too often confronted with IoT solutions created by brands which therefore only host objects included in their ecosystem. And so, regretfully, despite the daily progress of all the technologies involved, it is almost impossible for an end user without programming knowledge to interact with all its objects in one place or to define customised rules between different connected objects. There is a huge lack of easy to use and user-friendly middleware/authoring solutions for IoT objects.

In this thesis, we created a prototype of a control interface for IoT objects with such properties. We developed a cross-platform application that combines augmented reality (AR) and the Web of Things in order to control any devices and their interaction between each other in a way that is both easy to configure and deploy for the non-expert end users.

1.2 Research Question

In our research, the main research question that we address is:

How could we use the Web of Things principles to create a powerful cross-platform, easy to use application that enables the interaction with any smart objects?

In order to answer this question, we will have to ask ourselves a number of other sub-questions such as:

- *How could we use Augmented Reality to create natural ways to interact with connected objects using such an IoT middleware?*
- *How to design a lightweight middleware that generates the control panel of any intelligent object?*

We will also have to answer multiple questions linked to the implementation of this type of solution. Such as:

- *What are the best tools currently available to achieve the needed tasks?*
- *What is the best architecture for our solution?*

1.3 Contributions

This thesis has provided many contributions to our field of research. Here they are:

- We have relied on the resources of the Web of Things and its principles in order to create one of the first cross-platform IoT middleware available through the web.
- We have explored the use of Augmented Reality as a medium to create ways of interacting with connected objects that are as natural as possible to the end users.
- We have created a parser capable of generating the control panel of any object, as long as certain conditions are met.
- We have created a cross-platform system that consumes very little computing power. To do this, it we only used web languages and as few libraries as possible in order to preserve the fact that the application has to be able to be used with any web browser.

On a more scientific note we also:

- Investigated how to create more natural interactions with connected devices by using augmented reality to create several alternative designs that combine the principles of Human-computer interaction and many other principles such as the colour theory amongst others.
- Used the principles of the Web of Things to generate a faultless/fool-proof interface that displays the possible interactions with any objects in a simple and intuitive way for the users.
- Studied existing systems and technologies to implement the architecture of our middleware using the best means available.

1.4 Research Approach

To achieve our objectives, we are going to draw inspiration from the design science research method, an empirical research approach for science, proposed by Hevner et al. [15]. Here is how our research was conducted (Figure 1.2):

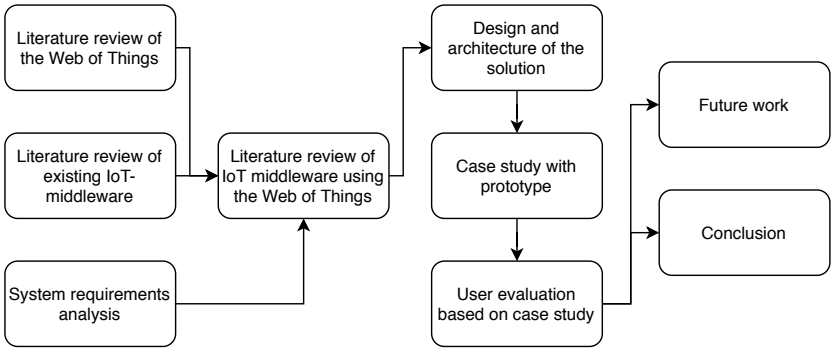


Figure 1.2: Research approach taken in this thesis

As one can see, we respect the methodology defined in their theory. First we proceed with the problem identification by doing literature reviews and meetings. Then we set the solution's objectives by defining the system's requirements based on the identified problems and the existing solutions. Next, we designed our solution in several iterations in order to get feedback during meetings and improve the quality of the final product. Afterwards, we tried to build a case study involving the prototype and conducted a user evaluation in order to complete the evaluation step of the methodology used. Finally, by explaining how to use our prototype to our testing users along with the writing of this thesis to documents the whole process, we carried the demonstration and communication part of the methodology.

1.5 Structure

The results of our research will be presented in the next chapters. This section presents an overview of what is contained in the remainder of this document.

2 Related Work and Background

This chapter introduces the existing concepts and principles related to the subject of this thesis in order to understand the knowledge on which this study is based and to bring the reader up to date with existing advances. To conclude this chapter, the requirements of the system will be stated.

3 Solution

This chapter explains the concepts, design choices and the architecture that we put in place for our system without going into details. This will allow the reader to better understand the purpose of this thesis and will therefore prepare them to understand the information given in the implementation chapter. This chapter addresses each research questions and answers them by explaining the concepts imagined for the purpose.

4 Implementation

This chapter describes in detail how we have implemented each feature of the system proposed as a solution in this thesis. For each feature the used libraries, languages and ideas will be explained in order to promote maximal reproductibility for our work. Also a section will be devoted to the challenges we faced while implementing our prototype.

5 Demonstration

This chapter contains a detailed explanation of how to navigate through the application and access its many features.

6 Evaluation

This chapter details how we conducted the user evaluation. Each topic will be addressed in detail from the test setup through the questions asked at the end of the test to the evaluation results.

7 Future Work

This chapter lists some points of improvement that could be introduced in the future. It also discusses the hopes we have for the technology developed in this work.

8 Conclusion

This chapter concludes this report by summarising the contributions, environment and difficulties experienced throughout this research work. It also suggests some improvements to the concepts on which this work is based and also talks about things we could have done better.

2

Related Work and Background

As one can see in our research questions, the objective of this thesis is to create a tool that will work as a middleware for existing technologies and allow its users better interactions with their connected objects in only one place. To this end, in this chapter, we will study and analyse the existing literature related to our subject in order to understand the advances already made and to learn from the lessons and mistakes made in the past.

As a first step, we will look at a recent definition of what an IoT middleware is and its fundamental principles. We will give examples to illustrate our findings and draw the first conclusions related to implementations of our prototype based on our new knowledge. Secondly, we will review the progress made recently in the technologies that will enable us to implement our solution such as augmented reality or object identification among others. To conclude, we are going to state the system requirements based on our findings in this chapter.

2.1 IoT Middleware

First and foremost, it is important to understand what an IoT middleware is and what its main characteristics are. In a recent article [16], Bansal and Kumar define IoT middleware as being "*a component of IoT ecosystem that provides the raw data to the user in form of web application. The information gathered from IoT devices must be manipulated in a form such that it can support all types of applications related to the IoT ecosystem*". According to them, "*an IoT middleware should be designed in such a way that it should be scalable, adaptable, flexible, secure and open source. This kind of middleware would enable researchers and developers to configure and compose new applications and also introduce new IoT devices into the ecosystem and would avoid low-level reprogramming of complete IoT ecosystem.*" The figure 2.1 bellow shows the place of an IoT middleware within the IoT systems architecture.

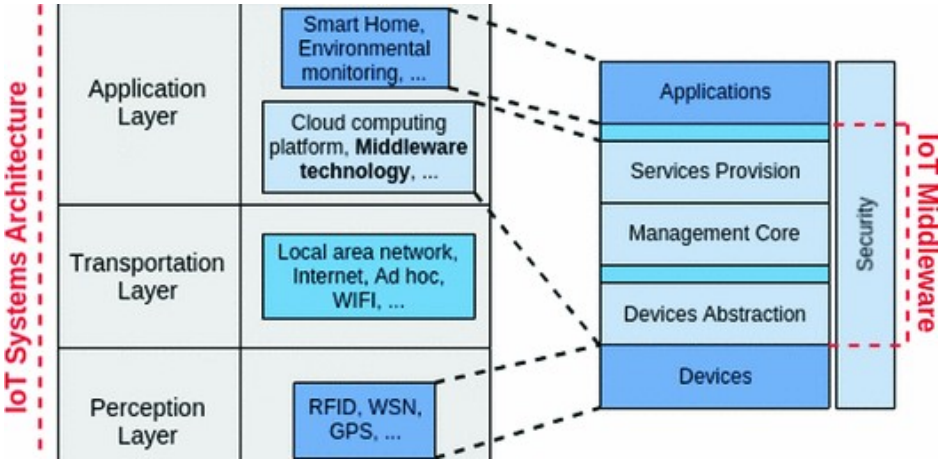


Figure 2.1: The place of an IoT middleware within the IoT systems architecture

They categorise the middlewares in three categories based on usability, flexibility and adaptive nature.

1. Service-oriented Middleware: "*This kind of middleware enables end-users and developers to add and modify IoT devices into IoT ecosystem as services. It provides services like access control, storage management, event processing engine. Security models are costly in terms of time and memory so they provide limited support in terms of security, privacy and trust. In such a system, the control of user data is limited.*"

2. Cloud-oriented Middleware: "*This kind of middleware enables the collection and interpretation of data with ease. However, it limits the growth of the ecosystem in terms of types of IoT devices. The security model in cloud-oriented architecture is defined by the cloud which is being used. So, privacy and security are defined by the cloud system and is not configurable by users. The major concerns in such a system are the control of sensitive data and such system are also not designed to support constrained resources.*"

3. Actor-oriented Middleware: "*This kind of middleware is open source and is designed on a plug and play model. IoT devices can be added to the IoT ecosystem as a plugin and when an IoT device is not required, it can be easily removed without impacting IoT ecosystem. The security model here is configurable by users through plug and play mechanism. Actor based architecture is designed to support constrained resources.*"

Many middleware solutions for IoT have been studied already, here are the most promising ones within each category:

Service-oriented Middleware

- **Hydra** was a European Union funded project developed for networked embedded systems that is now called LinkSmart [17, 18, 19]. In this middleware, the affordances of the IoT devices are described using ontologies in OWL. Even if this solution provides secure and trustworthy devices and services though distributed security, it is not suitable for consumers. LinkSmart applications must be custom-made by programmers and are then more fit for enterprise-level IoT applications.

- **Global Sensor Networks** (GNS) is a middleware that aims to provide flexible integration, sharing and deployment for IoT devices using XML-based object descriptors [20]. Using a virtual sensor abstraction, GNS provides a full set of functionalities for the management of the sensors (from security to event processing to presistency and more). In addition to the virtual abstraction, GNS also provides access to SQL-based database if requested. However, adding new sensor types using GNS is tedious, since we have to write an XML descriptor. Similarly, it is hard to compose or interpret the sensor data. For that reason, GNS was extended to a version called XGNS (eXtended GNS) [21] that provides, basic, composition capabilities thanks to its distributed nature.

Actor-oriented Middleware

- **ROS** (Robot Operating System) [22, 23] is an open source software that comes from the robotics domain where it is very popular. ROS adopts a publish-subscribe information distribution and a peer-to-peer communication system which allows information to be propagated by multiple nodes and thus reduce the fault occurrence. This contributes to a high-level hardware abstraction that permits devices to be easily integrated into the ecosystem. Although adding devices into the ecosystem requires programming skills, ROS supports multiple programming languages (C++ and Python). ROS stands at the limit between actor-oriented and service-oriented middleware. With some adaptations it could be used connect a big number of nodes and allows machine to machine communication thanks to its publish-subscribe architecture. However, ROS is not backwards compatible with existing objects (install a huge piece of software on them) and lacks of privacy and security management.

- **Calvin** [24] is a light-weight and portable middleware developed by Ericsson. It uses the abstraction of actors which is based on the Asynchronous Atomic Callbacks pattern to build application. The actors represent some devices and/or services in the form of reusable pieces of software that can react to input and thus produce output based on a predefined comportment. Calvin's reusability properties make it easy to build and manage applications once developers get used to the platform. However, Calvin is developed to be an industrial solution. Even if some principles and ideas are valuable, this middleware is not made to be extended to a daily and broad use.

Cloud-oriented Middleware

- **Google Fit** is another middleware used in the fitness domain [25]. It centralise the fitness data of a user into a cloud shared by multiple fitness applications. Basically, data from different type of applications are put on the same shared cloud and can be used by all the apps that are a part of the Google Fit ecosystem (e.g. Nike, Adidas or Polar). The users can select the security and privacy settings themselves which makes this middleware trustworthy even if, in this kind of architecture, the performances in terms of security and latency among others depends on the quality of the infrastructure of the cloud. A weaknesses of this middleware is that that its application scope is very specific and that makes it hard to generalise it to other fields or add more sensors. This technology is built with the Bluetooth Low Energy (BLE) as connection means at its core and it is very complex to add applications or sensors that do not have this characteristic to the ecosystem in place.

- **Paraimpu** is a prototype of social aware middleware based on WoT principles [26, 27]. It makes it possible to add, use, share and interconnect objects or virtual things like services on the Web through their Restful API. For example, an interaction with a connected device can happen via social networks like Twitter where users use hashtags to communicate with the object. As seen in this example, Paraimpu defines multiple abstractions that allow the user to compose with the data and do mash-ups in order to create more complex rules. However, only a predefined set of sensors and actuators are available for use and it is not mentioned anywhere how the user could add new ones to the set. Some drawbacks are that there is no support for service discovery and no device-to-device communication possibilities which limits the rules and interaction that can be performed.

As we can see, regardless of the category most of the solutions listed above all share similar characteristics with the Web of Things, some even use the principles of the WoT. By considering the WoT as an option for implementing this type of program, we are taking advantage of its benefits. For example its ease of use for end users because it is similar to the regular Web. Also the security concerns are fewer because the web protocols are already pretty secure and well understood by developers. The following figure summarises the situation.

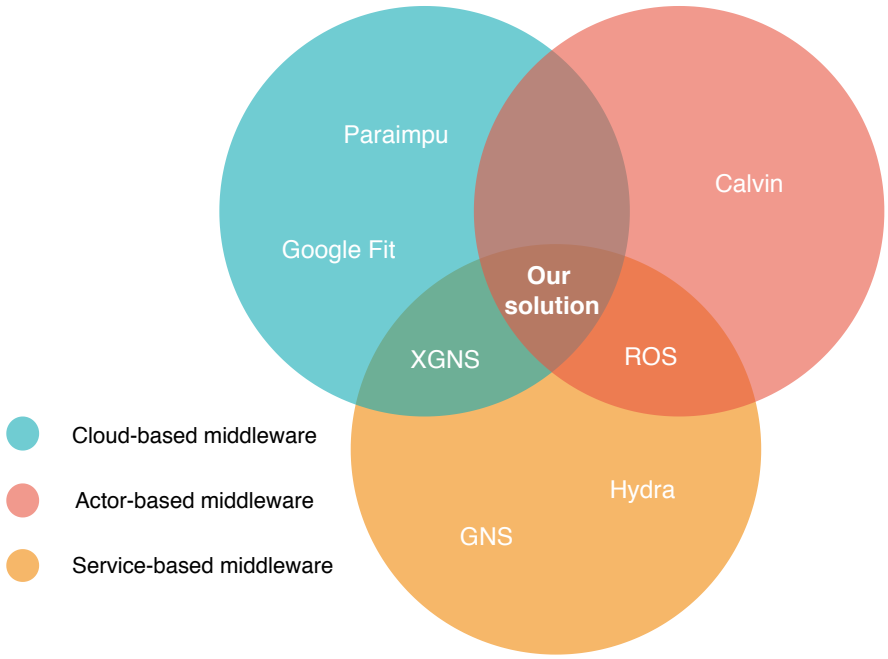


Figure 2.2: Visualisation of the three categories of middleware

We also found two important papers that highlight the combination of WoT and AR to produce an IoT middleware the same way we would like to do it. Even though they do not provide an implementation, we will still consider these papers to be the basis of our work.

In their paper, *Browsing the Web of Things in Mobile Augmented Reality* [28], Zachariah and Dutta discuss the characteristics and services that such a model should provide to its users. After introducing the subject, the authors discuss the potential uses of such application in smart homes, to do a device setup and for quick interactions with ephemeral devices. Then, they propose an overview of the architecture that they came up with for a home environment. They implemented an IOS application prototype using ARkit that uses image recognition in order to identify objects. Their interface is a web page enhanced with AR. When the user activates a light switch, it creates a server-side command that goes to the smart switch and turns on the light physically. They conclude their paper by discussing the challenges and question that should be taken into account when designing a Mobile Augmented Reality application (MAR). This paper was very useful for us because it defines a lot of the requirements needed in such applications and provides a solid foundation on which we could build our work.

In an article that is more illustrated and much more precise in terms of architecture than the previous one, another middleware for IoT prototype combining the use WoT and AR is presented [29]. This one was developed with the WebXR API using a model-driven engineering methodology. In this article, the authors also explain the different stages of the application flow. First, the discovery step is where the application is used to discover the objects that one can interact with. Then, the application will retrieve the object's description. The second step of the flow is the model management. This step consist of the modelling of the object representation within the app in order to have an environment as close as possible to the outside world. The next step are the augmented interactions, when users get to interact with the now discovered and modelled objects in order to get informations about them (e.g. affordances and description). Lastly comes the actuation step where the user is able to change the properties and behaviours of the controlled objects. This paper describes a system really close to what we aim to achieve in our work. However, the authors do not provide access to their source code and there are no possibilities for the user to define rules.

In conclusion, for our prototype's development, we wanted to to create a middleware at the crossroads between the three demonstrated categories. We mainly wanted to come close to the multitude of service provided by a service-oriented IoT middleware, the data management abilities of a cloud-oriented IoT middleware and mostly the ease of use and openness of an actor-oriented IoT middleware. In this regard, we based ourselves on the architecture and principles described in the two previously mentioned articles. But even if those papers were a good foundation for our project, it is obvious that we planed on redefining, improving and adding important notions to what was already developed. In Figure 2.2 representing the three categories of middleware, one can see that our solution is therefore at the intersection of the three types of middleware

2.2 Enabling Technologies and Useful Concepts

We know it now, several concepts will be important for the implementation of our solution. It is therefore essential to review what has already been achieved in each of the useful fields. That way, we will be able to extract the benefits and advantages of each enabling technologies.

2.2.1 Web of Things

As this will be the cornerstone technology in our implementation, it is important to do a small review of this technology. However, we will focus ourselves on the new developments as we already defined the concept of WoT in the introduction.

In their book [30] Guinard and Trifa, who are two of the WoT founders, present the Web of Things in extenso. They review the concepts that enabled and led to the creation and development of the WoT as a fundamental principle of middleware design. In addition, a major part of the book is used to demonstrate to the reader how to build the different layers of the WoT architecture. In these chapters, the authors approach each necessary technology, introduce and explain them in detail so that even a neophyte developer can understand how easy it is to implement a WoT architecture like the one shown in Figure 2.3.

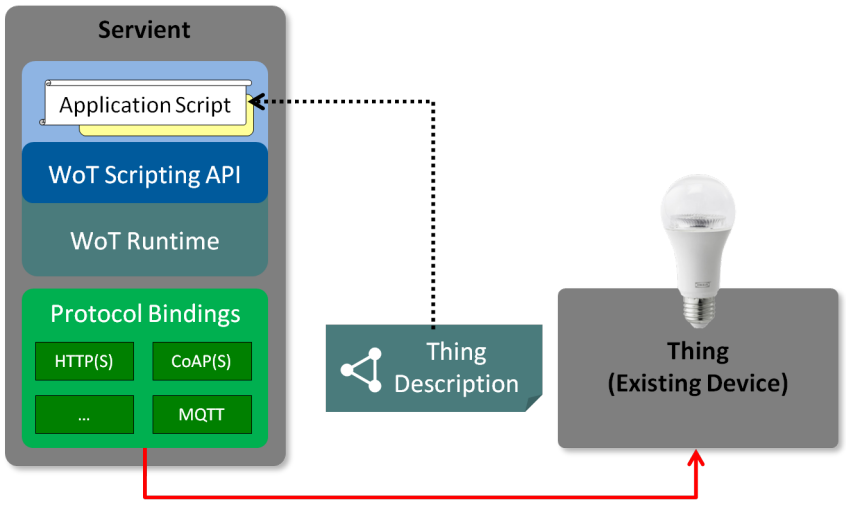


Figure 2.3: Simplified architecture of the Web of Things

As mentioned above, the W3C, the international body responsible for supporting Wold Wide Web technologies, is also responsible for the development of the WoT. Recently, they agreed on a way to represent Things, connected devices, via JSON encoded files called Thing Description (TD) [31]. A thing description contains metadata about the thing itself, information about how it can be used, some fields for machine understandability, and, at last, some web links to express any association with other Things or documents on the Web. Figure 2.4 shows what a Thing Description file looks like.

EXAMPLE 1: Thing Description Sample

```
{
    "@context": "https://www.w3.org/2019/wot/td/v1",
    "id": "urn:dev:ops:32473-WoTLamp-1234",
    "title": "MyLampThing",
    "securityDefinitions": {
        "basic_sc": {"scheme": "basic", "in":"header"}
    },
    "security": ["basic_sc"],
    "properties": {
        "status" : {
            "type": "string",
            "forms": [{"href": "https://mylamp.example.com/status"}]
        }
    },
    "actions": {
        "toggle" : {
            "forms": [{"href": "https://mylamp.example.com/toggle"}]
        }
    },
    "events":{
        "overheating":{
            "data": {"type": "string"},
            "forms": [{
                "href": "https://mylamp.example.com/oh",
                "subprotocol": "longpoll"
            }]
        }
    }
}
```

Figure 2.4: Thing description file example for a lamp

The W3C also works on a Thing Description Ontology [32]. They plan on using an idiomatic RDF axiomatisation of the TD information model to provide an alternative to the JSON representation. This will make it possible to take advantage of the benefits of using ontologies such as automated reasoning or easy navigation between the concepts amongst others. TD Ontology is still a work in progress and is only published as a draft on the W3C website. A lot of work still needs to be done and is being done at the time of writing this thesis. We hope to be able to use this technology to create our prototype.

The WoT topic is very popular lately and one can observe an ecosystem forming around the subject. Over the past year, a large number of articles have emerged. They range from empirical analysis of the current state of the technology [33] to studies that show comparisons between different architectural styles while building WoT applications [34]. There are articles that suggest improvements for example in the way to define how user requested action must be executed [35]. Another article suggest improvements on the way to test a Wot applications [36], there are even some articles that propose to extend the concept of Thing Description to be able to describe not only single Things but also groups of Things [37]. An important community is starting to take shape around the WoT. Similarly to the in 3D printing sector with platforms like Thingiverse, some cooperative platforms/tools are built (for example WoTify or the WoT Store [38, 39, 40]). Finally, keynotes and events are scheduled to discuss WoT related subjects and ideas [41].

2.2.2 Augmented Reality

As said, we decided that we were planning to use AR as a usability amplifier for our application. Its educational abilities and intuitiveness make it an essential technology for this kind of job. Some studies [42, 43] show that AR improves the learning abilities and creativity through interaction, visualisation and intuitive manipulation. It has a positive and direct influence on the state of mind of the users. Also, AR interfaces, thanks to their ease of use, motivate users to concentrate better and makes them want to discover more because they feel more satisfaction. Next, AR improves the way users visualise a problem and thus develops their intuitions. This allows them to better understand the concepts that surrounds them and develop better spacial abilities since they can visualise the different opportunities that are in front of them in a very detailed way. Finally, AR allows the creation of interactions that feel more natural to the users.

Many recent publications speak about the principles and good practices to design AR interfaces. But before we go through them, we are going to look at one of the masters of interface design, Don Norman and his famous instructions on product designing [44]. In his ideas to conceptualise an interface, he insists on six important principles that one has to take into account while designing interactions for technology products:

1. **Visibility**: If a functionality is hidden, it will be hard for a user to learn it and use it. We must decide what to make visible because not everything can be if we want a sober interface.
2. **Feedback**: Give clear information to the user when an action is taken. The user should always know what is happening in order to reduce the possible frustrations.
3. **Constraints**: Possible interaction should be clear otherwise the user may think that the interface has to act in a way that we did not intend ad get frustrated.
4. **Mapping**: Create clear relationship between control and effect for a functionality in order to make the functionality more intuitive to use for the users.
5. **Consistency**: In order for users to build experience while using the interface, similar tasks should have similar elements and operations. For example, the delete button should be the same in all views.
6. **Affordance**: The objects composing an interface have to give clues on how to use them.

A more recent paper focuses on the principle to design AR applications in particular [45]. The authors combined Norman's human computer interaction principles with the demands of AR to create derived principles and heuristics:

- **Affordance**: Is the same thing as described in Norman's work. Developers should use metaphors derived from real-world examples to facilitate the users apprenticeship.
- **Reducing cognitive overhead**: Let the user focus on the task to do and not the controls. Every extra cognitive effort acts as a distraction for novice users. This joins the mapping and consistency principles of Norman.
- **Low physical effort**: Be careful to create an interface that does not wear the user down. No unnecessary interventions and also consider that it is possible that the users might feel sick while in a virtual environment.
- **Learnability**: Again a derivative of Norman's mapping and consistency principles, make the interface easy to learn how to use for the user by using intuitive technique taken from the real world behaviours.
- **User satisfaction**: User engagement means that the user does not see themselves solving a task. A fun application always works better.
- **Flexibility in use**: Take into account different user preferences and abilities. Use multiple modalities that complement each other in order to be able to adapt to individual user preferences.
- **Responsiveness and feedback**: Closely related to Norman's feedback principle. It states that a user will not tolerate much system lag. Systems have to be built in a way that keeps the user informed on what is going on. Also, systems have to be designed in a way that optimises their performance and or power consumption. For example, if we know that we have small computational power, we could only render 2D objects to reduce the calculation costs.
- **Error tolerance**: Ensure that the application is robust and foul proof so that the user experience is more satisfying. This could be achieved for example by implementing redundancy.

There are a lot of different input modes that can be used in our prototype. The default input interfaces for each support like the touch screen for phones and tablets, mouse and keyboard for computers and a controller for the head mounted devices. Gestures can also be used as an intuitive way for the user to interact with the augmented environment. However, it require powerful algorithms to be interpreted otherwise only a few of them can be used which reduces the interaction range of the user. Eye tracking or gaze is also a renowned input method although it requires to add hardware to be able to track the users eyes and a good computational power to be able to interpret it. Speech is a good way to interact with our system too, nowadays, there are many techniques that allow vocal recognition in a light and efficient way.

Even though, it is important to implement a system that provides multimodal interactions [46] and make sure that it respects the social affordances or does not wear out the user. The main contribution we want to do in this work is more about the way to implement such application rather than the way to optimise the interactions possible with it, so for the time being, we decided to only implement one input modality (the default input interface) and leave room for improvement in this area for later.

2.2.3 Tools to Create Augmented Reality Applications

Now, we will take a look at the frameworks that are available to create AR applications. There are lots of frameworks that exist and can be used to design AR applications but the most popular ones are listed bellow:

- **Vuforia** is a software development kit (SDK) that enables its users to create mobile augmented reality (MAR) applications [47]. It uses computer vision for object recognition and tracking and allows cross-platform development. Unfortunately, this solution is not free of charge and since we want to create a prototype that can be used as a basis for further studies, we will use free solutions to facilitate the reuse of the principles developed.

- **Wikitude** is another MAR SDK that is renowned for its recent object recognition and tracking technique called SLAM (Simultaneous Localisation And Mapping) [48]. It is cross-platform and available for Android, iOS, Windows and diverse head-mounted devices. The main problem with this SDK is that not free to uses and requires a lot of money to be used.

- **ARToolKit** [49] was one of the first SDK to develop MAR applications in C/C++ and is still around today which is a proof of its high quality. It uses computer vision to draw the virtual objects. It allows to create cross-platform applications on Linux, Windows, OS X, iOS and Android. But most of all, it is open source and free-ware which makes it a top contestant for us.

- **ARKit** [50] and **ARCore** [51] are also widely used alternatives but do not allow cross-platform application development. They are to create MAR applications respectively on iOS and Android. They are free to use and provide a lot of features. However the fact that we would like to create a cross-platform application means that we will not use either of them.

But even if ARToolKit fulfils a good part of our expectations, there are better solutions in our opinion. WebXR and other lighter libraries such as AR.js or OpenCV are the ones we will be using depending on what we will decide to display. Those have the advantage to be very easy to use and to use very few computational power compared to the toolkits exposed above. Moreover WebXR, for example, is a W3C standardised API that allows one to create MAR cross-platform applications on the web for free. It is using the web protocols and programming languages that are much easier to master than C/C++ for example. Since it works on the web, it will be available for any operating system or device that has access to a web browser which will make our application more portable than with any of the tools mentioned above.

Tool	Cross-platform	Free to use
Vuforia	yes	no
Wikitude	yes	no
ARToolKit	yes	yes
ARKit	no	yes
ARCore	no	yes
WebXR	yes	yes
AR.js	almost	yes
OpenCV	yes	yes

Table 2.1: Solutions to create MAR applications

2.2.4 Object Identification

If we want to control the objects, it is necessary to be able to identify them, not only the brand and the features, but also to know how to dissociate two instances of the same model. There are a lot of methods possible but some are really pertinent to use in this case. Since the objects that we target to use will not all be stationary we cannot take into account the use of the position of an object to identify them.

Image recognition follows the principle of "What-you-see-is-what-you-get". It uses the camera to capture the image and different recognition algorithms to analyse it to recognise certain objects in it. Even if, with well-trained algorithms, this method produces excellent results, it is too complex and requires too much computing power for us to be able to use it. Also problems could have occurred when having to distinguish between two instances of the same model (for example with two instance of the same connected lightbulb, how to tell one from the other?)

There is also the RFID (radio frequency identification) technology but it would require hardware to be added and we wish to only use a web browser no matter what platform will run the application.

The best solution that we found is using fiducial markers on objects. In the same way that barcodes are unique on every product that can be purchased, we will place fiducial markers on the connected objects, which will allow us to identify each object and to differentiate each instances of a same object. There are multiple fiducial markers available: QR Codes/datamatrix, ReacTVision markers, ARTags and many others. Among the solutions available are ArUco Markers [52] which in addition to being robust to occlusions, allow to retrieve their position relative to the camera in six coordinates (three for translation, three for rotations) which can be useful in the next section of this report.

As an example, we provide the Figure 2.5 that shows a screenshot that we took during the tests we made in order to plan what to do with the ArUco Markers, the top left panel shows an ArUco Marker being detected by the application who adds a red border around it. The top right panel shows that after scanning the Marker, we can add some images on top of it. And the two panels on the bottom show two different method to estimate the distance between the Marker and the camera.

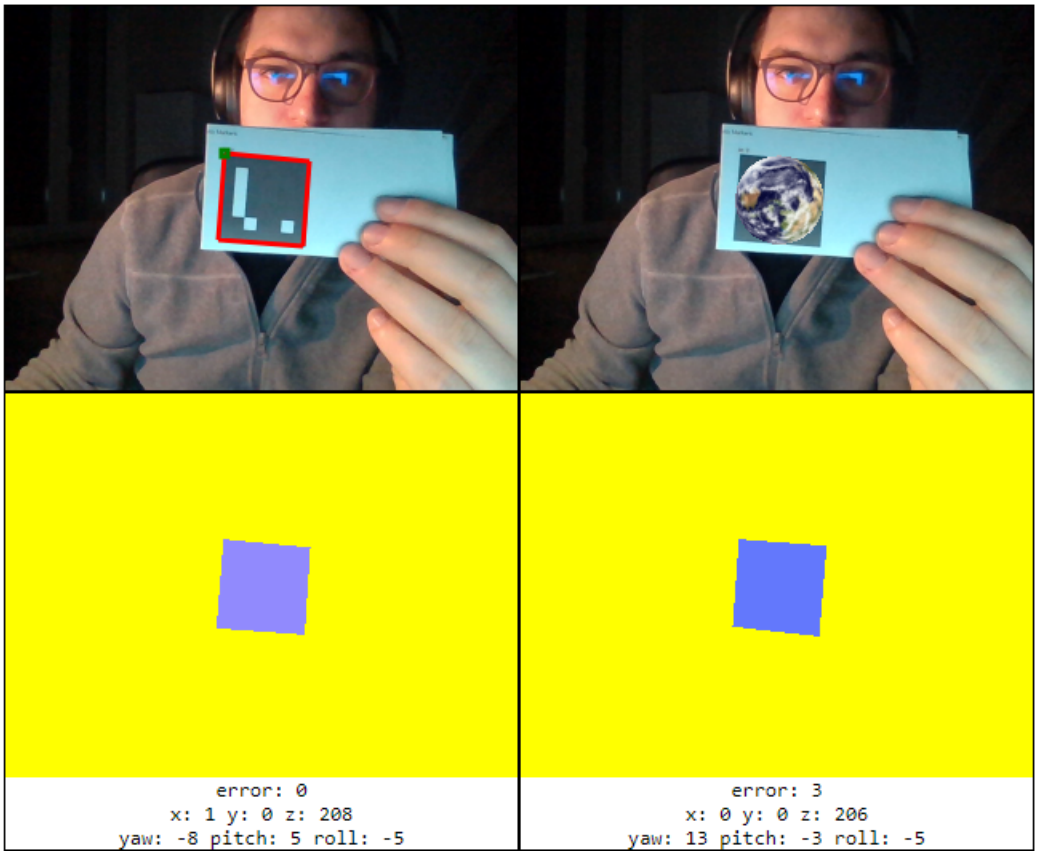


Figure 2.5: Example use of an Aruco Marker

2.2.5 Object Positioning in a Room

This last object identifier method allows us to bounce back on another important feature for our prototype. Being able to know the position of the objects in a room/home is useful because we need to be able to show the links between objects who are part of a common rule and thus be able to show the user where they are. The position can also be used to allow users to discover objects they did not know about. They are different techniques of indoor positioning available.

Visual positioning is one of them, a good example of it is the Simultaneous Localisation and Mapping technique (SLAM). It uses a camera or lasers in order to create a 2D or 3D map of the surroundings [53].

Another method is radio frequency (RF) based positioning. Using RF receivers and/or transmitters, we create a mapping of the surroundings. The RF signals used can vary from WiFi to Bluetooth to different types of beacons [54, 55, 56]. However, RF-based positioning need to be coupled with some models to estimate the distance between a receiver and a transmitter because we want to be precise, knowing that a device is close enough to receive data is not sufficient we need to be able to quantify it. We will not discuss those methods yet because in the first phase of prototype implementation we will use Aruco markers to deal with the object position. Later, we will try a visual positioning technique because it fits our architecture better that RF-based methods. As a matter of fact, we will already be using a camera to integrate the AR part of the architecture and therefore we will not have to add hardware to our solution.

2.2.6 Internet of Things for Smart Homes

Our solution will be designed for non-expert end users so it is important to know the domains in which their connected objects are or will be used in order to be able to understand the context and produce a good control interface for them, in our these we will focus on smart homes.

IoT in smart homes can be used to manage and monitor the home appliances (e.g. DVD, TV or air cooling system). Researchers want to provide a more comfortable environment for the home's inhabitant [6, 57].

It can also be used to monitor and improve health condition of the home's residents. Brazilian researchers discussed the use of images and emotion recognition, using video sensors and decision makers, to monitor patients that are suffering from illness like depression in their homes [7].

Another good use of IoT devices in smart homes is to improve energy efficiency and resources management. A lot of recent papers define architectures and implementations of systems that enable reductions of energy consumption and carbon footprint thus increasing the energy efficiency of the house [10]. Other studies discuss about how to design a household waste management system based on IoT technology [9, 58].

Finally, other subjects are also trending, such as building security systems for smart homes [59] and many others that are beyond the scope of our study.

2.3 System Requirements

Being aware of the previous progress made in our field as well as the tools available to design our solution. It was therefore important to proceed, before starting the implementation of our prototype, to an analysis of the requirements. It is well known that clearly defined requirements increases the success rate of any project.

The main requirements were defined according to what we want the application to enable its users to do. The core idea of this thesis is to use Web of Things principles and augmented reality to create a way for inexperienced users to connect, control and interact with smart objects through a single, easy-to-use and intuitive interface. Users will also need to be able to create rules that combine several actions and behaviours between connected objects in their ecosystem (we hope to be able to implement this functionality in time). An example of a very simple rule could be that when the TV is turned on, the lamps and shutter mechanisms are activated to dim the room and provide a better experience for the user who only needs to set this rule once. Also, the solution will have to be available on several platforms, we are targeting among others mobile phones, computers and augmented/virtual reality headsets.

There are also some requirement less linked to what the system should enable its user to do but more about how it should feel. First, it should respect the principles explained in section 2.2.2: low physical effort, learnability, user satisfaction, feedback and the others. It is also important that the system does not require more hardware than needed which is a camera and the necessary to run a web browser. This will allow our program to be used from a larger number of different devices. Another important point for us is to make sure that our middleware is downwards compatible, meaning that it should allow interoperability with connected objects designed before its conception.

A middleware is generally used to abstract behaviours that make a protocol/processes too difficult to perform otherwise, in our case we want it to allow to abstract for the user the hardware inputs and outputs. We also want it to abstract manufacturer-provided hardware and software interfaces, every interaction possible with an object should be doable through the interface of our middleware. Most of all, when using our application, the users should not worry about the data streams management, given that this is also an abstraction that our system has to provide.

Overall, particular attention will be paid to the users of the system and their satisfaction. In this case, the term "user" also includes the people who will install the system and maintain it. For this reason, the system should be easy to install in a new environment and its maintenance should as uncomplicated as possible to carry out. We would also like to involve the user in the design and improvement of our platform by encouraging community spirit and open source mentality. The user will therefore have an essential position in the implementation of the downwards compatibility requirement. Indeed, we rely on users to create and add new thing descriptions for objects that were designed before the creation of our platform. Therefore, a functionality has to be developed to promote the sharing within the community.

3
Solution

The last step before actually starting the implementation is to define the architecture and conceptualise each functionality of our prototype. The content of this chapter therefore goes beyond the requirements, to explain how we conceptualised the functionalities and crafted the strengths of our application thus explaining why our work is innovative compared to existing solutions. As a first step, we will once again define what we are aiming to achieve in order to clarify things one last time before the conceptualisation stage. Then, we will take each research questions listed previously in Section 1.2 one by one in order to explain our vision of things and how we plan to conceive their answer. Then, to conclude this chapter, we present an overview of the proposed architecture. As mentioned before, the aim of this work is to create one of the first cross-platform IoT middleware available through the Web. But what does it mean and why is it important to do so? Well the answer to that question is very simple. IoT middlewares that allow us to interact with all sorts of connected devices are very rare and often of poor quality. Most existing solutions are very difficult to use for inexperienced users. Also, few of them allow to control any kind of object because they have been designed in a specific context. Even if they are limited now, we believe that this kind of program is part of the future of connected objects. It is unthinkable that we will have to continue to use the merchant's applications to control every object around us. A unified solution must be found to avoid chaos in a world where the number of connected devices is constantly growing.

Now, let us take up the research question (RQ) again to explain how we plan to answer it:

RQ1: *How could we use the Web of Things principles to create a powerful cross-platform, easy to use application that enables the interaction with any smart objects?*

This is what we expect to be our most important contribution. The WoT principles are standardised by the W3C and as part of those principles lies the Thing Description (TD) files, a simple JSON encoded file that describes an object and its properties. This is where our application allows us to interact with any smart object, we just need their TD. Since the information those files contain is both human and machine readable, they are easy to produce if we need them. The Web of Things also implies that our application will run using web protocols to interact with objects. This means that if we build our application to be available on each existing web browser then it will be cross-platform. In short, our application will use the information embedded in the tags of the TDs to produce web-pages that allows the user to interact with the objects. We decided to call this part of the implementation the parser. As one can see in the Figure 3.1 bellow, our middleware is composed of two distinct programs the client application and the server.

For some people the line between middleware and simple application remains blurred. Here are the reasons that make our prototype a middleware rather than a simple application. Firstly, our system perfectly meets the definition stated in Section 2.1. The solution we have designed is indeed scalable, adaptable and flexible. And it allows the introduction of new objects without low-level reprogramming. Our system creates a device abstraction for the users, allows them to manage their devices in one place like no other. Moreover, it offers the opportunity to use some additional services such as an overview of all objects and, in the future, the creation of rules between objects or the object's location retrieval and many more.

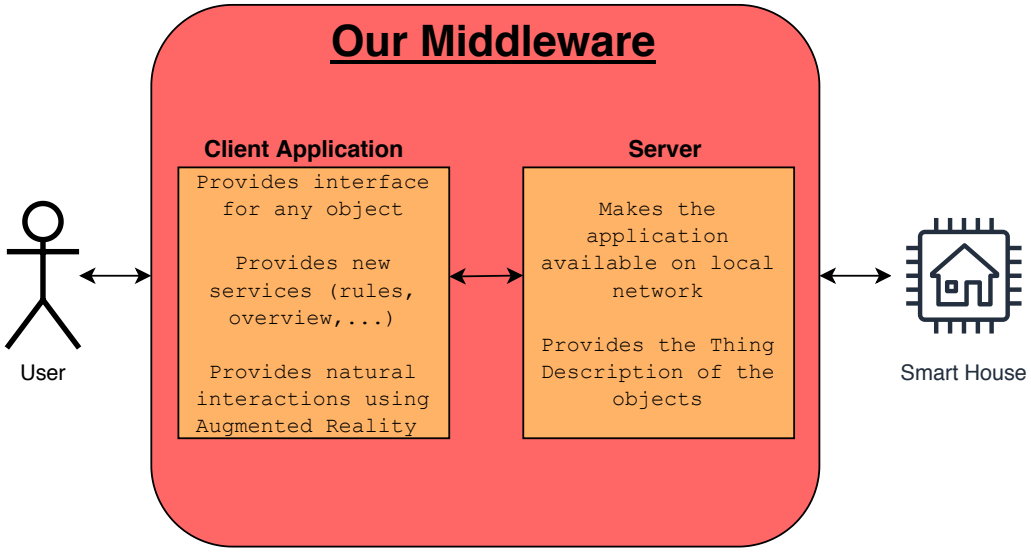


Figure 3.1: Structure of our middleware

Finally, let us go back for a moment to the fact that we decided to develop our middleware as web application. First of all, doing so allows us to easily create a cross-platform system because no installation other than a web browser will be necessary. Also, it should be easy to use for anyone since it is based on knowledge that the users already have, the use of a common web browser. In brief, we make the connected object appear as a web page to the users. Indeed, in our middleware, the users will interact with their connected device through a web page that has the same characteristics as any other. To do this, we need a server that provides these pages on the local network of the user. We provide the pages only on the local network because we have decided to stick to the smart home context.

RQ2: *How could we use Augmented Reality to create natural ways to interact with connected objects using such an IoT middleware?*

AR is known for its teaching properties and intuitiveness. We think that a simple interface to select the object that the user wants to interact with is enough to do the trick and help any user to interact in a more natural way with their connected devices. By having an interface as simple as possible, we avoid that the user gets confused, augmented reality interfaces are not yet widespread, so a period of acclimatisation is necessary. The client application's main view will therefore be composed of two buttons and the users will have to point their camera at the object they want to interact with. Then, once the application has identified the object, the users will be able to interact with it. So, we plan on using ArUco markers as identification means for the objects. In addition to being a simple marker, it also makes it possible to calculate how far away it is from the camera and therefore allows us in the future to add a localisation feature to our middleware. For this to work we assume that, as in Figure 3.2, each object is equipped with an ArUco marker visible enough for the application to scan it.

Figure 3.2: Example of connected object bearing a marker

RQ3: *How to design a lightweight middleware that generates the control panel of any intelligent object?*

Since the answer to the first RQ is already explaining the control panel generation part of the question, here we want to insist on the lightweightness part. It is really important to achieve, given that our application can be launched from a large number of devices of all kinds having necessarily different computing power and energy consumption. For this purpose, since we are producing a web application, we will try to use as much pure JavaScript as possible and limit the use of libraries that could slow down our prototype, waste energy or not be available on every device.

RQ4: *What are the best tools currently available to achieve the needed tasks?*

The answer to this question can be found in the next chapter, which explains in detail the implementation choices we made and consequently the tools we used for each part of the prototype.

RQ5: *What is the best architecture for our solution?*

A lot of work has to be done to implement our prototype while respecting the constraints seen in the previous questions. The architecture we came up with is composed of three main parts that communicate to offer the user a pleasant service and experience. First, there is the augmented reality interface where the users select the object they want to interact with. Then the information of the object is transferred to the server that relays the information received through the augmented reality and the control interface produced for it by the parser. The server also allows the users to access the web pages of the application on their local network. Finally, the user can see the dynamic interfaces created by our parser and through them interact with their devices thanks to some hidden HTTP request. Figure 3.3 shows a schematic of the workflow of our application and the interactions between the actors:

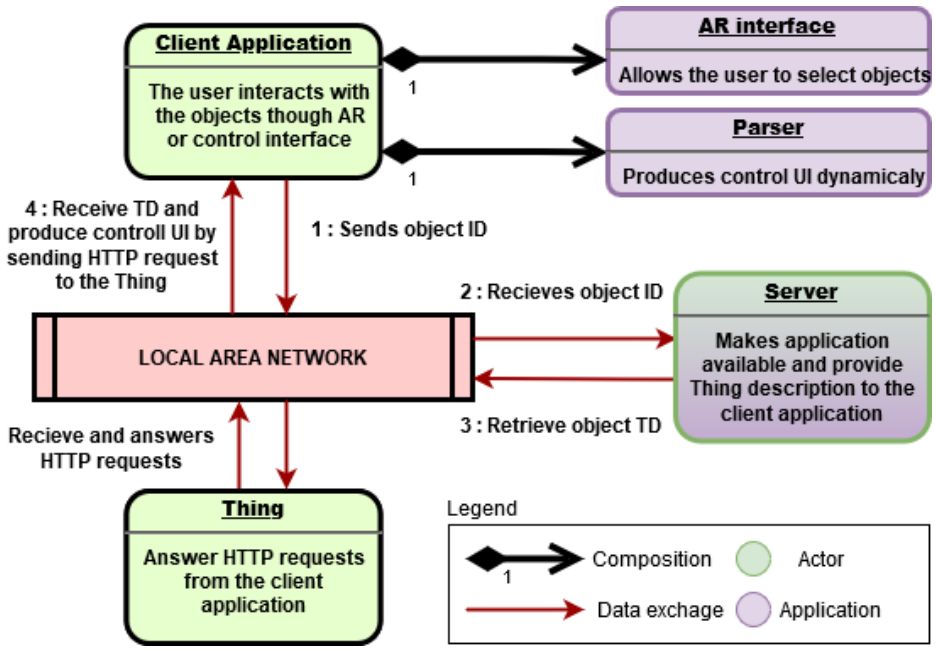


Figure 3.3: Schematic of the prototype's workflow

4
Implementation

In this chapter, we explain the design choices we made and the way we implemented our prototype. As explained in the previous chapter, our prototype is composed of three important parts, the augmented reality interface, the object control interface generated by a parser on the basis of the Thing description of the objects and the web server on which the application is running.

We start by presenting the design and implementation of the parser that creates the object's control interfaces. This is the most essential functionality of our prototype as it allows the users to communicate and interact with their connected objects on a local network. Then, we explain the implementation of some augmented reality functionalities in our prototype in order to allow the user to interact with the objects in the most natural way possible, by pointing at an object to get its information. Finally, we detail the implementation of the web server and the last components needed to provide a smooth and pleasant experience for users.

4.1 Control Interfaces

This part of the implementation is dedicated to the creation of control interfaces for the objects with which the user would like to interact. As of today, there is a huge amount of connected objects, and according to Gartner [60] there will be 25 billion connected things in use by 2021. This is why it is important to build a generic solution in order to be sure to be able to generate the interface of any object. The idea is to use information offered by the manufacturers or a reverse engineering community to create and share it with the world. As we mentioned earlier, we use the Thing Description of the objects to do so. Our control interfaces are generated using a parser that goes through the TD and generates the fields and buttons needed for the user to get information on the object and interact with it.

In order to explain how our parser works, it is necessary to explain in more detail how a Thing Description document is structured. As we know, TDs are documents encoded in JSON format which makes them easy to read and edit for humans but also easy to parse and generate by machines. These files are all structured in the same way, they contain tags that represent information about the object's interaction affordances such as its properties, the actions it can invoke and the events it can communicate. Our parser will go through those tags to generate the interface of each object. Appendix B contains the list of tags that can be present in a TD.

Our parser consists of pure JavaScript code because it is necessary to keep the cross-platform abilities[1] of our application and make the application as light as possible. As a result we use as little as possible existing libraries, tools or programming languages. As said above, our parser scans the information (TD) sent by the server in order to find the necessary tags to generate the interface. Those tags are *title, actions, properties, events* and then other tags that decribe in more details the actions, properties and events such as *description,units,forms,href and others* The scanning operates thanks to the JavaScript native function called: "*JSON.parse(data)*". It allows to create an object that acts like a dictionary with keys and values that map the content of the JSON file which lets the parser access the contents of the fields in the JSON by simply doing "*value = object.key*".

[1]We also call that cross-browser abilities since the application will only be running in web browsers.

As seen in Appendix B, multiple tags can be used to describe the behaviour of an object in a TD. That means that the same object can be represented in multiple ways with more or less details such as description, id, multi-language tags and this also leads to the fact that a lot of cases have to be taken into account by our parser. For example, a TD has an actions tag that does not contain a description or unit tags versus a TD for the same object with description or unit tags. Our parser will always be able to produce an interface as long as there are the mandatory tags, an id tag and at least one action, property or event that contains a form tag that points to the data. Here is the minimal structure required to be a "parsable" by our system.

```
{
    "@context": [
        "https://www.w3.org/2019/wot/td/v1",
        { "saref": "https://w3id.org/saref#" }
    ],
    "id": "urn:dev:ops:32473-WoT-1234",
    "title": "MyThing",
    "securityDefinitions": {"basic_sc": {
        "scheme": "basic",
        "in": "header"
    }},
    "security": ["basic_sc"],
    "properties": { //can be also actions or events
        "status": {
            "forms": [{
                "href": "https://example.com/status"
            }]
        }
    }
}
```

Listing 4.1: Minimal structure of a TD required to be "parsable"

Figure 4.1 below is an example of Thing Description parsing using our web application. We will use this figure to show and explain how the parser works in detail for the properties, actions and events tags.

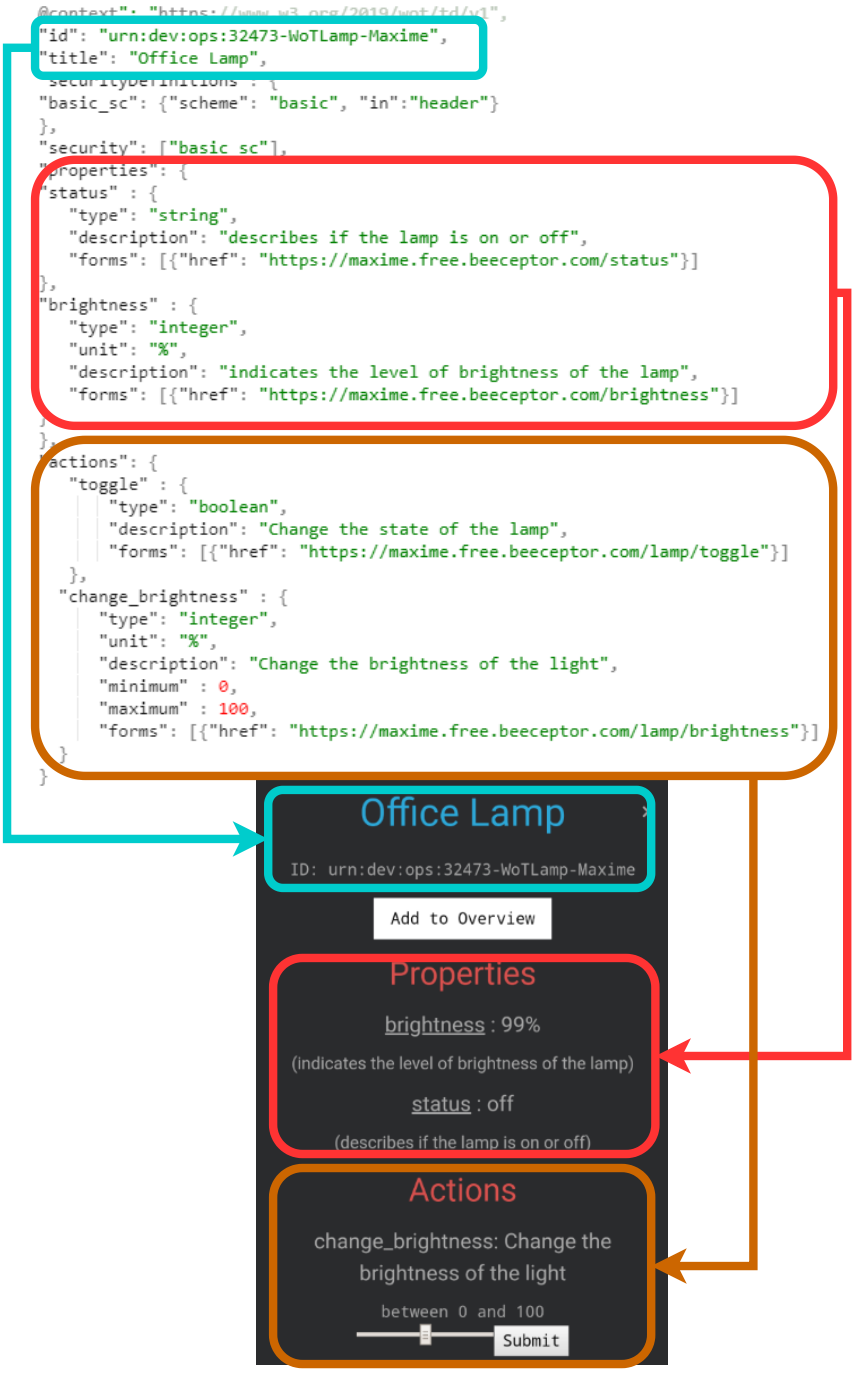


Figure 4.1: Thing Description to user interface: a parsing example

We can see that the parser creates for each property a simple text paragraph that contains the information located at the addresses mentioned in the "*forms*" tag that the parser retrieves via an HTTP GET request. It is logical that the parser produces simple text because the user is not allowed to modify all the properties, those properties represent the current state of the connected object and some of them should not be touched although most of them are available to change through an action (see next paragraph). Very simply, we have also added to the interface the units and the description of the properties if they were mentioned in the TD. Another more hidden feature that the parser provides is that it sets up a long polling system, which is a mechanism to update fields after a given period of time. Thanks to that, the properties displayed on the interface are updated at the slightest change made by any other means than the interface. To carry on with the lamp example, if someone manually turns off the lamp, its status changes to "off" in the UI. Here is the pseudocode of the function called by the parser to display the properties on the screen:

```
for (elems in object.properties) {
  create paragraph for elem
  if (elem has no description tag){
    if (elem has no unit tag){
      add in paragraph: "elem : value"
      setup longpool mechanism
    }else{
      add in paragraph:" elem : value + unit"
      setup longpool mechanism
    }
  }else{
    if (elem has no unit tag){
        add in paragraph: "elem : value <br> (
           description)"
      setup longpool mechanism
    }else{
      add in paragraph: "elem : value + unit <br> (
         description)"
      setup longpool mechanism
    }
  }
}
```

Listing 4.2: Pseudocode for the printProperties function of the parser

For events, our parser will proceed in the same way as for the properties. Although, events differ from properties by the fact that events may be triggered through conditions that are not exposed as properties. For example, an overheating event might be triggered when some parts of the physical device are starting to burn. In both cases we will use the longpooling mechanism to keep the user interface consistent with external changes regardless of whether they are virtual or physical.

To handle the actions, the parser generates a paragraph containing a different input mode based on the type of the action (found in the type tag). Those input means send the user's data through an HTTP post request to the link mentioned in the form tag. The parser recognises the most recurring tags *"integer, number, color and string"* and provides for them robust and foolproof input methods. If there is no type mentioned or the type is not recognised, the parser will display a generic input field and a submit button. The only type not considered yet is *"object"* which could be tackled in a future work. Figure 4.2 shows the input methods used for each types.

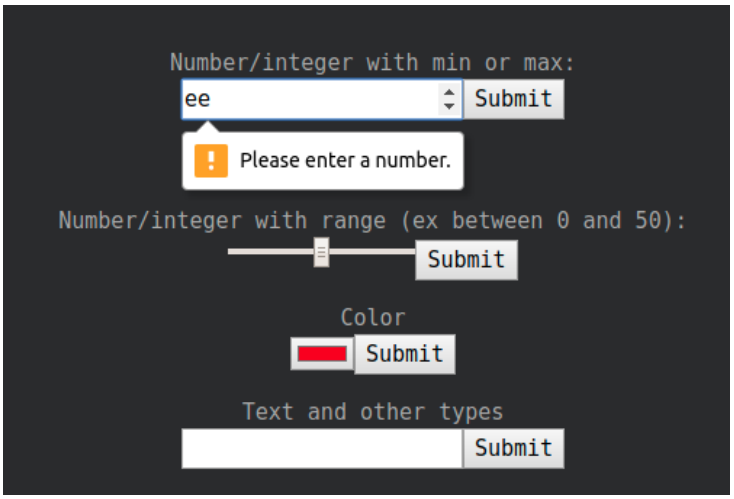


Figure 4.2: Different input methods based on the type of action

Here also is the pseudocode of the function called by the parser to display the actions on the screen:

```
for (elems in object.actions) {
  create paragraph : elem + description (if tag)
  if(elem.type = "integer" or elem.type = "number"){
      display number/range input method
  }else if(elem.type = "color"){
      display color input
  }else if (elem.type = "object"){
    display message : type not taken into account
  }else {
    display default input method
  }
}
```

Listing 4.3: Pseudocode for the printActions function of the parser

4.1.1 Comparison With Existing TD Parsers

As we were developing our parser, we came across a number of other implementations. The major difference between these solutions and ours is that they all use many libraries or programming languages that are not compatible with our desire to produce a cross-browser application. Out of all the implementations we have seen, some of them stood out and inspired us, one of the most advanced was "ThingWeb" which is a project supported by Siemens and the Technical University of Munich [61, 62]. Another one we looked at were the tools provided by Guinard and Trifa in their book [30, 63]. And finally, we also looked at a solution implemented in VUE.js [64] by some of the authors of the second the paper that most inspired us [29]. Even if those implementation have a good quality, we could not take a lot of inspiration from them because the first two solutions listed above are more than 5 years old. And after having discussed with the authors of the last solution, they told us that they were starting over the implementation of their parser with another technology because VUE.js was limiting them in their development.

Another important observation that we made while investigating existing solutions, beside the fact that most of them are nearly impossible to run, is that everyone is using the Thing Description documents differently. The W3C is actively working on TDs, thus there are often changes in the standard and, over the years, the way of using the TDs has changed which would explain in part our observation. We also think that the way to use TD is subject to many interpretations. If we take the "Properties" tag for example, we decided to consider its content as read only, however some other developers allow the user to make changes in the generated part for the properties, who is right? As we read the documentation [65] on the Properties, we can see "*the state exposed by a Property MAY be updated (writeable)*" but it is not really mentioned if this property should be updated via a specified action in the Actions tag or through the Properties interface. This is only one example out of the many we observed, this means according to us that W3C should be less vague about what they mean and provide a better view of the changes made from one version to another. Such adaptations would, according to us, allow the people who are reading the literature to better understand the concepts and how they evolve over time. It would also result in fewer groups proposing changes and people being able to discern changes that deviate from what the W3C has accepted. For example, Mozilla decided to add a tag called "links" to help with reaching the content of a property [66]. But the W3C has not ratified as of now, hence, people might use it and then have to change later to continue to respect the standard.

4.2 Augmented Reality Functionalities

In this section, we will explain the resources deployed in order to create an interface containing augmented reality features that enable more natural interactions. Initially we wanted to use WebXR as a tool to generate our AR interface but soon we noticed some problems. In the first place, the implementation concepts for WebXR are still being developed by the W3C and their documentation is very limited and therefore, in our opinion, did not allow us to do a satisfactory job. Also, as the documentation is poor, it is very hard to see what is possible or not with this technology. For all these reasons, using WebXR in our prototype seemed to us too complex to achieve the minimalist interface we designed. Although, we still think the WebXR is a good candidate for future developments. Since we decided to create a very simple AR interface, it was obvious for us when we discovered it that using the OpenCV-based "aruco.js" library was the best choice. Not only does it allow us to introduce our object identification tool, the ARuco marker, into the code. But because it also is available for any browser and contains the necessary functions to display simple objects in augmented reality. By reusing found examples and modifying them as desired, we managed to set up the interface we had imagined for our prototype.

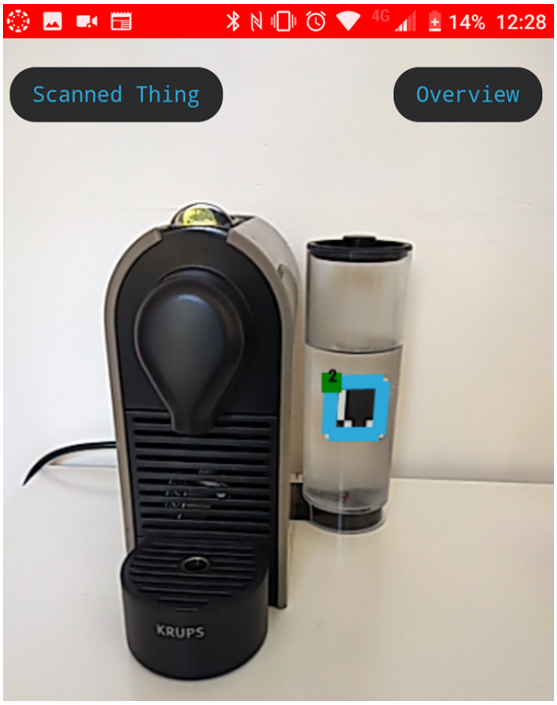


Figure 4.3: Screenshot of the AR part of our interface

As one can see on Figure 4.3, when the users points their camera at an object and the program detects a marker, a coloured border appears on the screen to notify them that the object has been scanned. Along with this border comes other simple information such as the marker's id. This functionality has been programmed only using the "aruco.js" library and some basic HTML/CSS to build the web page. As mentioned in the previous paragraph, for a solid start, we have taken the code from the examples[2] provided by the developers of the library in question. And then, we modelled and enhanced it to get to the result we wanted.

As previously mentioned and shown in Figure 4.3, we have chosen to keep a very minimalist interface. Although Augmented Reality is more and more present in our society, we believe that since our application is intended for all types of users, it is better to reduce the number of AR interactions to its strict minimum in order to allow everyone to adapt to AR interactions. Often, simplicity implies ease of use and clarity. We want the users who have our application in their hand, to directly understand that they have to scan the object to be able to interact with it. But simplicity also implies that the users are limited in the actions they can perform. This is why, in the future, we would like to develop more functionalities while keeping the simplicity of use that currently characterises our prototype. To do this we have found a very light and fully cross-platform library called "AR.js" that allows the insertion of simple objects in augmented reality. It would allow us to, for example, give the user the position of an object in the window using small arrows.

[2] `https://github.com/jcmellado/js-aruco/tree/master/samples`

4.3 Server

We will now explain how we were able to link the two implementation parts explained above. Since our application is a web application available on a local network, it was required to build a server that would make the necessary web pages available to the users. Of course, the server also makes the link between the AR part and the control interface part. Indeed, when the user scans an object, the application sends the detected id to the server, which responds by sending the TD of the object in question which allows the application to produce the control interfaces dynamically. In the WOT architecture proposed by the W3C there are several ways to retrieve the TD of an object, we have chosen the one that centralises all the TDs on a database that the server queries. But it is necessary to know that there is also the possibility, if the object allows it, to retrieve its TD by sending a simple HTTP GET request to the URL of the object which retrieved via the server by sending him the id scanned. In this case, of course, the server would know all the objects connected to the local network.

Our server is programmed in Python using the Flask framework which allows light and simple web development. Flask allowed us to build our prototype on a solid foundation, its flexibility has allowed us to develop the TD provider and web page publisher functionalities in very few lines of code. In the best case, this server would be launched on a micro computer such as a Raspberry Pi connected to the local network so that the user does not have to worry about it. The sever functionalities are very light at the moment, it is not linked to a database to retrieve the TDs, we hard coded them into the server for our prototype. Even if we had to add this feature, the server would still be light since it provides only one web page with a limited number of users. The conditions of use of our applications make it easy to run this kind of server on a machine with small computing resources.

We tested the application on different browsers, different devices and different operating systems. The only thing to do before the first use is to change the parameters of the browsers to accept the requests from insecure origin by specifying the IP address of the server. For example in Chrome, we need to go to `chrome://flags/#unsafely-treat-insecure-origin-as-secure` and add the server's IP address in the accepted origins. This is obviously a temporary solution, which should be improved if we want to make our application public. Appendix C contains a table that shows all the browser that we tested and from which device, feel free to contact us if you want more information.

4.4 Rest of the Implementation

The last important feature that we have not mentioned yet is the overview creation. Simply, when users scan an object, they can add it to an overview. This list of objects allows users to interact with any object it contains without having to scan them again. To implement this feature, we simply add the TDs of the objects selected by the users to a list. Then, when users want to access the overview, our middleware creates the interface based on the contents of this list.

As explained above, we decided to make our application as simple as possible but as Leonardo Da Vinci said "Simplicity is the ultimate sophistication". During the creation of our prototype, a lot of thought were spent to design a pleasing user interface that is within everyone's reach. A special thought has even been given to the colour code used within our prototype. We decided to go for a dark theme because it is usually less energy-consuming[3] and better for users eyes. Moreover, the colours we chose are easy to distinguish for colourblind people. The figure 4.4 below shows the colour that we use in our middleware:

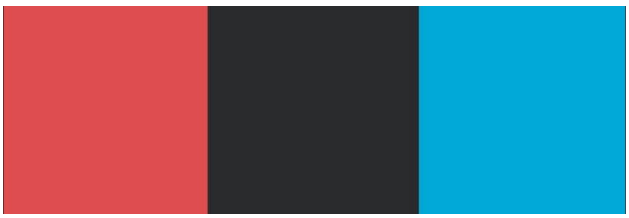


Figure 4.4: Colour code for our prototype

Additionally, we had to be particularly careful to use programming techniques that kept the cross-browser our application cross-platform. Each time we found tools that corresponded to our needs but were not available on all devices. We also wanted to produce interfaces that are responsive and adapt to any screen size since the devices on which our application will be displayed are multiples.

[3] `https://bit.ly/3cSGjDJ`

There is one feature that we had no time to finish, but an initial idea has been imagined. Since the beginning of this thesis report, we have mentioned a number of times the possibility for the user to create rules that combine several devices. For example, linking an alarm clock and a coffee machine so that they turn on at the same time in the morning. The initial idea to do this is to change the TDs on the server and add a "rules" tag that specifies the details of the rule and the objects involved. The parser will also have to undergo a small modification so that it can interpret and implement these rules. As we said, this is unfinished work. Unfortunately, the backend and authoring part of the rule feature implementation is still to be done and it might also be necessary to reconsider the original ideas. Some aspects such as the authoring or the fact of changing the TDs in the server can be reviewed to find a more suitable solutions. In any case, we are happy with completion of the visualisation part. When two objects are part of the same rule, their markers are surrounded by a border of the same colour. Figure 4.5 shows how the interface looks in such configuration. We think that the absolute best way to represent the rules visually would be to add to our visualisation the functionality to locate objects and show their direction to the users.

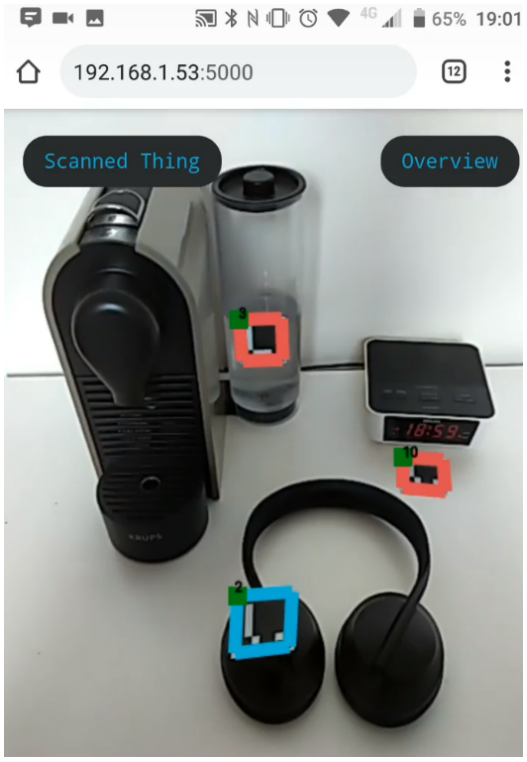


Figure 4.5: Example of multi-rules visualisation

We also spent some time producing an object connected by ourselves using a raspberry Pi. Unfortunately, we are writing this thesis in times of crisis, the COVID-19 virus is making it very difficult to have access to the university lab and therefore to the connected objects it may contain. This is due, on one hand, to the fact that the university was closed for a while, but also to the fact that we did not want to take a risk by going out. Also, the postal services are slowed down too and that is why we had to find a hand-made solution even if we ordered a device. Nevertheless, we found this experience interesting and it allowed us to understand the point of view of the manufacturers of such objects. We also had the opportunity to create this time a Thing Description file which is not only theoretical but concrete this time. During our last week of work, we finally received some smart plugs but we did not have time to deploy them in our architecture. However, we are still planning to demonstrate them during the oral presentation of our thesis in order to prove that it is possible to integrate them with our middleware.

4.5 Challenges Encountered

The previous sections do not do justice to the overall work done in this thesis. Indeed, within them, we explain how we imagined and implemented our final solution. At no point do we talk about alternative design, the reasons for our choices and even more importantly about the challenges we faced which are in fact the reason why our system is what it is. The purpose of this section is to explain a little bit about all these things.

For the parser, the biggest challenge was having to use the Thing Description as is, we really wanted to use only what was standardised and avoid adding personal modifications to the concept. This challenge created several problems for us, the most significant one being that since the TDs are not expressive enough when it comes to the information that the object can receive, we had to imagine a way to create a fairly generalist interface even in theses cases. This is why on Figure 4.2, there is the last field “Text and other types”. In the cases where nothing is mentioned about the conditions that must be respected by the inputs, we have for the moment left the user free to choose, assuming that the error message returned by the object would be expressive enough to make the user understand what to write.

For the Augmented Reality interface, the biggest challenge was to find a library that guaranteed cross-platform properties and that consumed very little computing power. As our knowledge of augmented reality theory was limited before this work, it was also necessary for us to educate ourselves on the subject in order to understand the tools of the library we had selected. On top of that, the choices in terms of interface had to be revised several times. Indeed, we had designed other AR interface alternatives, one of them with floating information next to the scanned object. We decided to drop them in favour of what we have now because they seemed to us too dense in information only by showing the users the properties of the object. So we would have had a hard time adding the interface to interact with the object. As we wanted our middleware to be available on any device with a web browser we had to create a responsive interface. The creation of such an interface, which offers a comfortable consultation on screens of very different sizes, is already perilous in most cases but the addition of augmented reality functionalities on top of it made the task even harder. This is why we finally opted for simplicity.

The server has also been the subject of many reflections, it was not clear to us at the beginning if it was more appropriate to access the TDs by the client application (in JavaScript) or by the server (in Python). So we designed both and after some efficiency testing we realised that in general when the server accesses the database, the system is faster. Also acting this way decreases the computing power needed to run the client application.

In a more general way, we studied each choice we made and never chose the first solution found as definitive. Questioning all of our choices was the biggest challenge we faced, but it ensured that we implemented our middleware using the best means available.

5
Demonstration

In this chapter, we will take a tour of our application in order to explain how to use it but also to demonstrate its functionalities. We will start first with a bit of theory by explaining all the use cases that users have access to. Then we will have a guided tour through our prototype.

5.1 Use Cases

As one can see in Figure 5.1, there are three possible use cases for the users of our application.

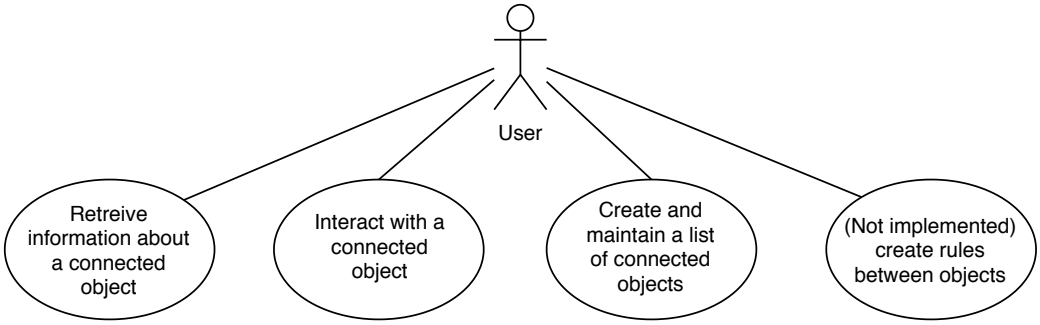


Figure 5.1: Use cases

- Retrieve information about a connected object: the user has to be able to scan an object and monitor its parameters.
- Interact with a connected object: through some control panels, the users can interact with the devices connected to their local network.
- Create and maintain a list of connected objects (also called the overview in our prototype): thanks to this list, the users will not have to scan the objects they want to interact with or monitor.
- (Not implemented) create rules between connected objects: through our middleware, users are allowed to create some rules between devices. Thanks to the created rules they can automate the behaviours of certain objects and therefore avoid many interactions.

5.2 Prototype Walkthrough

5.2.1 Step 1: Connect the Object to the Local Network

To do this part, the user should use the manufacturer's application and follow the process there.

5.2.2 Step 2: Access our Application

As seen on Figure 5.2, the first thing to do is to connect through a web browser to "http://ipofserver:5000/"

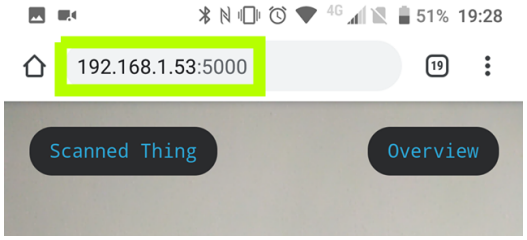


Figure 5.2: Scan an object

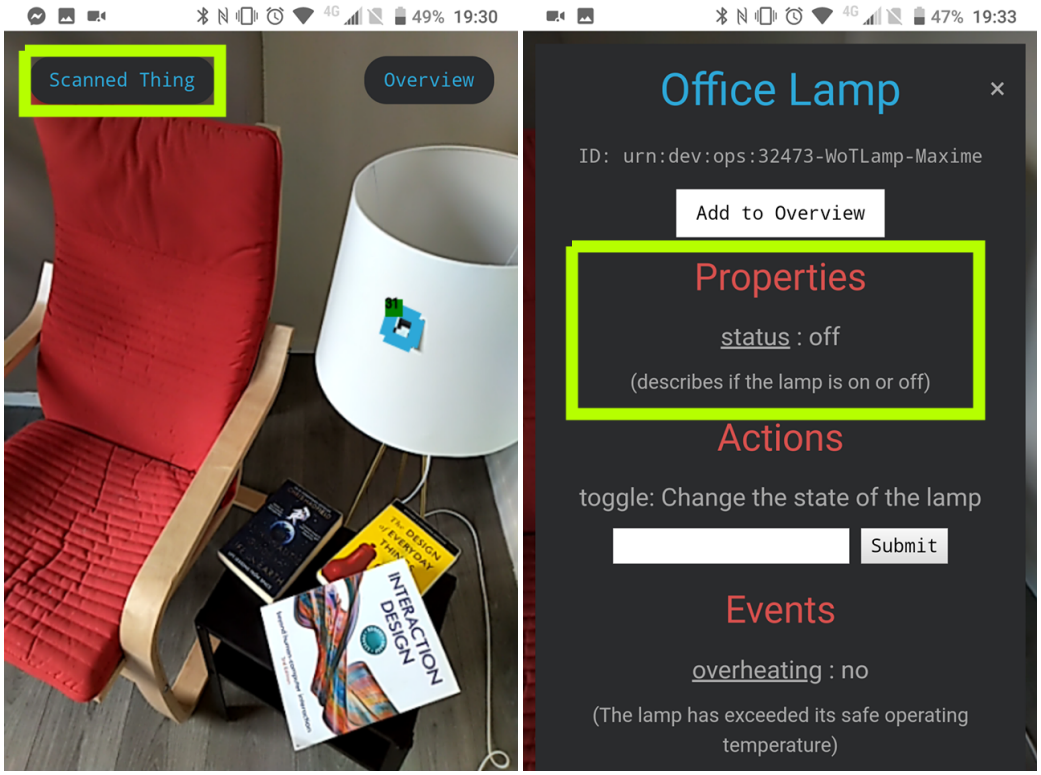
5.2.3 Step 3: Scan an Object

Point the phone to an object and wait for the program to detect the marker, when a coloured square appears, that means that the object has been scanned. Figure 5.3 shows how the interface acts when an object has been scaned.

Figure 5.3: Scan an object

5.2.4 Step 4: Retrieve Informations About an Object

After scanning the object, click on the "Scanned Thing" button (or the "Overview" button if the object has already been added). Then read the properties if there are some. The Figure 5.4 bellow shows in detail the steps to retrieve the object informations.

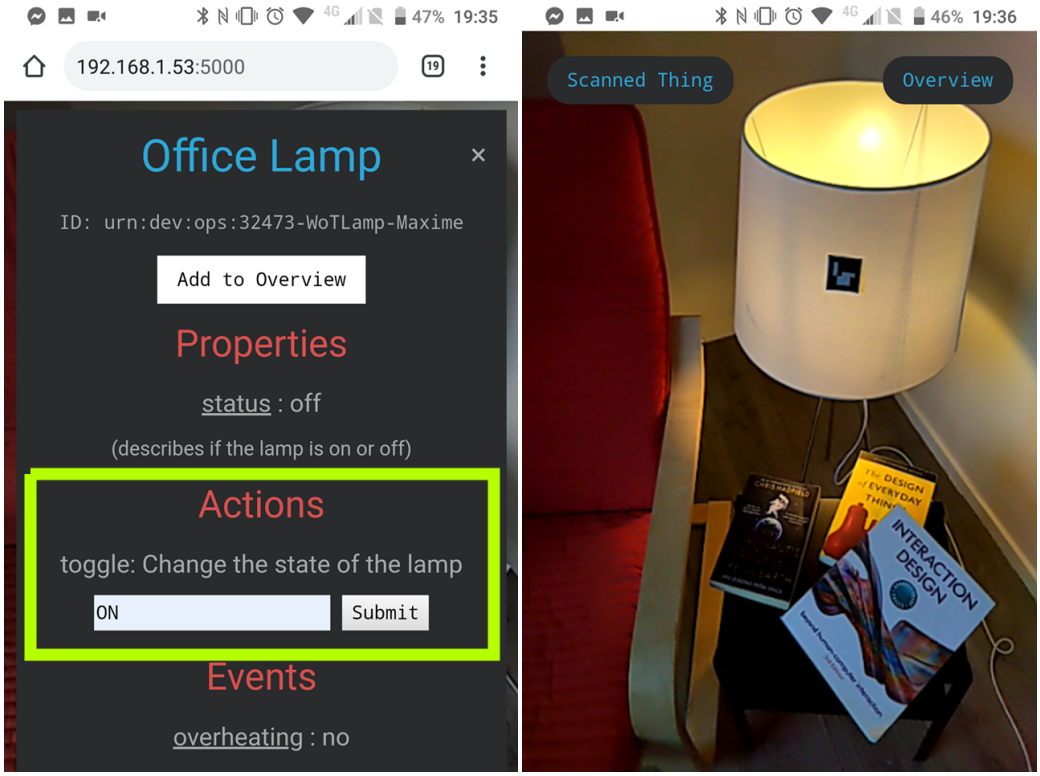


(a) Click on "Scanned Thing" button (b) Look at the properties

Figure 5.4: Retrieve the object informations

5.2.5 Step 5: Interact With an Object

After scanning the object, click on the "Scanned Thing" button (or the "Overview" button if the object has already been added). Then enter a modification in the input fields that appears in the "Actions" section then submit it. The Figure 5.5 bellow shows in detail the steps to interact with an object.

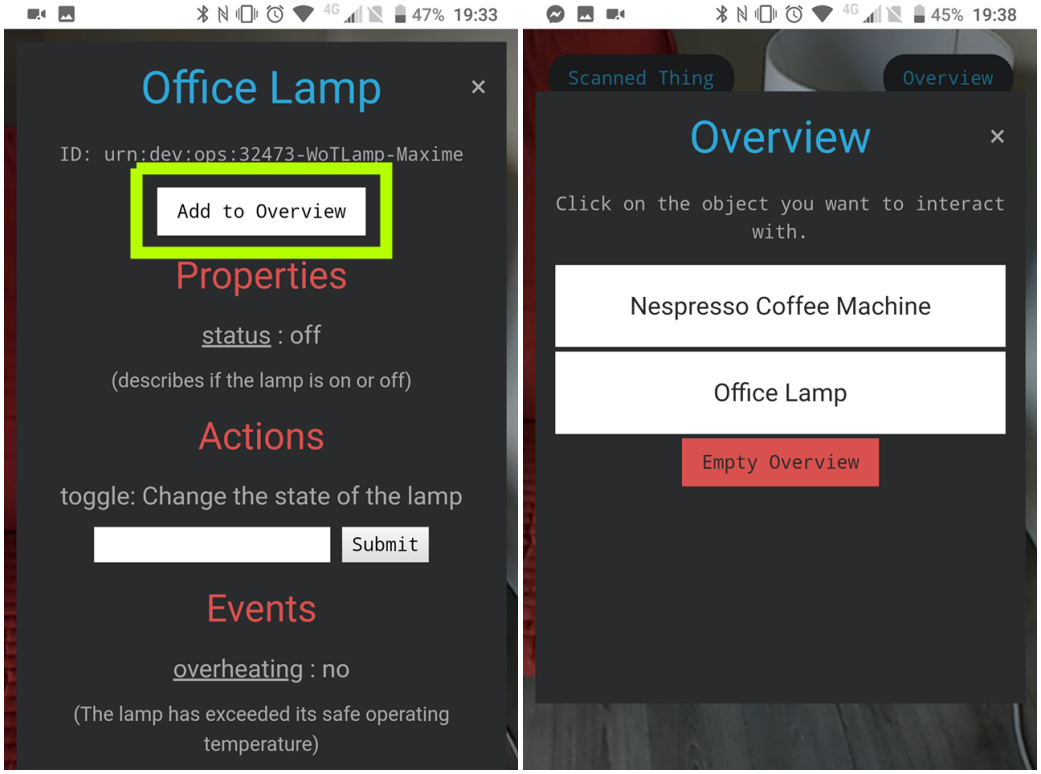


(a) Enter Modification (b) See Results

Figure 5.5: Interact with an object

5.2.6 Step 6: Add Items to the Overview

Click on the "Scanned Thing" button then click on the "Add to Overview" button. After that, go back to the AR view and click on the "Overview" button to see its content. It is also possible to empty the overview by clicking on the "Empty Overview". The Figure 5.6 bellow shows in detail how to add objects to the overview.



(a) Click on "Add to Overview" button (b) Go in overview via AR view

Figure 5.6: Adding an object to the overview list

6
Evaluation

As we explained in the previous chapters, we created a prototype as a proof of concept that the use of Web of Things concepts coupled with Augmented Reality interactions could make it easier for users to interact with their connected objects. However, so far no users have been involved in the process. So how can we be sure that we have succeeded in providing an application that satisfies the system requirements we previously agreed upon? This chapter will address this need by explaining how we have been able to gather the feelings of some people about our application.

In general, a good method to evaluate a user program is to use the User Experience Questionnaire (UEQ) provided by Martin Schrepp et al. [67, 68, 69]. Over the years, they have developed and maintained a quick way to measure user experience for interactive products such as ours. In addition to providing a means of collecting information, they did not stop there. They also provide a data analysis tool that makes life easier for its users. After encoding the answers received to the questions asked to the end-users, this tool creates graphs that help visualise several measures such as attractiveness, perspicuity, efficiency, dependability, stimulation or novelty. Unfortunately, the current Covid-19 crisis does not allow us to carry out this type of user study. Our system would require too much logistics to be tested in this way in such conditions.

It would have implied to that we had to bring several people to our home/laboratory which was forbidden by law. A virtual evaluation was also impossible to perform since our application works only on a local network and requires ArUco markers to activate the object control interface. Also, if we could have avoided these problems, we would have had to send users connected objects and a manual to set up the test. This is why we initially decided to drop the idea of providing a user evaluation. However, at the end of May, the government now allows a few people to come and visit, so we decided to create a questionnaire based on the ideas of the UEQ to evaluate several things. The best we were able to do was to collect the experience of seven users, however we think that there is already enough information to have a first idea of the appreciation of our prototype. We have to keep in mind that the use of the UEQ was impossible because it requires a lot of data to make the results analysis tool work properly.

6.1 Used Questionnaire

As explained previously, the COVID virus did not allow us to perform a large scale test and also limited us in the means we could put in place to perform this test especially in terms of connected devices as we will see later. This is why we have decided to set up a small questionnaire with only 10 questions. The latter, even if it is very basic, will allow us to collect the first opinions on our application. It concentrates mainly on the first impression of the user about five main aspects: likeliness, feedback, interface, ease of use and novelty. One will probably notice some similarities with the UEQ and this is not insignificant. This tool is very high quality and it was important for us to try to reproduce its effect even for a small test like ours. We also took the redundancy aspect of the UEQ, several questions are of the same nature but formulated differently to make sure that the answers provided by the testers are consistent and not given without a thought. Here is the list of question that the tester were asked to answer after using our prototype:

1. Is this system simple to use?
2. Are you happy with the amount of feedback provided by the system?
3. Is the interface of this system pleasant?
4. Is the organisation of the information in the system screens clear?
5. Have you always been aware of the state of the system at all times?

6. Is it exciting to use the product?
7. Was it easy for you to learn how to use this system?
8. Are you satisfied with this system?
9. Do you find this system innovative?
10. Considering your complete experience with our software, how likely would you be to recommend its use to a friend or colleague?

It is very apparent that questions 1 and 7 are targeting the easiness to use of our prototype. Questions 2 and 5 are asking about the feedback received by the user. Questions 3 and 4 investigate the interface and its organisation. The 6th and 9th question are surveying for the novelty of our prototype. Finally the 8th and 10th question are asking the users about their satisfaction of the product. We also added after the questionnaire two open fields for the testers to write down some negative and/or positive aspects about their experience with our system.

6.2 Test Setup

First, the user is offered a phone that runs the prototype and is placed in front of a fake connected lamp bearing a marker. Here, we make the assumption that the users already connected the device they will interact with using the vendor's application and changed the security flags of their browser. This is mainly because we want to test the interface more than the whole experience with the connected device which cannot be optimised as much.

As explained before, due to the Covid-19 virus, we did not have proper connected devices before starting the test. To overcome this problem, we have set up Wizard of Oz experiment. This means that we activated the lamp according to the input of the users to give them the feeling of controlling the object though the application. Even if, in the end, we received a connected device, we wanted the test to keep the same setup in order to be able to analyse all the results the same way. Again, we mainly want to test the application not the environment.

Finally, after letting the user navigate through the application as they wish, we provide them with the questionnaire included in Appendix D and ask them to answer it carefully. We will use the results of this simple evaluation in order to pin point some important modification to make in the future.

6.3 Test Results And Interpretation

Here are the average results for each category studied in this survey:

- **Likeliness**: The average score for both questions combined is 8.1 out of 10. This leads us to believe that the people who tested our prototype felt satisfied with it. This could be explained by the fact that we have taken care to develop a pleasant interface that follows the principles of Human-computer interaction, which includes pleasant colours and easy interactions through the use of augmented reality.

- **Feedback**: The average score for both questions combined is 7.6 out of 10. Indeed, when we designed our system, we put in place numerous means to keep the user informed of the system's status at all times. It is true that this was important to us because too often we encounter applications that leave the user unaware of the status of the system and we did not want to follow this trend.

- **Interface**: The average score for both questions combined is 7.7 out of 10. Even if it is a good score, it is still the second worst. This proves that there is still room for improvement in our system. For the moment it is true that although it is intuitive and has a nice look our interface could sometimes be less loaded with information. This is one of the remarks that came back the most during the evaluations this is why we think that it is partly the explanation of this lower score.

- **Ease of use**: The average score for both questions combined is 8 out of 10. We are pleased to see that we have a good rating for this property. Indeed our system has been designed to be as simplistic as possible and allow the most inexperienced users to handle it with ease and comfort.

- **Novelty**: The average score for both questions combined is 8.1 out of 10. This is our best mark, we attribute this to the fact that middlewares are not yet widespread in our daily lives. Thus users have found the concept innovative. As the problematic of our thesis mentions, it is common nowadays to have a dedicated application for each connected object we buy.

Since, as can be seen above, no category has an average of less than 7.5, we can consider that the system was widely appreciated by the seven participants of our survey. When interpreting the results, we also noticed that the redundancy of the questions in our survey has been successful. Apart from the " novelty " category in each case, people answered the two linked questions in approximately the same way. The open questions allowed us to realise that the users particularly appreciated the ease of use of our middleware and the fact that it could make their life easier in the future. On the other hand, several concerns were expressed about the security of our infrastructure. Some people advised to add a user rights mechanism to combat security abuses. It was also pointed out by the older users that this kind of system is only interesting if you are already interested in the world of connected devices. For them, the investment in terms of time and money does not seem to be worth it.

Considering these results, we are happy to see that our system was well liked by the people who were part of the evaluation. Even if this evaluation is only a quick and simple survey[1], we think that it is sufficient to say that the choices of implementation and the hours spent to develop our system have been successful. It is very rewarding after so much work to be able to see that our middleware is appreciated.

[1]The questionnaires from each user are not included in this document because they would have occupied too much space but they are available on request if you are interested.

7 Future Work

Even though we spared no expense to realise this work and our prototype, there is always room for improvement. This is why, in this chapter, we suggest several future extensions to our work.

7.1 Avoiding the Not Secure Warning

As explained above, it is mandatory that users update their browser's security flags the first time they want to connect to our prototype. According to us, this is one of the biggest drawback of our middleware. A solution must be found if the prototype is to be made public and/or marketed. After several inconclusive tests, we decided to continue and improve the content of our application. Since it is a prototype we decided not to waste too much time on it anyway. The application is now advanced enough to start looking at this subject again.

7.2 Improve the Parser

Our parser which produces object control interfaces based on their TD is very advanced. It already allows to integrate any TD, as long as it respects the W3C conditions and contains at least one of the three tags that contain content to display (Actions, Properties or Events). In spite of this, one could also imagine improving it by making it take into account more tags. It is true that for the moment it only takes into account useful tags to show the user the strict minimum (i.e. interactions and important information). The W3C however proposes tags to add content for the user like for example the possibility of producing interfaces in several languages with the tags "descriptions and titles". From time to time, the W3C adds tags to its standardisation to allow greater flexibility while using the TDs. This mainly allows us to improve the services that can be offered to the user.

Events on the other hand could be handled better by our parser. For the time being, the parser process them exactly the same way it does for the Properties. But in addition to this, one could imagine a user notification and visualisation mechanism for events appearing in the TD. We quickly gave up implementing this feature, the fact of not having a connected device capable of such events prevented us. Creating a virtualised model that integrates this kind of process would have taken too long for a contribution we felt was not as essential as other could be to the work we were doing.

7.3 Application Maintainability

Another important advancement that could be made in the future is to tidy the application files a little bit. We think that it is important to tidy up and refactor some things in the code since there are still a lot of testing scraps left. Of course, being a prototype, it is quite a common thing to have. But our application is also largely made up of big chunks of code that are understandable but that do not yet form an organised entity. Therefore, it would be a good idea to add more documentation to our project, either in the form of a comments in the code or as a user manual. By doing so, the maintainability of our application would be boosted and therefore it would motivate new users and maybe some enthusiastic developers to join our open source project.

7.4 Add New Functionalities

Naturally, adding new functionality is the biggest part of the future work. Here are some of our suggestions. Taking better advantage of the AR capabilities of our system seems to be a good plan. For example, to indicate to the user the location of an object, one could imagine setting up an object positioning module in our prototype. The latter would contain an interface made of augmented reality objects such as arrows pointing to where the user should look. Kind of like what Google does with its Augmented Reality Navigation System as shown in Figure 7.1 but adapted to indoor navigation.

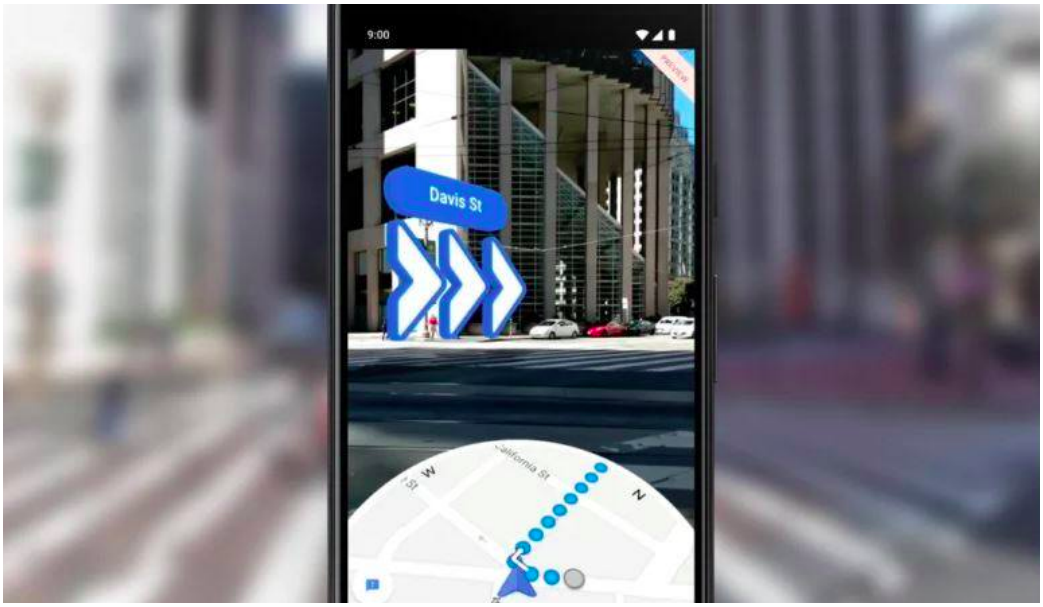


Figure 7.1: Preview of AR walking navigation in Google Maps

Another AR functionality that could be interesting to have is to create buttons or content visualisation near the scanned objects in order to allow faster interactions for specific actions or properties.

Also, other non-AR functionalities could be added to our system as well. The first things that come to our mind are a map of the premises on which the user would place the objects when they first appear in the application. More importantly, according to some people who participated in the evaluation of our prototype, we need ways to secure our system. The most proposed idea is to set up a system that groups users according to the permissions they would have: admin, V.I.P, user, ephemeral user, etc.

7.5 Take Care of the Unfinished Functionalities

Even if we do not intend to abandon this project after our thesis is submitted, all good things must come to an end. It had to be decided at some point to fix the things we wanted to submit with this thesis and therefore we are forced to leave some work unfinished for the moment. Mainly the functionality that will allow users to set up rules between connected devices. A great part of the work has already been done, the visuals are ready and the way to implement it has already been explained in this report and even if it is not the optimal way to do it, we still spent a bunch of time on it. The only thing to do, although it might not be as simple as it looks, is to implement it and deal with the challenges that this creates.

7.6 Create a Community

The dream would be to see this project lead to the creation of a strong community that helps each other and shares in the manner of the community of "makers" that has been growing around 3D printing through the use of websites such as `https://www.thingiverse.com/` for years now. That is why, in our opinion, one of the most beautiful extensions we could make to this project would be to create the infrastructure that will help grow such a community. Also, as soon as the project leaves the context of this thesis, we plan to make all its resources available on GitHub in order to allow other developers to help us evolve our concept in the hope of perhaps making it a must-have in the future when we talk about connected objects.

8 Conclusion

The main goal of this thesis was to design one of the first IoT middleware available on any device with a web browser. By using the AR in addition, we believe we have produced a satisfying outcome that is quite close to our expectations. Indeed, our application allows its users to easily interact with any connected device as long as they have a published Thing Description document. In addition, the results of the user evaluation show that our prototype and its features were highly appreciated. All users who have tested our middleware will recommend it to their friends. This leads us to believe that this kind of system will have an important place in our homes in the future. It also confirms that the design choices we made and our determination to always question our choices in order not to take the first solution found as definitive has paid off. In the end, the challenges we faced helped us to build our prototype cleverly. For these reasons, we conclude that our system is a successful proof of concept in terms of user interaction. For the functionality part, our opinion is that some improvements could make the prototype even more complete and usable. It is still a pity that some parts could not be implemented, in particular the rules. With those, our prototype would have been even closer to the beta program state than to a prototype. Our biggest regret are that we were not able to finalise this feature and that we were not able to test our application with more connected objects and and on more users.

Looking back at the problem statement, I guess we could say this is mission accomplished. The prototype we have developed does indeed solve the Internet of Groups problem. The barriers created by the brands to keep their customers and incite them to buy their products have been broken thanks to the creation of our middleware for which we have used of a little AR and the concepts of the Web of Things. In addition, during the evaluation, users found our system very easy to use. All this leads us to believe that the concepts developed in this thesis will be of central importance for the future of the IoT. This is why we have decided to continue the implementation of this project in the future in our spare time and to make its resources public in order to perhaps interest other people to participate in its conception. Some things, however, can be improved. As explained above, we think the W3C should make an effort to make their WoT documentation less open to interpretation and clearer for less experienced developers. One of the biggest challenges we faced was having to cope with a standard that was changing as our work evolved. On our part, too, some things could have been done better or more efficiently. Our parser, which creates the control interfaces for the objects, could be better and feel more legit with more functionalities. Even if it takes into accounts perfectly every existing object under certain conditions, we can always improve everything. Our interface could also benefit from a little more attention to add more Augmented Reality content. Despite those small imperfections, we are still proud of the overall quality of the work done and the results obtained.

Finally, thesis has taught me many important things for my future professional life. First of all, I learned throughout this year to conduct a large research work. I also learned about the challenges involved in setting up the architecture of a project available on a large number of devices and the prototype such systems. I must say that I am happy to have had the opportunity to carry out my thesis at the VUB surrounded by such comprehensive and helpful people. Thanks to them, I learned how exchanges between colleagues are taking place and how to behave in a working environment. The health crisis context in which a large part of this thesis was written is also responsible for many of the lessons learned. Having worked in extreme conditions made me better. Despite the global, logistical and family problems encountered, I learned to do as any human being does, move forward and do the best I could to produce what I think in the end is a good thesis. To wrap this up, I will use my last words of to thank again each person who have made possible not only this thesis but also this period of my life that is coming to an end. A page is being turned, but I will continue to learn every day.

A

Appendix A: Figures References

Figure	Reference
1.1	https://www.spectralengines.com/articles/industry-4-0-and-how-smart-sensors-make-the-difference
1.2	Self-made
2.1	https://media.springernature.com/lw785/springer-static/image/chp%3A10.1007%2F978-3-319-30913-2_15/MediaObjects/335127_1_En_15_Fig3_HTML.gif
2.2	Self-made
2.3	https://www.w3.org/TR/2017/WD-wot-architecture-20170914/images/wot-existing.png
2.4	Screenshot taken on the 18/04/2020 from https://www.w3.org/TR/wot-thing-description/#introduction
2.5	Screenshot taken in the early stages of the conception of our prototype.
3.1	Self-made.
3.2	Self-made.
3.3	Self-made.
4.1	Self-made.
4.2	Self-made.
4.3	Screenshot taken in the prototype.
4.4	Self-made.
4.5	Screenshot taken in the prototype.
5.1	Self-made.
5.2	Screenshot taken in the prototype.
5.3	Screenshot taken in the prototype.
5.4	Screenshots taken in the prototype.
5.5	Screenshots taken in the prototype.
5.6	Screenshots taken in the prototype.
7.1	https://geospatialmedia.s3.amazonaws.com/wp-content/uploads/2019/08/ARG.jpg

B

Appendix B: List of Tags in a Thing Description Document

Tag line	Description	Assignment
@context	Define short-hand names called terms that are used the TD	mandatory
@type	Label the object with semantic tags	optional
id	Identifier of the Thing	optional
title	Human-readable title	mandatory
titles	Multi-language human-readable titles	optional
description	Additional human-readable information	optional
descriptions	Human-readable information in different languages	optional
version	Version information	optional
created	When the TD was created	optional
modified	When the TD was last modified	optional
support	Information about the TD maintainer as URI scheme	optional
base	Define the base URI that is used for all relative URI references in the TD	optional
properties	All Property-based Interaction Affordances of the Thing	optional
actions	All Action-based Interaction Affordances of the Thing	optional
events	All Event-based Interaction Affordances of the Thing	optional
links	Provides Web links to ressouces linked to the Thing	optional
forms	Set of form hypermedia controls that describe how an operation can be performed	optional
security	Set of security definition names	mandatory
securityDefinitions	Set of named security configurations	mandatory

C

Appendix C: Browsers That Support our Prototype

Browser	Device	Operating system
Chrome	Computer	Windows
Chrome	Computer	Linux
Chrome	Tablet	Android
Chrome	Smartphone	Android
Chrome	Smartphone	IOS
Mozilla	Computer	Windows
Mozilla	Computer	Linux
Mozilla	Smartphone	Android
Mozilla	Smartphone	IOS
Safari	Smartphone	IOS
Edge	Computer	Windows

Note: The first time, to make it work, it is necessary to adapt the security flags of the browser in order to trust the URL of the server. This process can be time consuming and since it would be necessary to find a solution for that in the event of a public release, we do not advise you to try all of them. Contact us for more informations.

D

Appendix D: User Evaluation Questionaire

User evaluation questionnaire: Thesis Maxime Libert

Name :

1. Is this system simple to use?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

2. Are you happy with the amount of feedback provided by the system?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

3. Is the interface of this system pleasant?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

4. Is the organization of the information in the system screens clear?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

5. Have you always been aware of the state of the system at all times?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

6. Is it exciting to use the product?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

7. Was it easy for you to learn how to use this system?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

8. Are you satisfied with this system?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

9. Do you find this system innovative?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

10. Considering your complete experience with our software, how likely would you be to recommend its use to a friend or colleague?

Disagree 0 1 3 3 4 5 6 7 8 9 10 Agree

Write down the main positive aspects of this system:

-
-
-

Write down the main negative aspects of this system:

-
-
-

Bibliography

[1] V. Bush, "As We May Think," *The Atlantic Monthly*, July 1945.

[2] T. Berners-Lee, "Information Management: A Proposal," *No. CERN-DD-89-001-OC*, Mar. 1989.

[3] T. H. Nelson, "Complex Information Processing: A File Structure for the Complex, the changing and the indeterminate," in *Proceedings of the 20th national conference*, (Cleveland, Ohio, United States), pp. 84–100, ACM Press, 1965.

[4] J. Romkey, "Toast of the IoT: The 1990 Interop Internet Toaster," *IEEE Consumer Electronics Magazine*, vol. 6, pp. 116–119, Jan. 2017.

[5] G. Lampropoulos, K. Siakas, and T. Anastasiadis, "Internet of Things in the Context of Industry 4.0: An Overview," *International Journal of Entrepreneurial Knowledge*, vol. 7, pp. 4–19, June 2019.

[6] R. Dotihal, A. Sopori, A. Muku, N. Deochake, and D. Varpe, "Smart Homes Using Alexa and Power Line Communication in IoT," in *Proceedings of the International Conference on Computer Networks and Communication Technologies*, pp. 241–248, Springer, 2019.

[7] L. Y. Mano *et al.*, "Exploiting IoT Technologies for Enhancing Health Smart Homes Through Patient Identification And Emotion Recognition," *Computer Communications*, vol. 89, pp. 178–190, 2016.

[8] M. A. Raja, G. R. Reddy, *et al.*, "Design and Implementation of Security System For Smart Some," in *Proceedings of the International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, pp. 1–4, IEEE, 2017.

[9] M. Kaur and K. S. Saini, "A Framework for Recyclable Household Waste Management System in Smart Home Using IoT," in *Computing and Network Sustainability*, pp. 213–223, Springer, 2017.

[10] N. Skeledzija, J. Cesic, E. Koco, V. Bachler, H. N. Vucemilo, and H. Džapo, "Smart Home Automation System for Energy Efficient Housing," in *Proceedings of the 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 166–171, IEEE, 2014.

[11] E. Rukzio, M. Paolucci, M. Wagner, H. H. Berndt, J. Hamard, and A. Schmidt, "Mobile Service Interaction With the Web of Things," in *Proceedings of the 13th International Conference on Telecommunications (ICT 2006), Funchal, Madeira island, Portugal, 2006c*, Citeseer, 2006.

[12] E. Wilde, "Putting Things to REST," Tech. Rep. UCB ISchool Report 2007-015, UC Berkeley, Nov. 2007.

[13] B. Ostermaier, B. M. Elahi, K. Römer, M. Fahrmair, and W. Kellerer, "Dyser: Towards a Real-Time Search Engine for the Web of Things," in *Proceedings of the 6th ACM conference on Embedded network sensor systems - SenSys '08*, (Raleigh, NC, USA), p. 429, ACM Press, 2008.

[14] D. Guinard, V. Trifa, and E. Wilde, "A Resource Oriented Architecture for the Web of Things," in *2010 Internet of Things (IOT)*, (Tokyo, Japan), pp. 1–8, IEEE, Nov. 2010.

[15] Hevner, March, Park, and Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, p. 75, 2004.

[16] S. Bansal and D. Kumar, "IoT Ecosystem: A Survey on Devices, Gateways, Operating Systems, Middleware and Communication," *International Journal of Wireless Information Networks*, pp. 1–25, 2020.

[17] M. Hoffmann, A. Badii, S. Engberg, R. Nair, D. Thiemert, M. Matthess, and J. Schütte, "Towards Semantic Resolution of Security in Ambient Environments," in *Developing Ambient Intelligence*, pp. 13–22, Springer, 2008.

[18] M. Eisenhauer, P. Rosengren, and P. Antolin, "A Development Platform for Integrating Wireless Devices and Sensors Into Ambient Intelligence Systems," in *Proceedings of the 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*, pp. 1–3, IEEE, 2009.

[19] "LinkSmart - Free, Open Source IoT platform." `https://linksmart.eu/`. Accessed: 2020-04-21.

[20] K. Aberer, M. Hauswirth, and A. Salehi, "A Middleware for Fast and Flexible Sensor Network Deployment," in *Proceedings of the International Conference on Very Large Data Bases (VLDB 2006)*, 2006.

[21] J.-P. Calbimonte, S. Sarni, J. Eberle, and K. Aberer, "XGSN: An Open-source Semantic Sensing Middleware for the Web of Things.," in *TC/SSN@ ISWC*, pp. 51–66, 2014.

[22] V. A. Hax, N. L. Duarte Filho, S. S. da Costa Botelho, and O. M. Mendizabal, "ROS as a Middleware to Internet of Things," *Journal of Applied Computing Research*, vol. 2, no. 2, pp. 91–97, 2013.

[23] L. Roalter, M. Kranz, and A. Möller, "A Middleware for Intelligent Environments and the Internet of Things," in *Proceedings of the International Conference on Ubiquitous Intelligence and Computing*, pp. 267–281, Springer, 2010.

[24] P. Persson and O. Angelsmark, "Calvin-Merging Cloud and IoT.," in *ANT/SEIT*, pp. 210–217, 2015.

[25] "Google Fit: An Open Platform That Lets Users Control Their Fitness Data, Developers Build Smarter Apps, and Manufacturers Focus on Creating Amazing Devices.." `https://developers.google.com/fit`. Accessed: 2020-04-21.

[26] A. Pintus, D. Carboni, and A. Piras, "Paraimpu: a Platform for a Social web of things," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 401–404, 2012.

[27] "Paraimpu You are web." `http://www.paraimpu.com/`. Accessed: 2020-04-21.

[28] T. Zachariah and P. Dutta, "Browsing the Web of Things in Mobile Augmented Reality," in *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications*, pp. 129–134, 2019.

[29] J. D. H. Bezerra and C. T. de Souza, "smAR2t: a Models at Runtime Architecture to Interact with the Web Of Things using Augmented Reality," in *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, pp. 124–129, 2019.

[30] D. D. Guinard and V. M. Trifa, *Building the web of things: with examples in Node.js and Raspberry Pi*. Shelter Island, NY: Manning, 2016. OCLC: ocn930683046.

[31] "Web of Things (WoT) Thing Description." `https://www.w3.org/TR/wot-thing-description/`. Accessed: 2020-04-21.

[32] "Thing Description (TD) Ontology." `https://www.w3.org/2019/wot/td`. Accessed: 2020-04-21.

[33] F. Parwej, N. Akhtar, and Y. Perwej, "An Empirical Analysis of Web of Things (WoT)," *International Journal of Advanced Research in Computer Science*, vol. 10, no. 3, p. 32, 2019.

[34] J. Berrocal, J. Garcia-Alonso, and J. M. Murillo, "Architectures Server-Centric vs Mobile-Centric for Developing WoT Applications," in *International Conference on Web Engineering*, pp. 578–581, Springer, 2019.

[35] M. Noura and M. Gaedke, "An Automated Cyclic Planning Framework Based on Plan-Do-Check-Act for Web of Things Composition," in *Proceedings of the 10th ACM Conference on Web Science*, pp. 205–214, 2019.

[36] K. Baek, H. Moon, and I.-Y. Ko, "Vr-Powered Scenario-Based Testing for Visual and Acoustic Web of Things Services," in *International Conference on Web Engineering*, pp. 514–518, Springer, 2019.

[37] K.-H. Le, S. K. Datta, C. Bonnet, and F. Hamon, "WoT-AD: A Descriptive Language for Group of Things in Massive IoT," in *Proceedings of the 5th World Forum on Internet of Things (WF-IoT)*, pp. 257–262, IEEE, 2019.

[38] E. Korkan, H. B. Hassine, V. E. Schlott, S. Käbisch, and S. Steinhorst, "WoTify: A Platform to Bring Web of Things to Your Devices," *arXiv preprint - arXiv:1909.03296*, 2019.

[39] "WoTify EveryThing." `https://wotify.org/`. Accessed: 2020-04-21.

[40] M. Di Felice, L. Sciullo, L. Gigli, C. Aguzzi, L. Roffia, A. Trotta, and T. S. Cinotti, "WoT Store: a Thing and Application Management Ecosystem for the W3C Web of Things," *arXiv preprint - arXiv:1910.04617*, 2019.

[41] I. Torre, "Keynote talk: Augmented, Adaptive, Accessible, and Inclusive Things.," in *Proceedings of the IUI Workshops*, 2019.

[42] M. B. Ibáñez, Á. Di Serio, D. Villarán, and C. D. Kloos, "Experimenting with electromagnetism using augmented reality: Impact on flow student experience and educational effectiveness," *Computers & Education*, vol. 71, pp. 1–13, 2014.

[43] P. Diegmann, M. Schmidt-Kraepelin, S. van den Eynden, and D. Basten, "Benefits of Augmented Reality in Educational Environments a Systematic Literature Review," *Benefits*, vol. 3, no. 6, pp. 1542–1556, 2015.

[44] D. Norman, *The Design of Everyday Things: Revised and Expanded Edition*. Basic books, 2013.

[45] A. Dünser, R. Grasset, H. Seichter, and M. Billinghurst, "Applying HCI Principles to AR Systems Design," 2007.

[46] "Multimodal Interaction - Human Car Interaction." `http://humancarinteraction.com/multimodal-interaction.html`. Accessed: 2020-04-21.

[47] "Vuforia: Market-Leading Enterprise AR ." `https://www.ptc.com/en/products/augmented-reality/vuforia`. Accessed: 2020-04-21.

[48] "Wikitude Augmented Reality: the World's Leading Cross-Platform AR SDK ." `https://www.ptc.com/en/products/augmented-reality/vuforia`. Accessed: 2020-04-21.

[49] "ARToolKit Home Page." `http://www.hitl.washington.edu/artoolkit/`. Accessed: 2020-04-21.

[50] "ARKit - Augmented Reality - Apple Developer." `https://developer.apple.com/augmented-reality/arkit/`. Accessed: 2020-04-21.

[51] "ARCore Build the Future." `https://developers.google.com/ar`. Accessed: 2020-04-21.

[52] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic Generation and Detection of Highly Reliable Fiducial Markers Under Occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[53] M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem," *IEEE Transactions on robotics and automation*, vol. 17, no. 3, pp. 229–241, 2001.

[54] "Infsoft GmbH Indoor Positioning, Tracking and Indoor Navigation with Beacons." `https://www.infsoft.com/technology/positioning-technologies/bluetooth-low-energy-beacons`. Accessed: 2020-04-22.

[55] P. P. OnkarPathak, R. Palkar, and M. Tawari, "Wi-Fi Indoor Positioning System Based on RSSI Measurements From Wi-Fi Access Points a Tri-Lateration Approach," *International Journal of Scientific & Engineering Research*, vol. 5, no. 4, 2014.

[56] A. Lindemann, B. Schnor, J. Sohre, and P. Vogel, "Indoor Positioning: A Comparison of WiFi and Bluetooth Low Energy for Region Monitoring," in *HEALTHINF*, pp. 314–321, 2016.

[57] G. Panwar, R. Maurya, R. Rawat, R. Kanswal, and P. Ranjan, "Home Automation Using IOT Application," *International Journal of Smart Home*, vol. 11, pp. 1–8, Dec. 2017.

[58] I. Hong, S. Park, B. Lee, J. Lee, D. Jeong, and S. Park, "IoT-based Smart Garbage System for Efficient Food Waste Management," *The Scientific World Journal*, vol. 2014, 2014.

[59] C. Chilipirea, A. Ursache, D. O. Popa, and F. Pop, "Energy efficiency and robustness for IoT: Building a smart home security system," in *Proceedings of the 12th international conference on intelligent computer communication and processing (ICCP)*, pp. 43–48, IEEE, 2016.

[60] "Gartner Identifies Top 10 Strategic IoT Technologies and Trends." `https://gtnr.it/36efCaV`. Accessed: 2020-05-18.

[61] "A Runtime and Tooling for Web Technology That Enables Easy Creation of Applications for the Internet of Things." `https://www.thingweb.io/`. Accessed: 2020-05-20.

[62] "Git Code for javaFX UI, a Thing Description Parser coded in JAVA." `https://github.com/danielpeintner/wot-fxui`. Accessed: 2020-05-20.

[63] "Code Examples from "Building the Web of Things" @ Manning." `https://github.com/webofthings/wot-book`. Accessed: 2020-05-20.

[64] "A POC Project That Aims to Generate Reactive and Customizable UIs According to WoT's Thing Descriptions." `https://github.com/Jagni/td_interface_builder`. Accessed: 2020-05-20.

[65] "Web of Things (WoT) Architecture W3C Recommendation." https://www.w3.org/TR/wot-architecture/. Accessed: 2020-05-20.

[66] "Mozzila IoT Introduces Level Links tags: A Way to Read All Properties at once?." https://github.com/w3c/wot-thing-description/issues/151. Accessed: 2020-05-20.

[67] B. Laugwitz, T. Held, and M. Schrepp, "Construction and Evaluation of a User Experience Questionnaire," in *Symposium of the Austrian HCI and Usability Engineering Group*, pp. 63–76, Springer, 2008.

[68] M. Schrepp, A. Hinderks, and J. Thomaschewski, "Applying the User Experience Questionnaire (UEQ) in Different Evaluation Scenarios," in *International Conference of Design, User Experience, and Usability*, pp. 383–392, Springer, 2014.

[69] M. Schrepp, A. Hinderks, and J. Thomaschewski, "Construction of a Benchmark for the User Experience Questionnaire (UEQ).," *IJIMAI*, vol. 4, no. 4, pp. 40–44, 2017.