Remote Phobia Treatment as a Tactile Internet Application Case Study in Edge augmented with Mobile Ad Hoc Clouds Environment

Yassine Jebbar

A Thesis
in
the Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science
in Electrical and Computer Engineering at
Concordia University
Montréal, Québec, Canada

August 2020

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Yassine Jebbar

Entitled: Remote Phobia Treatment as a Tactile Internet Application Case Study in Edge augmented with Mobile Ad Hoc Clouds Environment

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Electrical and Computer Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. F. Khendek

_____ External Examiner
Dr. Y.G. Gueheneuc (CSE)

_____ Internal Examiner
Dr. F. Khendek

_____ Supervisor
Dr. R. Glitho (CIISE)

Approved by: _____
Dr. Y.R. Shayan, Chair
Department of Electrical and Computer Engineering

_____ 20___ _____
Dr. Mourad Debbabi, Interim Dean,
Gina Cody School of Engineering and
Computer Science

Abstract

Remote Phobia Treatment as a Tactile Internet Application Case Study in Edge augmented with Mobile Ad Hoc Clouds Environment

Yassine Jebbar

Tactile Internet is a next generation Internet that allows the transmission of haptic sensations in addition to audio and video content. It is expected to enable new latency-sensitive and critical use cases, such as remote phobia treatment, tele-surgery and autonomous driving. However, the current networking infrastructure cannot ensure the strict requirements that come with Tactile Internet, namely ultra-responsiveness and ultra-reliability.

Edge computing can help in solving this issue. While Cloud Computing offers powerful computing resources at distant data centers, Edge computing provides resources closer to the end user. To this end, computations can be offloaded from the cloud to the edge to obtain lower latency. In addition, as the Edge itself may prove to be insufficiently close to the end users' devices in some cases, it can be augmented with Mobile Ad-Hoc Clouds. The Mobile Ad-hoc Clouds refer to a group of mobile devices located at the immediate vicinity of the end users, offering their available resources for computation, leading therefore to a reduced latency. Nevertheless, the design and implementation of an architecture based on edges augmented with mobile ad-hoc clouds for Tactile Internet raises several challenges. Firstly, a tactile internet-based architecture for remote phobia treatment should allow the exchange of auditory, visual and haptic information to ensure the efficiency of the therapy. Secondly, the end to end latency should be in the order of a few milliseconds to avoid "cyber-sickness".

This thesis provides a case study of edge augmented with mobile ad-hoc clouds architecture for remote phobia treatment. The contributions are threefold. First, a software architecture for remote phobia treatment is designed for an edge augmented with mobile ad hoc clouds environment. Second, a proof of concept prototype for the proposed architecture is implemented and evaluated using a set of haptic devices, which include the HTC Vive VR headset, the Leap Motion hand tracking device, as well as the Gloveone haptic glove. Third, a set of experiments consisting of placing the components in the different layers (i.e. Cloud, Edge and Mobile Ad-hoc Cloud) were conducted, which allowed an evaluation of the impact on performance. A set of high-level interfaces were introduced to allow communication with the heterogeneous devices. The design of the architecture as a set of software modules allows the reusability of the architecture.

Acknowledgments

I would like to express my deepest gratitude to my supervisor Dr. Roch Glitho for his patience, support, and his valuable feedback that inspired me during this thesis. I would also like to thank Dr. Fatna Belqasmi for her patience, help and feedback during this work.

I also thank my colleagues at Telecommunication Services Engineering Lab for their help and encouragement. I would particularly like to express my deepest appreciation to Nattakorn Promwongsa for being a supportive friend during this journey.

I would like to thank my family for their unconditional love and support before, throughout and certainly after this journey. I would also like to express my gratitude to my brother Oussama, not only for his financial and mental support since coming to Canada, but also for being an ultimate source of inspiration for me to improve myself on a daily basis.

I would equally like to thank my friend Omar Hassane, for all the good memories and funny moments during the past 2 years.

Last but not least, I express my deepest thanks and gratitude to my dear girlfriend Afaf, for her love, patience, support and help in any capacity during this journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we begin with a definition of the key concepts related to our work, namely remote phobia treatment, cloud computing, edge computing as well as the mobile ad-hoc cloud. Afterwards, we present the motivation and the problem statement. Then, we discuss the thesis contributions. Finally, the last section introduces the thesis organization.

1.1 Definitions

In the following subsections, we will define the relevant concepts to the research domain in this thesis. To be more specific, we define the following: Remote Phobia Treatment, Cloud Computing, Edge Computing and Mobile Ad-hoc Cloud.

1.1.1 Remote Phobia Treatment

In the field of psychology, it is not uncommon to come across people who express extreme fears when exposed to particular objects or situations. Those are known as specific phobia patients. Phobia treatment refers to the psychological discipline of determining the adequate medical/psychological processes to treat those patients. Specific phobia is characterized by an excessive, irrational fear of a specific object or situation, which is avoided at all cost or endured with great distress [7]. With the recent technological developments in the last two decades, Virtual Reality (VR) has been leveraged in phobia treatment [8]. Although VR is currently used for therapy sessions where the patient and the therapist are physically present on the same location, it has opened a new possibility where the therapy session can be conducted remotely. With the emergence of Tactile Internet, such therapy sessions can become even more feasible. Tactile Internet is a next generation Internet. It allows the exchange of haptic sensations in addition to audio and video content [9]. Tactile Internet has been described as a communication infrastructure combining low latency, very short transit time, high availability, and high reliability with a high level of security [10] [11]. Finally, a variety of haptic devices may be used to transmit tactile sensations. Haptic devices are mainly divided into two broad categories: control devices such as the Falcon Haptic device [12] and the Touch Haptic Device [13], and haptic rendering devices such as Avatar VR [14].

1.1.2 Cloud Computing

Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services [15].

Cloud computing has five essential elements: on-demand self-service, broad network access, resource pooling, rapid elasticity as well as measured service [16]. Cloud computing offers three main facets: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). While the IaaS sits at the bottom layer of cloud computing, the PaaS is located at the middle layer on top of IaaS, which allows it to provide a rapid development environment to build and deploy applications as SaaS.

1.1.3 Edge Computing

Edge computing refers to the enabling technologies allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud services and upstream data on behalf of IoT services [17]. The environment of Mobile Edge Computing is characterized by low latency, proximity, high bandwidth, and real-time insight into radio network information and location awareness. Application provisioning at the edge is enable by several key concepts including cyber foraging, cloudlet, multi-access edge computing as well as fog computing. In cyber foraging, resource-limited mobile devices exploit the capabilities of nearby servers, connected to the Internet through high-bandwidth networks. However, cloudlets rely mainly on reusing modern cloud computing techniques such as virtual machines (VM) based - virtualization. They are resource-rich servers or clusters of servers located in a single-hop proximity of mobile devices. As for Multi-Access Edge Computing, it is an industry initiative under the auspices of the European Telecommunication Standards Institute (ETSI). It was initiated in 2014 under the name of Mobile Edge Computing (MEC), with a focus on mobile networks and VMs as virtualization technology. Finally, fog computing is a concept introduced by CISCO in 2012. It is an extension of cloud computing paradigm from the core to the edge of the network. It enables computing at the edge of the network, closer to IoT/the end-user devices. It also supports virtualization. However, unlike cloudlet and MEC, fog is tightly linked to the existence of a cloud [3].

1.1.4 Mobile Ad-hoc Cloud

As the edge itself may, in some very specific cases, prove to be not sufficiently close to the end user to fulfil very strict applications requirements, it can be augmented with a mobile ad hoc cloud. A mobile ad-hoc cloud is a group of mobile devices in the vicinity of the end user willing to share their resources with each other by taking some incentives. It is usually deployed over mobile ad-hoc networks [18], which allows the execution of compute-intensive applications by leveraging the resources of other mobile devices [19]. As an alternative solution, mobile ad-hoc cloud is an emerging paradigm that mitigates several bottlenecks of the current existing computing paradigms (i.e. Cloud and Edge), such as longer delay and low throughput. Moreover, mobile ad-hoc clouds offer a viable solution for a mobile device to execute an application when there is no or weak wireless internet connection to the remote cloud, or the nearby cloudlet is not available [20].

1.2 Motivation and Problem Statement

In the U.S., specific phobia is one of the most common psychiatric disorders, with a lifetime prevalence of 12.5% [21]. Phobia treatment is commonly treated using systematic exposure [22], where a patient is gradually exposed, across several therapy sessions, to the feared

object or situation, in the hope that this process will eventually lead him to overcome his phobia. Treating such psychological disorders is not an easy task, even in the physical presence of a therapist. To this end, conducting a remote phobia treatment therapy session can be even more challenging. First, the therapist should be able to conduct the therapy session thoroughly, as patients can react differently during the therapy session, depending on the severity of the patient's phobia. This raises an initial challenge where the therapist should be able to interact with the patient in a multimodal manner (auditory, visual, and tactile). Second, another challenge that is directly related to the multimodal interaction arises. Indeed, as the therapy session is taking place over the Internet, it is very likely that typical networking issues such as high latency can occur. This can result in an asynchronization between the visual, auditory and tactile information transmitted from one end to another during the therapy session. This problem is commonly known as "cyber-sickness", where, for instance, the visual information captured by the therapist or the patient does not correspond to the received tactile sensations. Finally, to increase the efficiency of the therapy session, haptic devices must be incorporated. However, as such devices are provided by different vendors, they require different technologies and APIs to be handled and integrated. Therefore, another challenge is to provide high level interfaces to handle and integrate those devices.

1.3 Thesis Contributions

The thesis contributions are as follows:

- Set of requirements for remote phobia treatment applications in the context of Tactile Internet.
- Review of the state of the art with an evaluation based on our sets of requirements.
- An architecture for remote phobia treatment.
- A high-level interface for uniformly accessing the heterogeneous tactile devices.
- A prototype implementation and performance evaluation.
- A simulation using Gloveone haptic gloves for haptic rendering, HTC Vive headset for virtual reality, and Leap Motion for hand movement tracking.

1.4 Thesis Organization

The rest of the thesis is organized as follows:

Chapter 2 discusses comprehensively the key concepts behind this research work.

Chapter 3 introduces the motivating scenario. A set of requirements for the remote phobia treatment applications are derived from the scenario. The state of the art is also evaluated against the requirements.

Chapter 4 presents the proposed architecture for remote phobia treatment. Architectural components and the proposed interfaces are discussed.

Chapter 5 describes the architecture of the implementation and the technologies used for the proof-of-concept prototype. The performance measurements evaluating the architecture are also presented.

Chapter 6 concludes the thesis by providing a summary of the overall contributions and identifying the future research directions.

Chapter 2

Background

This chapter presents the background concepts behind the research domain of this thesis. The explained concepts include: phobia treatment, tactile internet, cloud computing, edge computing, and mobile ad-hoc cloud. The concepts are introduced and defined in the following sections, followed by a conclusion section at the end of the chapter.

2.1 Phobia Treatment

In this first section, the general definition of phobia treatment is given. Then, we give a brief description of the main protocols used for phobia treatment.

2.1.1 General Definition of Phobia Treatment

According to Diagnostic and Statistical Manual of Mental Disorders (DSM-IV), a specific phobia is defined as a marked and persistent fear that is cued by the presence or anticipation of a specific object or situation [23]. The specific phobia patients tend to experience excessive or unreasonable fear, which is usually associated with functional impairment or subjective distress, and is typically accompanied by an immediate anxiety response and avoidance of the feared object or situation. DSM-IV includes five main specific phobia types: animal type, natural environment type, blood-injection-injury type, situational type, and "other type" [23]. "other type" was included in DSM-IV to describe phobias not easily classified using the four main specific phobia types. Examples of phobias from the "other type" include fears of choking, vomiting, and balloons, although any phobia not easily classified as one of the other four types would be classified in this category. Phobia treatment refers to the set of processes and protocols followed by therapists to cure these psychological disorders. In psychology literature [8] [24], systematic exposure to the feared object is often considered to treat phobias. Systematic exposure therapy consists of exposing the patient to the subject of phobia (i.e. the spider in this use case), gradually and under the therapist's supervision. The therapy can start with the therapist exposing the patient to a related but more friendly animal (e.g. a butterfly in case of spider phobia), and it can be done in several steps. The patient can, for instance, be encouraged to interact with the friendly animal visually, and physically afterwards. When the patient is sufficiently comfortable with the friendly animal, he/she will then be introduced to the feared object following similar gradual steps. In addition to systematic exposure therapy, different protocols are used depending on the age and severity of the phobia, among other factors. Those protocols are described in what follows.

2.1.2 Phobia Treatment Protocols

In this subsection, we present briefly the main protocols used in phobia treatment. The introduced protocols are the following: behaviourally focused treatments, physiologically focused treatments, and cognitively focused treatments.

2.1.2.1 Behaviourally Focused Treatments

Behaviourally focused therapy refers to a form of therapy that seeks to identify and help change potentially self-destructive or unhealthy behaviours. The fundamental idea behind this treatment is that all behaviours are learned and that unhealthy behaviours can be changed. This treatment focuses usually on current problems and how to change them. It benefits people who suffer from various kinds of psychological disorders, ranging from severe depression to anxiety and phobias. It is also applied across several age ranges from children to adults. Although behaviourally focused therapy can be applied in various ways by therapists, the most common way to perform this therapy for phobia treatment purposes is via in-vivo exposure therapy and social skills training. In vivo exposure therapy is a form of therapy that aims to reduce the fear associated with the objects or situations which cause the phobia. This is achieved either via rapid exposure to feared situations from the beginning, or gradual exposure coupled with relaxation exercises when anxiety levels become too great. As for social skills training, it is a type of behavioural therapy used to improve social skills in people with mental disorders or developmental disabilities, hence why it is mostly used for social phobia treatment.

2.1.2.2 Physiologically Focused Treatments

The physiologically focused treatments in phobia concentrate on handling the stressful responses and anxiety when a phobia patient is exposed to the feared object or situation. This form of treatment is mainly performed following two methods: system desensitization and applied relaxation. System desensitization relies heavily on classical conditioning. In other words, this form of therapy consists of teaching phobia patients to replace a fear response to a phobia with relaxation responses. The patient starts with learning relaxation and breathing techniques. Once mastered, the therapist will slowly expose them to their fear in heightened doses while they practice these techniques. Meanwhile, applied relaxation is a coping-skill technique based on progressive relaxation, cue-controlled relaxation, etc. The purpose is to teach the patient the skill to relax rapidly (30-40 seconds) in any situation in order to counteract anxiety and tension [25].

2.1.2.3 Cognitively Focused Treatments

Cognitively focused therapy refers to a practical, short-term form of psychotherapy. It relies primarily on patients to develop skills and strategies to overcome their psychological issues. In the context of phobia treatment, cognitively focused therapy encourages the patient to face and resolve his fear sensations when exposed to the feared object or situation. This type of therapy is usually performed in two forms: self-instructional training (SIT) and fading. The purpose of SIT is to teach the patient to substitute negative, catastrophic thoughts, for constructive, positive thoughts before, during and after encountering the phobic situation [26]. In contrast, the purpose of fading is to teach the patient to voluntarily arouse positive feelings by looking at slides of persons, pets, places, etc. to which these feelings are

connected. In the second phase of the fading technique, these positive feelings are used to counteract the negative feelings (e.g. anxiety) that are aroused by watching slides of the phobic object [27].

2.2 Tactile Internet

In this section, we define Tactile Internet from a high-level perspective. Afterwards, we present tactile internet architecture as well as its key technical requirements. Finally, we discuss the enabling technologies of Tactile Internet.

2.2.1 General Definition of Tactile Internet

Tactile Internet refers to a next generation internet, in which ultra-responsive and ultra-reliable network connectivity will enable it to deliver physical haptic experiences remotely [28]. To this end, Tactile Internet is expected to add a new dimension to human-machine interaction through building real-time interactive systems. In contrast to the traditional wired internet and Mobile Internet, which are mainly used to deliver content services (e.g. voice telephony, text messaging, video streams, etc), Tactile Internet is providing a totally new paradigm, in the sense that it allows a smooth transition from content-delivery to skillset/labor-delivery networks, hence why it is also labelled as the internet of skills. At the end user layer of interactive Tactile Internet-based systems, haptic devices for control and feedback are used. In remote robotic surgery, for instance, the interaction begins when the surgeon issues a control command to the distant robot operating on the patient. As soon as the robot touches the patient, haptic feedback is sent from the robot to the surgeon, consequently closing the interaction loop. Although several similarities can be drawn from a comparison between Tactile Internet and simple conferencing applications, the significant differences cannot be ignored. First, Tactile Internet comes with strict requirements, namely in terms of ultra responsiveness and ultra reliability, which is not ensured in conferencing use cases. Second, while conferencing applications are restricted to the conventional transmission of audio and video content, Tactile Internet goes beyond this content to allow equally the transmission of haptic data, which, in itself, is a very significant upgrade, not only as it opens up the possibility for new use cases, but also for the constraints and requirements that come with the introduction of the transmission of such content. In this sense, Tactile Internet applications cannot be considered as enhanced conferencing applications, but rather as distinctive applications for which the requirements, the design, and implementation must be handled in a totally different manner.

2.2.2 Tactile Internet Architecture

Although Tactile Internet will still rely on the common concepts and protocols that constitute the fundamentals of the previous internet generations, it brings novelty in terms of its general underlying architecture. To this end, we use this section to present the key architectural entities of Tactile Internet, which is followed by a functional description of the architecture. Finally, the interfaces in the architecture are discussed. The Tactile Internet Standard (IEEE P1918.1) [1] is used as a reference to present the following subsections.

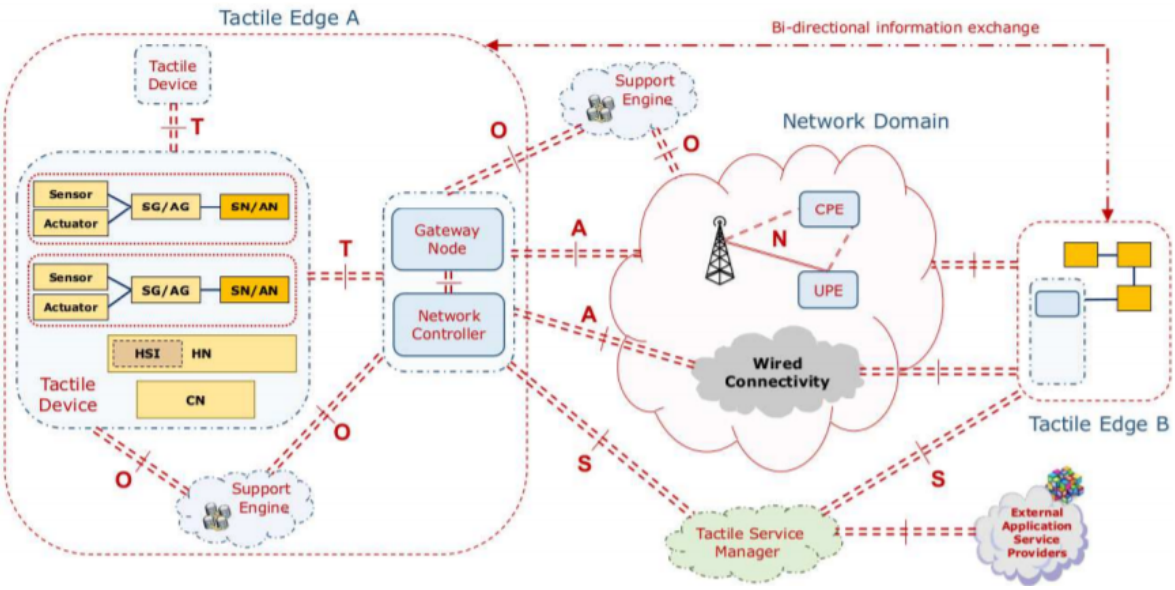


Figure 2.1: The IEEE P1918.1 Tactile Internet Reference Architecture [1].

2.2.2.1 Architectural Entities

• **Tactile Device:** The tactile device is the core element of any tactile edge. Various tactile devices can be used depending on the underlying Tactile Internet application/use case. A tactile device consists of a system of sensor nodes (SNs) and actuator nodes (ANs), which are entities with sensing, actuation and limited processing capabilities. In addition, a tactile device also embodies necessary connectivity modules for networking capabilities. Sensor gateway (SG) and actuator gateway (AG) serve as connectivity gateways to SNs and ANs for third party sensors and actuators. The tactile device can also refer to a human system interface (HSI) that converts human input to haptic input. Finally, another form of the tactile device is a controller which runs control algorithms for controlling a system of SNs and ANs. If equipped with networking capabilities, such a device is referred to as the controller node (CN).

• **Gateway Node:** The gateway node refers to an entity with enhanced networking capabilities. The gateway node provides inter-working functionality between a tactile edge and the network. The reference architecture can alternatively be represented wherein the gateway node and the network controller are part of the network domain (Fig. 2). The gateway node can be co-located with the network controller, and may exist in the tactile edge or the network domain.

• **Network Controller:** The network controller refers to the entity that handles the operation of the tactile edge through intelligent network-side and device-side functions. The main components of the network controller are the network management controller (NMC) and device management controller (DMC). In the case where the network controller is co-located with the gateway node, the resulting entity is termed as the gateway network controller. It may exist in the tactile edge or the network domain.

• **Support Engine:** The support engine is an entity that provides different resources to

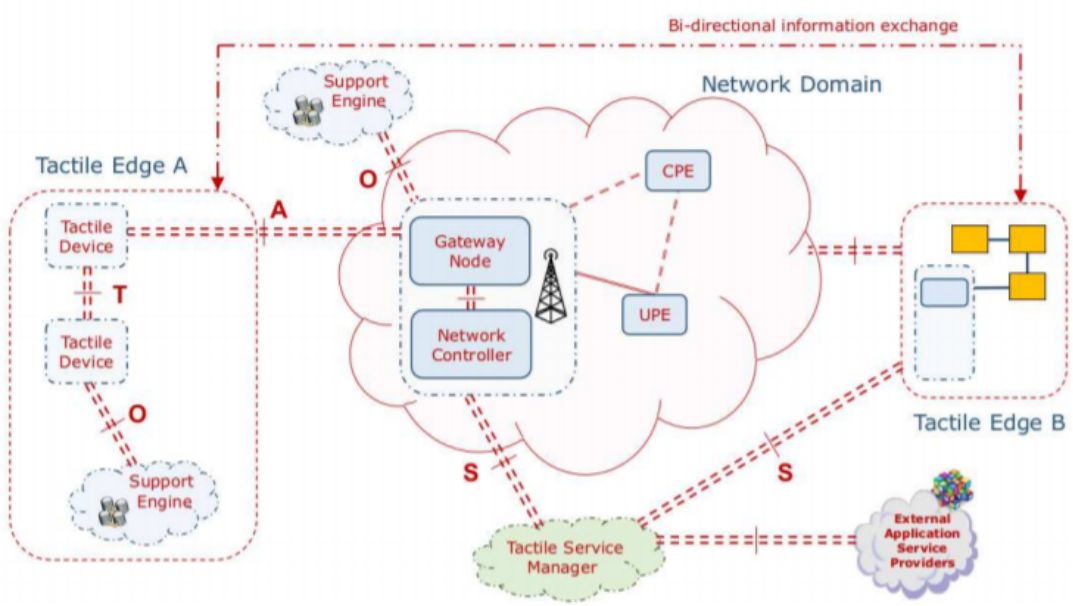


Figure 2.2: Another Representation of the Tactile Internet Reference Architecture [1].

improve the performance/experience at the tactile edge. Those resources can range from computing to storage and networking. The support engine can be part of the tactile edges or the network domain or both. In some cases, the support engine's main responsibility is to run predictive intelligence algorithms to enable the perception of real-time connectivity, especially if we take into account the imperfections of wireless environments. In addition, the support engine can provide computation offloading capabilities by handling processing operations that are resource intensive for the tactile devices. Finally, the support engine may be tasked with providing caching capability.

• **Tactile Service Manager:** The tactile service manager is a network domain entity. It is primarily responsible for providing an interface to the external services and application providers. It may also be responsible for session-level functionalities (e.g. authentication, access rights, session establishment, charging and billing)
• **User-Plane Entity (UPE):** This entity handles the user-plane functions for the tactile edge inside the network domain (e.g. context activation, data forwarding to external networks, and quality-of-service (QoS) support).

• **Control-Plane Entity (CPE):** This entity handles the control-plane functions for the tactile edge inside the network domain, such as authentication, session establishment, and mobility management.

2.2.2.2 Functional Description

As we have already mentioned in the previous subsection, two variants of the reference architecture can be distinguished depending on the placement of the gateway network controller (Fig. 2.1 and Fig. 2.2). In the first scenario (Fig. 2.1), the gateway network controller is located at the tactile edge. The tactile edges A and B represent the master and

slave domains, which can also be simplified as two peer domains. The network domain provides a way to exchange information of control and feedback signals bidirectionally between the tactile devices in the two tactile edges. The network domain may represent wireless connectivity (cellular, WiFi, etc.), wired connectivity (e.g. Ethernet), or hybrid wired/wireless connectivity. In the second scenario (Fig. 2.2), the gateway node and the network controller reside in the network domain. The support engine can be realized by relying on different concepts of the edge computing paradigm such as fog computing, cloudlet-computing, and mobile-edge computing. The support engine can be either centralized or distributed. As mentioned earlier, the support engine provides predictive intelligence, computation offloading. In addition, it may also provide storage/caching functionalities. For predictive intelligence, the support engine runs a model of the Tactile Internet application which could be obtained in either online or offline manner. The support engine may provide full or partial computation offloading capabilities. Furthermore, The offloading decision is made by either the tactile device or the gateway network controller. The tactile service manager is an optional entity. To this end, its existence is dependent on the use case and the underlying connectivity technology. The user-plane and control-plane entities are also dependent on the underlying connectivity technology used. The reference architecture contains mainly entities which are important from a networking and standardization perspective. Other entities that are part of the tactile edges or the network domain can be added to the reference architecture depending on the use case/application. For instance, in some Tactile Internet applications, the master tactile edge may have the provisioning for audio/visual feedback from the slave tactile edge. Hence, the tactile edge may additionally contain necessary audio/visual equipment (e.g. microphones, VR headset, etc).

2.2.2.3 Interfaces

As illustrated in the previous two figures, a set of interfaces are defined between the different architectural entities. In general, the interfaces can be distinguished between physical interfaces and logical interfaces. We will describe each of those interfaces in the remainder of this subsection.

A. Physical Interfaces

• **Access Interface:** The access interface (the A interface in Fig. 2.1) provides connectivity between a tactile edge and the network domain. It is the main reference point for user-plane and control-plane information exchange between the network domain and the tactile edge.

• **Tactile Interface:** The tactile interface (the T interface in Fig. 2.1) provides connectivity between entities of the tactile edge. It is the main reference point for user-plane and control-plane information exchange between the entities of the tactile edge.

• **Open Interface:** The open interface (the O interface in Fig. 2.1) provides connectivity between any architectural entity and the support engine.

• **Service Interface:** The service interface (the S interface in Fig. 2.1) provides connectivity between the tactile service manager and the gateway node and the network controller. The S interface only carries the control-plane information.

• **Network-side Interface:** The network-side interface (the N interface in Fig. 2.1) refers to any interface providing internal connectivity between network domain entities. The N interface is the main reference point for user-plane or/and control-plane information exchange between various network domain entities.

B. Logical Interfaces

In addition to the aforementioned physical interface, the reference architecture has identified various logical interfaces. Those are described in what follows.

• **L0 Interface:** The L0 interface interconnects the gateway node and the network controller.

• **L1 Interface:** The L1 interface interconnects a tactile device and the tactile service manager.

• **L2 Interface:** This interface interconnects a tactile device and the control plane entity.

• **L3 Interface:** The L3 interface interconnects a tactile device and the user-plane entity.

2.2.3 Key Tactile Internet Requirements

The design challenges for the Tactile Internet have been presented in [28]. Human interaction with a technical system can only be perceived as intuitive and natural if the feedback of the system is adapted to our human reaction time. Consequently, the requirements for tactile internet-based systems enabling real-time interactions depend on the participating human senses. To this end, reaction times of about 100 ms, 10ms, and 1ms are required for auditory, visual, and manual interaction, respectively. In what follows, the key Tactile Internet technical requirements are highlighted.

2.2.3.1 Ultra-Responsive Connectivity

Tactile Internet requires ultra-responsive network connectivity i.e., end-to-end latency in the order of 1 ms [29] [30]. Depending on the use case, if real-time transmission goes beyond the end to end latency of a few milliseconds, the tactile users will experience cyber-sickness, which occurs primarily as a result of conflicts between visual, auditory, and tactile sensory systems [31]. Thus, if eyes perceive a movement which is slightly delayed compared to what is perceived by the human ears while the remainder of the human being's body remains static, this delay leads to cyber-sickness. In systems where mission critical communications take place, it becomes even more important to meet these technical requirements (e.g. machine-type communication which enable real-time control and automation of dynamic processes in industrial automation, manufacturing, traffic management, etc). The end-to-end latency in tactile internet-based systems includes the time spent in the transmission of the information from a sensor (or human in case of haptic interaction) via the communication infrastructure to a control server; the processing of the

information and the eventual retransmission via the communication infrastructure back to the actuator (human).

2.2.3.2 Security and Privacy

Safety and Privacy are also among the key requirements for Tactile Internet. With stringent latency constraints, security must be embedded in the physical transmission and ideally be of low computational overhead. New coding techniques need to be developed for tactile applications that allow only the legitimate receivers to process a secure message. Security will be achieved if an illegitimate receiver cannot decode the date even with infinite computational power. This raises another challenge, especially in massive connectivity applications. Identification of legitimate receivers requires novel reliable and low-delay methods. Another method that could be used in this context is relying on hardware specific attributes such as biometric fingerprints.

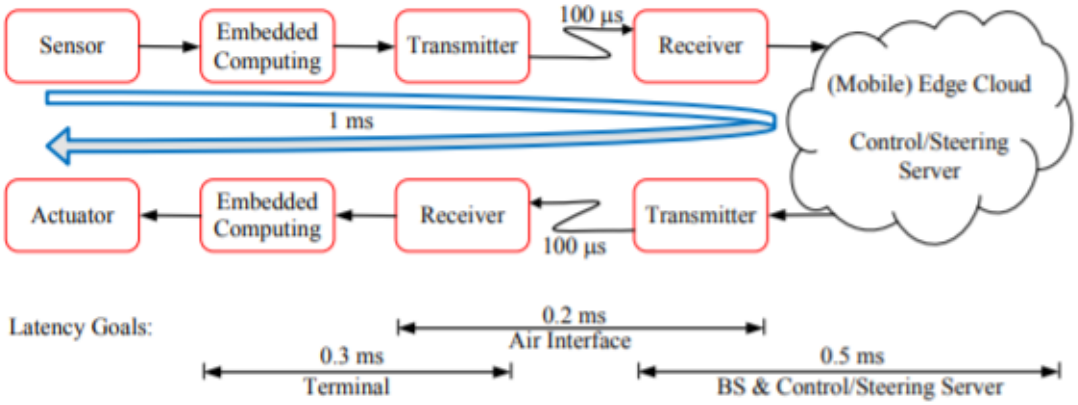


Figure 2.3: Latency Requirement for Tactile Internet [2]

2.2.3.3 Tactile Data

Tactile Internet must handle the tactile information in the same way as the conventional audio/visual information. Therefore, tactile encoding mechanisms are needed, which facilitate transmission of tactile information. Moreover, there must be provisioning of audio/visual feedback due to the highly multi-dimensional nature of human tactile perception [32].

2.2.4 Enabling Technologies of Tactile Internet

To meet the aforementioned requirements of Tactile Internet, the technology needs to be innovated around three major areas: agile networking and communication protocols, artificial edge-intelligence and standardized haptic data digitization and encoding [33]. A skill-oriented design of those areas would ensure a reliable haptic experience. The areas mentioned above are discussed briefly in what follows.

2.2.4.1 Networking and Communication Protocols

Tactile Internet networking infrastructure has to adhere to the following requirements:

- It should be ultra-reliable since many critical tasks will be executed remotely.
- It should have a very low perceived delay, since the transmission of kinesthetic (movement) data requires closed control loops to support action and reaction. In contrast, a longer delay will result in an insatiable system.
- It should rely on edges to enable true scale.

In other words, for the network to be able to deliver the Skills, it needs to have the lowest possible delay whilst being extremely reliable and robust. Network slicing [34] [35] will be a key enabler of reliability and low-latency. Network slicing will allow Tactile Internet to have its own logical core network, which will not be affected by the other traffic in the physical network. By implementing isolation of resources, network slicing provides reliability and robustness to Tactile Internet, as it will guarantee stability of performance during an ongoing session. Network slicing will be implemented through Software Defined Networking (SDN) and Network Functions Virtualization (NFV) [36], where the former will manage the physical communication infrastructure (i.e., switches, links among network entities, path management), while the latter will manage the network functions needed to handle the Internet of Skill session. SDN and NFV are two fundamental building blocks of the incoming 5G networks, where such technologies are expected to provide flexibility to the network to accommodate a wide range of services with heterogeneous QoS needs. Such technologies need to be extended to assure low latency.

2.2.4.2 Artificial Edge-Intelligence

AI plays an instrumental role in giving the perception of low latency networks. Indeed, with AI-based predictions, we have a viable solution in the case of network instability. The haptic control loops typically require a delay of 1-10ms, which translates to 100-1000km range under typical networking conditions. Open research problems here pertain to environment modelling (geometry and physical properties); stable force rendering on the master side; standardized database of environmental models and cloud placement of intelligence and functionalities.

2.2.4.3 Haptic Codec

Finally, the haptic codecs will enable scale in the future, as it will avoid vendor lock-ins, as well as the combination of tactile (touch) and kinesthetic (movement) information into the available modalities of video and audio. Open challenges, in this regard, are to find trade-offs for joint tactile and kinesthetic information and trade-offs for integration of codecs with each other (i.e. visual, auditory and tactile). Finally, compressed sensing solutions can also prove to be useful in this context [33].

2.3 Cloud Computing

In this section, we present a general overview of Cloud Computing. We start with a definition of Cloud Computing. Then, the types of cloud are briefly discussed. Finally, this section is concluded with the benefits which stem from the Cloud Computing paradigm.

2.3.1 Definition

Cloud computing has various definitions. NIST (US National Institute of Standards and Technology) defines Cloud Computing as "model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [37]. Another way to define cloud computing is a "large pool of easily usable and accessible virtualized resources that can be dynamically reconfigured to adjust to a variable load (scale), allowing for an optimum resource utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the infrastructure provider by means of customized SLAs" [38]. Although both definitions are explicit, the former definition (provided by the NIST) covers the essential characteristics of Cloud Computing. Therefore, it is widely accepted and used as the definition of Cloud Computing. There are three layers in Cloud computing: SaaS, PaaS, and the IaaS. Software as a Service (SaaS) is the highest layer in the cloud. In the SaaS, a software vendor can offer a set of cloud-based applications, on top of an already existing infrastructure and platform. To this end, the user does not have to own the used resources. Instead, the user can simply pay for the specific services that has been utilized. Examples of SaaS are Google Docs and Microsoft Office 365. As for the PaaS, it is defined as an enabler for the service providers to develop and deploy their services onto the cloud without worrying about underlying infrastructure [37]. Moreover, It provides an abstraction level on top of virtualized infrastructure located in the IaaS, provisioning resources on demand during execution of running services [38]. Examples of PaaS are Microsoft Azure and Google App Engine. Finally, Infrastructure as a Service (IaaS) refers to the capability provided to the consumer to provision processing, storage, networks, and other fundamental computing resources, where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications [22]. Amazon Elastic Compute Cloud (EC2) and IBM Cloud are examples of an IaaS.

2.3.2 Types of Cloud

Based on the owner(s) and the user(s), a cloud can be classified as either a private cloud, a public cloud, or a hybrid cloud [39] [40]. We describe the cloud types in brief in the following:

• **Private Cloud:** A private cloud is generally owned and used by a specific organization. It is not open for public usage. It allows the employees with the organization to interact with the local data centers while having the same advantages of the cloud. This type of cloud provides performance, reliability and security [40].

• **Public Cloud:** A public cloud refers to the cloud that is available for the general users as a pay-per-use manner, typically expressed in hours, months, year, or a long contract. It is usually owned by big corporations such as Amazon, Google, or Microsoft. This type of cloud lacks some control over data, network and security settings. However, it has full control over the deployed application itself [40].

• **Hybrid Cloud:** A hybrid cloud is a combination of a public and a private cloud, thus, combining the advantages of both worlds. It also allows cloud bursting to take place, which

means a private cloud can burst-out to a public cloud when it requires more resources [39]. The main benefit of hybrid clouds is that it provides more flexibility than both public and private clouds [40].

2.3.3 Benefits of Cloud Computing

The Cloud Computing paradigm offers important benefits. Those are discussed in what follows.

• **Scalability:** Virtually, unlimited scalability is possible because of the massive capacity offered by the cloud providers [41]. Services hosted on the cloud can be easily scaled which is very useful in the event of rapid service demand change.

• **Elasticity:** It refers to a system's capability of adapting to variable workload by provisioning and de-provisioning resources in an autonomic manner [42]. For example, the IaaS, depending on the provider, can provide elasticity to a service. If a task that requires high workload is being executed, the IaaS can automatically allocate more resources. This will allow the service to run as expected even under high workload. However, once this sudden workload is not required anymore, the additional resources that were allocated can be de-provisioned to save costs.

• **Reliability:** Services running on the cloud should meet several desired requirements such as Quality of Service (QoS), availability, performance, fault tolerance, etc. These requirements are regulated under the framework of Service Level Agreement (SLA) between cloud service providers and customers. SLAs contain the details of the service as well as the penalty for violations [41].

• **Multi-tenancy:** Cloud providers can serve multiple customers by assigning and reassigning the virtualized and physical resources dynamically depending on the resources' demand. It facilitates resources sharing, which results in optimum resources utilization and cost.

• **On-demand self-service:** Customers can provision cloud resources any time without human interaction with the cloud service providers [39]. This can be enabled, for instance, by exposing the cloud services to the web via a programmable interface (i.e. Application Programming Interface - API).

• **Pay-per-use Model:** Customers are charged only for the amount of resources they consumed. This measurement parameter can vary based on the services offered. For instance, usage of a virtual machine (of a particular configuration) per hour, number of users consuming a service, etc. [40].

• **Easy Access:** Customers are able to access the provisioned resources easily over the network through various types of devices.

2.4 Edge Computing

As Cloud Computing may prove, in some specific cases (e.g. latency sensitive applications), to be too distant from the end user's device, edge computing is chosen as an alternative. In this section, we begin with a brief definition of Edge Computing. Then, the related concepts to Edge Computing are discussed, namely cyber foraging, cloudlets, Multi Access Edge Computing (MEC) and Fog Computing. Finally, the benefits of Edge Computing are presented.

2.4.1 Definition

Edge computing is a new computing paradigm with data processed at the edge of the network [43]. MEC was introduced to the industrial field as an initiative by the European Telecommunication Standards Institute (ETSI). It was initiated in 2014 under the name of Mobile Edge Computing (MEC), with a focus on mobile networks and VM as virtualization technology. However, its scope has been expanded in March 2017 to encompass non-mobile network requirements (thus the replacement of "Mobile" by "Multi-Access" in the name), as well as virtualization technologies other than VM [3]. Before the scope expansion, the Edge Computing concept aimed at providing cloud computing capabilities at the edge of mobile networks, and within the Radio Access Network (RAN) [3]. The envisioned applications for Edge Computing include augmented reality, intelligent video acceleration, and connected cars. The edges of non-mobile networks and related applications started being considered due to the new scope. The functional entities (before the scope expansion) include the mobile edge platform and the mobile edge host. While the mobile edge platform provides the functionality required to provision mobile edge applications on a specific virtualization infrastructure, the mobile edge host anchors the platform and the virtualization infrastructure.

2.4.2 Related Concepts

Although Edge Computing's main principle is to bring computation and operations to the edge of the network (i.e. closer to the end user), there are a few related concepts that rely on this same principle of resources provisioning. In this subsection, we discuss the main concepts related to Edge Computing, which include cyber foraging, cloudlets as well as Fog Computing.

2.4.2.1 Cyber Foraging

Cyber foraging is among the first concepts for edge computing, although other Edge Computing concepts were introduced afterward (i.e. cloudlets, MEC, fog). Cyber Foraging was introduced by Satyanarayanan and al. in [44] in 2001 and was further refined by Balan et al. [45] in 2002. In cyber foraging, resource-limited mobile devices exploit the capabilities of nearby servers, connected to the Internet through high-bandwidth networks [3]. These servers are called surrogates and perform computing and data staging. Data staging is the process of pre-fetching distant data to nearby surrogates. This process may be required, for instance, when a mobile device has to process a request of a compute-intensive component such as face recognition, which needs to access a large volume of data for face matching. In this case, the mobile device captures raw images and offloads the complex processing to a surrogate. The surrogate, on its end, performs the face detection and matching processes

using a database. This surrogate may stage the database on its local disk and perform the whole or some part of the processing on behalf of the mobile device. The results are subsequently delivered to the mobile device with low latency, since it is close to the device. If the mobile device does not find a surrogate server nearby, it may provide a degraded service to the end-user due to its limited capabilities, which is not ideal.

2.4.2.2 Cloudlets

Cloudlets reuse modern cloud computing techniques such as virtual machine (VM) based - virtualization [3]. They are resource-rich servers or clusters of servers located in a single-hop proximity of mobile devices. They run one or more VMs in which mobile devices can offload components for expensive computation. In the face recognition application example given in the previous subsection, if we were to use cloudlets, the face detection and matching processes will be performed on VMs instead of real machines. Because they rely on VM technology, cloudlets can expand and shrink dynamically, leading eventually to scalability with respect to the mobile users' service requests. Moreover, the VM provides abstraction by separating the guest software environment from the host software environment of the cloudlet. This leads to increasing the chance of mobile users finding a compatible cloudlet to offload their computation-intensive requests anywhere in the world. Using cloudlets, resource-poor mobile devices offload their intensive computations to the cloudlets they use, thereby guaranteeing real-time interactive responses. If the mobile device moves away from the cloudlet, it may connect to a distant cloud. This may lead to a degraded service. Although cloudlets represent the middle tier of a three-tier hierarchy (i.e., mobile device – cloudlet – cloud), in the current definition of cloudlets, the interactions with the cloud is not given significant importance. Cloudlets can also act as a full cloud on the edge. Even when totally isolated from the cloud, cloudlets can exist as a standalone environment, as the cloud does not intervene in the VM provisioning process provided by the cloudlets.

2.4.2.3 Fog Computing

Fog computing, a concept introduced by CISCO in 2012, is an extension of the cloud computing paradigm from the core to the edge of the network [3]. It enables computing at the edge of the network, closer to IoT and/or the end-user devices. It also supports virtualization. Unlike cloudlet and MEC, fog computing is tightly linked to the existence of a cloud, which means that it cannot operate in a standalone mode. Fog has an n-tier architecture, thereby offering more flexibility to the system [46]. The figure below (Fig. 2.4) shows a fog system with a three-tier architecture. It has three strata: The cloud stratum, the fog stratum, and the IoT/end-users stratum. The fog stratum can be formed by one or more fog domains.It can be controlled by the same or different providers. Each of these fog domains is formed by the fog nodes (e.g. edge routers, switches, gateways, access points, PCs, smartphones, set-top boxes, etc). As for the IoT/end-users stratum, it is formed by two domains. The first domain includes end-user devices and the second domain includes IoT devices. It should be noted that one of these two domains may be absent in the stratum. The communication between the IoT/end users stratum and the fog stratum takes place via Local Area Network (LAN). However, Wide Area Network (WAN) is required for the communication between the IoT/end-users stratum and the cloud stratum, whether it is through the fog or not.

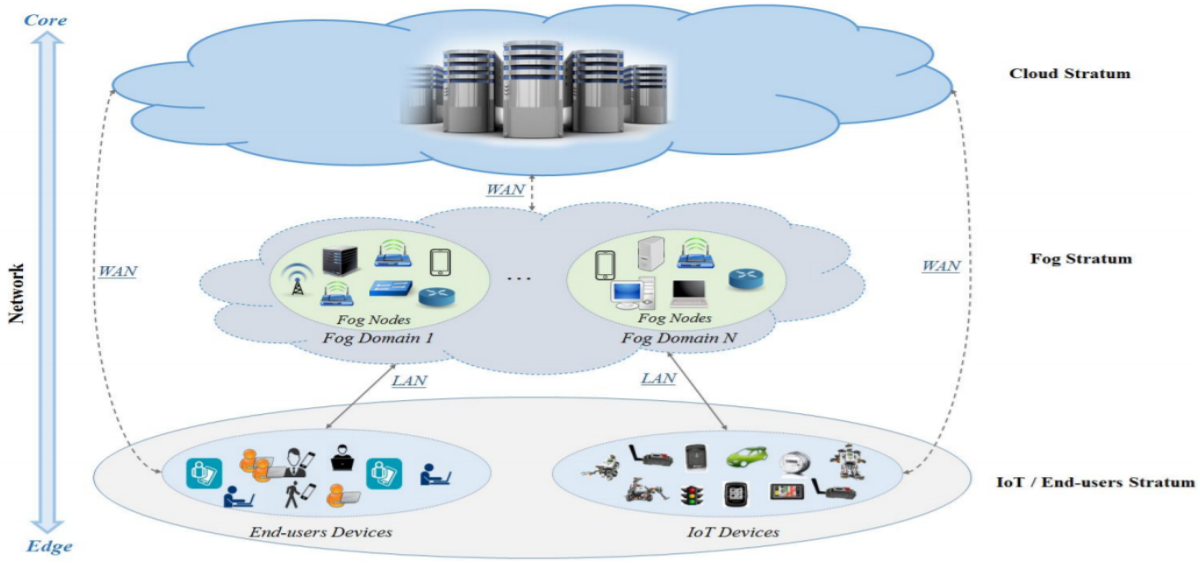


Figure 2.4: The Fog System [3]

2.4.3 Benefits of Edge Computing

Exchanging the powerful distant cloud data centers for computation at the edge of the network brings several benefits. Some of those are discussed in what follows.

• **Reduced Delay:** Indeed, offloading computation from the limited mobile devices to the edge of the network (instead of offloading to the cloud) results in a lower latency. In the research work achieved in [47], the authors built a proof-of-concept platform to run face recognition application , and the response time is reduced from 900 to 169 ms by moving computation from cloud to the edge. Moreover, Ha et al. [48] used cloudlets to offload computing tasks for wearable cognitive assistance, and the result shows that the improvement of response time is between 80 and 200ms.

• **Lower Energy Consumption** The energy consumption is also reduced by 30%–40% when the computation is offloaded to cloudlets. By combining partitioning, migration with merging, and on-demand instantiation of partitioning between mobile and the cloud, CloneCloud's prototype in [49] reduced running time and energy by 20 times for tested applications.

• **Location Awareness** Because edge infrastructure is provided at base stations, applications can easily find an abundance of locations where computation can be offloaded. This can also result in location aware applications, where the deployment location changes depending on the device/user mobility [50].

2.5 Mobile Ad-hoc Cloud

In this section, we introduce the Mobile Ad-hoc Cloud paradigm. Afterwards, we present the various benefits brought by the Mobile Ad-hoc Cloud.

2.5.1 Definition

Edges will be provided by telecom and cloud providers at specific point of presences (PoPs). However, these PoPs might still be too far from some end-users, compared to mobile ad-hoc clouds. A mobile ad-hoc cloud is a new type of Mobile Cloud Computing (MCC) [20]. It is usually deployed over mobile ad-hoc networks [18], which allows the execution of compute-intensive applications by leveraging the resources of the nearby mobile devices that are willing to share their resources [19]. In addition to mobile devices, local stationary devices such as personal computers and set-top boxes can also become members of a mobile ad-hoc cloud [19]. Mobile ad-hoc cloud mitigates several bottlenecks of the existing paradigms (i.e. Cloud and Edge), such as longer delay and low throughput [20]. Moreover, mobile ad-hoc cloud offers an alternative solution to the aforementioned existing paradigms for a mobile device, especially when there is no or weak wireless internet connection to the remote cloud or the nearby edge is not available [20]. A mobile device can be part of a mobile ad-hoc cloud either via negotiation with the discovered nearby mobile devices to form a mobile ad-hoc cloud, or simply ask to join a mobile ad-hoc cloud that has already been formed. The exchange of information and tasks offloading from a specific mobile device to the mobile ad-hoc cloud nearby usually takes place by relying on WiFi direct or Bluetooth to communicate among the devices. Although mobile ad-hoc clouds are used primarily to augment the cloud and edge, the mobile devices in this layer should equally be able to manage the cloud, authenticate the users, monitor the resources, and schedule the tasks, besides executing the application. Such additional functionalities influence mobile device consumption in terms of energy and processor cycles.

2.5.2 Benefits of Mobile Ad-hoc Cloud

The existing mobile ad-hoc cloud solutions offer a variety of benefits, which includes a low latency, resource sharing, maximizing resource utilization, capability enhancement, and security enhancement. Those benefits are discussed in what follows.

• **Low Latency:** As the mobile ad-hoc cloud consists of making mobile devices available at the vicinity of the end users' mobile devices, the deployment of an application on top of a mobile ad-hoc cloud infrastructure can result in a low latency.

• **Resource Sharing:** Another benefit that can be derived directly from mobile ad-hoc cloud infrastructures is the resource sharing. Indeed, as mobile devices collaborate to form a mobile ad-hoc cloud, those mobile devices can offer their computational resources to any other device of the same mobile ad-hoc cloud.

• **Maximizing Resource Utilization:** Mobile devices can volunteer to be part of a mobile ad-hoc cloud formation. Therefore, those devices ensure that they can offer their maximum available resources to be used by any device of the mobile ad-hoc cloud.

• **Capability Enhancement:** Because mobile devices are limited in terms of computational resources, their capabilities are enhanced when they participate in a mobile ad-hoc cloud formation. To this end, a mobile device can offload computation to other mobile devices in the case of computation-heavy tasks.

• **Security Enhancement:** Due to the increased usage of smart mobile devices, malware affecting such devices is rapidly evolving as well. Security risks affecting the confidentiality, integrity, and privacy of smart devices are rapidly emerging. Although mobile security suites exist to defend the device against malware and other intrusions, they require extensive resources which is a constraint of the device itself. Therefore, mobile ad- hoc cloud based intrusion detection frameworks can be provided, which take advantage of WiFi Direct, for instance, to achieve connectivity and sharing resources and services for providing security [51].

2.6 Conclusion

In this chapter, we discussed the background concepts that are related to this thesis. First, we introduced the concept of phobia treatment and described the various protocols used for it. Afterwards, we followed by discussing the tactile internet concept, describing its underlying architecture, as well as the key requirements for tactile internet and its enabling technologies. Then, we discussed the cloud computing concept, starting with a general definition, which was followed by describing the different types of cloud and ending the section with the various benefits of cloud computing. Next, we discussed the edge computing concept, as well as the related concepts, which includes cyber foraging, cloudlets and fog computing, and then we finished this section by listing the main benefits of edge computing. Finally, the mobile ad-hoc cloud concept was discussed in the last section, followed by listing the benefits of the mobile ad-hoc cloud.

Chapter 3

Use Case and State of the Art

In order to draw up the requirements of the tactile internet architecture for remote phobia treatment, we begin with a presentation of a remote phobia treatment use case. Afterwards, the requirements are derived from it. Finally, we evaluate and summarize the state of the art against the requirements.

3.1 Use Case

Let's consider the case of remote spider phobia treatment. As we described in the previous chapter, systematic exposure is the main protocol used for phobia treatment. To offer such treatment remotely via tactile internet, the patient and the therapist can be equipped with a set of tactile devices (e.g. a VR headset and tactile gloves to send and receive haptic sensations). The therapist starts with creating a new session with the patient, and then gradually adds the friendly and feared object to the VR environment. The same therapist may interact with several patients during the same therapy session (i.e. a group therapy) or restrict the session to a single patient (i.e. a one-to-one therapy session). For the remote spider phobia treatment use case, we envision the following steps:

1. The therapist initiates a remote therapy session with the patient over the internet.
2. The therapist gives initial instructions to the patient to make him/her feel comfortable. These instructions are mostly verbal but may also include textual information exchange.
3. When the therapist and the patient both have access to the VR environment and the patient is ready, the therapist adds a butterfly to the VR view.
4. When the patient gets comfortable with the view of the butterfly in the VR environment, the therapist starts gradually touching the insect. The haptic feedback from this interaction is felt both by the doctor and the patient via the haptic gloves.
5. As a next step, the therapist encourages the patient to interact himself/herself with the virtual butterfly. The patient therefore starts touching the insect under the therapist's guidance and instructions.
6. The following steps are to repeat steps 3 to 5 with a virtual spider.

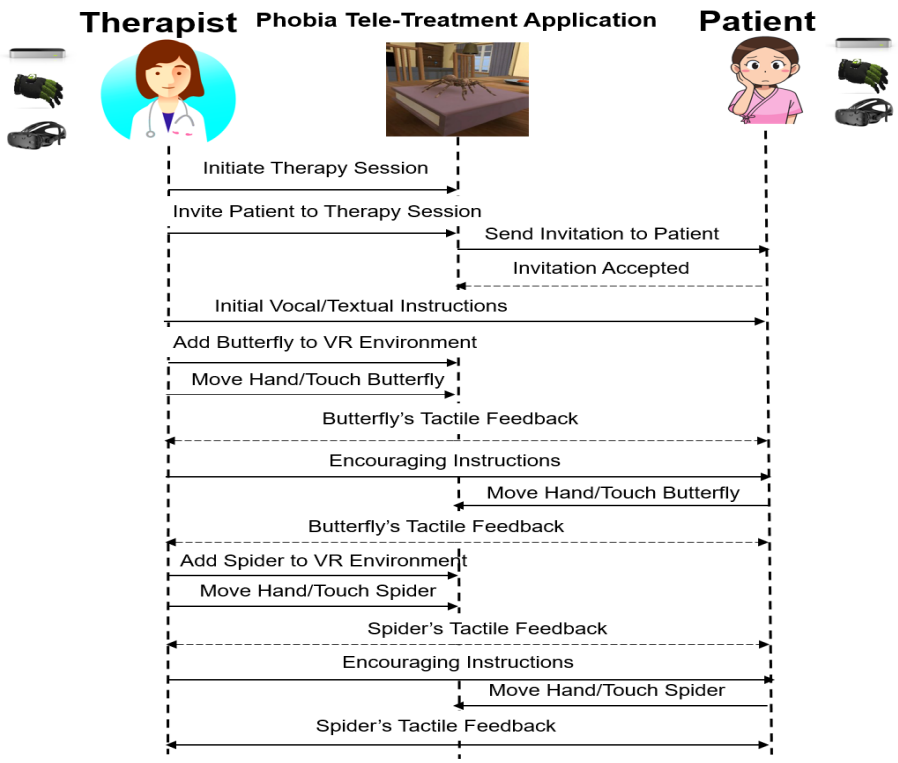


Figure 3.1: Sequence of Interactions During Individual (one to one) Therapy Session

Depending on the severity of the phobia of each patient, therapy sessions can be conducted following those steps either individually or collectively. Figure 3.1 illustrates the "one to one" therapy session, while Figure 3.2 illustrates the "one to many" therapy session. In addition, severe phobia patients may require a session where only one or a few of those steps are performed. When the patient gets accustomed with the therapy process, he is subsequently guided by the therapist through the next steps. Meanwhile, patients with less severe phobia may only need a few sessions to overcome their fear from spiders.

3.2 Requirements

In contrast to in vivo phobia therapy, where the therapist can interact directly with his patient(s) in a face to face session, a remote phobia therapy session raises several challenges. As previously mentioned in the use case description, the therapist needs visual, auditory and tactile interaction with the patient(s). Such multimodal interaction allows the therapist to ensure the efficiency of the therapy session, as he can guide his patient(s) with vocal instructions and encourage them visually to interact with the virtual insects. Indeed, patients can be overtaken by their fears when exposed to their feared objects. Only a gradual and methodical guidance from the therapist can ease patients to the therapy process and the virtual environment where therapy takes place.

To this end, a very first requirement that can be derived from the use case description is that the proposed architecture should allow the exchange of tactile, auditory and visual information.

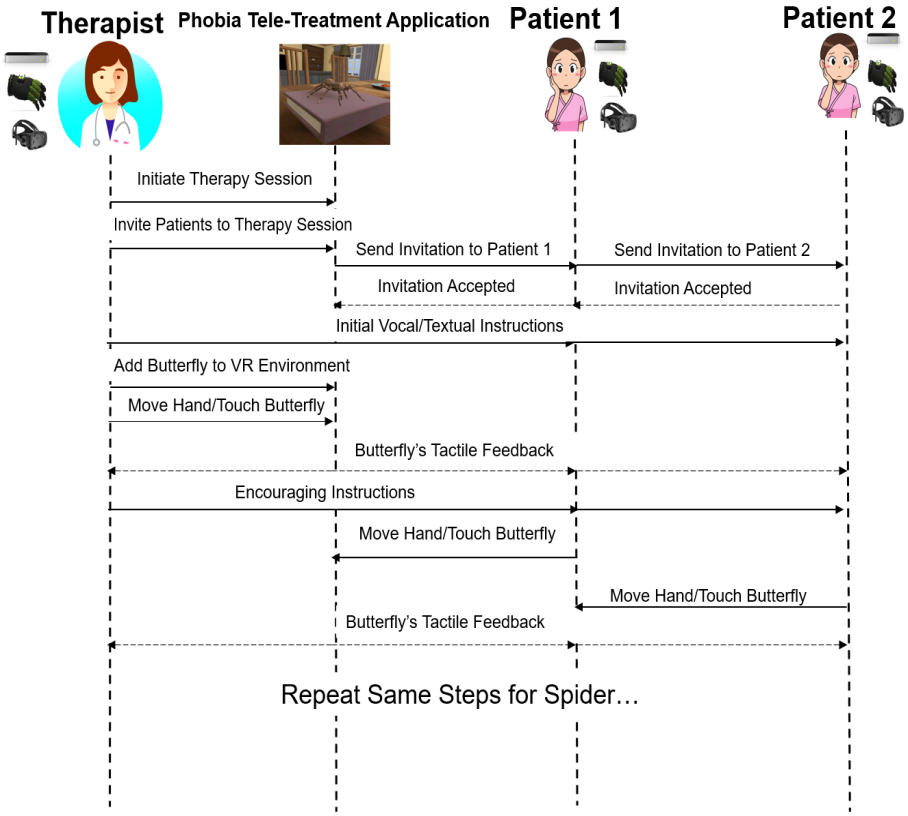


Figure 3.2: Sequence of Interaction for Collective (one to many) Therapy Session

Taking into account that this is primarily a tactile internet use case, the proposed architecture should adhere to tactile internet requirements. Namely, the end to end latency should be in the order of a few milliseconds to avoid what is commonly known as "cyber-sickness" inside virtual environments [2]. Cyber-sickness may occur when, for instance, the perceived sensations at the haptic device of one of the participants (therapist or patient) does not correspond to what is visually perceived in the virtual environments. Such asynchronization occurs mainly due to the difference between high networking latency and the time required to perceive such sensations in human beings.

Hence, the second requirement is that the end to end latency should be in the order of a few milliseconds.

Nevertheless, phobia patients can be treated in groups or individually, depending on the severity of the phobia in each patient. Therefore, a therapy session might be conducted in groups where a therapist interacts with a group of patients simultaneously during a therapy session. An individual therapy session can equally take place, in the case where a patient with a severe phobia needs special treatment from a therapist.

Therefore, a third requirement is that the architecture should allow multi-participant communication, including one-to-one and one-to-many communications.

The therapist and the patient(s) use tactile devices to interact with each other and with the virtual environments. Those tactile devices include VR headsets for visual interaction inside a VR environment, haptic gloves for perceiving tactile sensations, as well as a hand tracking device to track the hand movement, in the case where this functionality is not already provided by the haptic glove. Due to the heterogeneous nature and variety of functions offered by those underlying hardware devices, used by the therapist and the patient(s), to interact with each other during a therapy session, as well as the need to conduct different types of therapy sessions (individual, collective, etc.), high level interfaces must be provided.

Finally, the fourth and last requirement is to provide high level interfaces. This is critical for supporting the plethora of available haptic and virtual reality devices. It is also critical for a smooth and easy extensibility.

3.3 State of the Art

In this section, we review the state of the art for tactile internet architectures for remote phobia treatment. We categorize and review the state of the art in four sections: The first section discusses research work related to the use of virtual reality for phobia treatment. Afterwards, the second section reviews the literature in terms of use of virtual reality in the cloud. Then, the third section evaluates the literature for tactile internet applications. The fourth section explores the papers related to the mobile ad-hoc clouds. Finally, a conclusion section evaluates the papers vis-à-vis the architecture's requirements that are introduced in the previous section.

3.3.1 Use of Virtual Reality for Phobia Treatment

Examples of solutions using virtual and augmented reality for phobia treatment are discussed in [52], [53] and [54].

In [52], for instance, the authors present in their study a comparison between using In Vivo, VR and AR and their respective efficiency to treat phobia of small animals. This study explored the efficacy of three forms of exposure therapy in the treatment of small animal phobia: in vivo based therapy (iVET), virtual reality based therapy (VRET), and augmented reality based therapy (ARET). Aggregated data from three randomized control trials were used for this purpose. In addition, the patients' characteristics were used to detect subgroups and distinguish among patients who showed a different response to certain treatments. The criteria used for the evaluation of the used therapy process were the distance covered by the patient to approach the small animal subject of the phobia, the anxiety felt during the behavioral avoidance test (BAT), and the overall fear of small animals. The main study finding was that the three treatment processes were similarly efficient in the treatment of small animal phobia. Only for the distance covered by the patient, the results revealed a tendency for iVET to be more effective than VRET and ARET in participants with worse performance on the BAT before treatment. The study findings also provide further evidence for the comparable efficiency of the three forms of exposure. The results suggest that, overall, treatments are likely to be similarly effective, regardless of the individual baseline characteristics (i.e., fear, anxiety, and age). As this work is mainly focused on the psychological aspect, there is more emphasis on the efficiency of the used methods than the piece of software used for AR and VR. Therefore, no details

are given about the piece of software used for therapy. In addition, the various treatment processes take place in a room under direct supervision of therapists, which means that the patients were not treated remotely.

Meanwhile in [53], Almeida et al. study the efficiency of using virtual environments containing a spider to treat patients with spider phobia. This paper focuses mainly on the design of a virtual reality based environment for an efficient phobia treatment, as the authors claim that it is necessary to take into account not only the real phobia treatment scenarios that the virtual environment intends to generate, but also the 3D perspectives and the high-level 3D computer graphics. They also state that special attention should be paid to the textures and its details in 3D modeling, as well as its relationship with the lighting of the environment. This will ensure, according to the authors, that the user's immersion in a virtual world is total. To this end, the authors state that in the case of the use of virtual reality exposure therapy, it has shown that users have a very intense sense of immersion, especially when using semi-realistic techniques in the representation of the virtual world. Their results show that virtual reality can be used as an alternative means to cure phobias, especially for arachnophobia. Finally, the paper notes that, with the proper guidance of an expert, The VR-based systematic exposure technique can be combined with those that are used more frequently (e.g. OST: One Session Treatment). Although the authors give some details about the piece of software used in the treatment, they do not state if any high-level interfaces were defined. Moreover, they do not state if the used application allows a remote phobia treatment session.

In [54], the paper suggests a VR-based approach to treat phobias relying on systematic exposure to the feared objects. In addition to visual communication via VR, the phobia treatment process in this paper relies on a camera to detect the motion of the patients. The authors of the paper claim that VR systems used in VRET so far do not fully integrate exposure therapy characteristics. They state that the reason behind this is that they do not provide sufficient, or occasionally not any at all, interaction with the feared stimulus, which is a key factor for full exposure therapy implementation. To this end, their study proposes a way to include natural interaction between the phobia patient and the system during the treatment procedure. The method used suggests an addition to the existing therapy session protocols for mental health professionals through which they can apply exposure therapy in full extent. For this purpose, motion tracking sensors are used. Specifically, they add a Motion Recognition Camera, which tracks the patient's movements and places their physical body within the virtual environment. This leads to increasing the patients' feeling of presence and making the system more immersive. Therefore, the paper states that clinicians can assign interactive tasks for their patients to practice within a controlled virtual environment. The impact of this addition, according to the authors, is that patients stand a better chance with real-time interaction and VR based exposure therapy to truly acquire the necessary skills to overcome their phobias. Indeed, The authors of this paper give more details about the software used in phobia treatment than the previous research works. However, they do not state if any high-level interfaces were defined, nor do they mention if their software can be used in remote phobia treatment.

In sum, the solutions in this category are not cloud or edge-based, but mainly standalone solutions for desktop or mobile devices. They are not meant for remote treatment and they do not address in particular any of the requirements mentioned in the previous section.

Table 3.1 summarizes the evaluation of the related works of this category in regards to our requirements.

Papers	Requirements			
	Multimodal Information Exchange	**Ultra Low Latency**	**Multi-participant Communication**	**High Level Interfaces**
Suso-Ribera et al. [52]	Visual	x	x	x
Almeida et al. [53]	Visual	x	x	x
Kritikos et al. [54]	Visual and partially Tactile	x	x	x

Table 3.1: Summary of the Evaluation of the Related Work for Use of Virtual Reality for Phobia Treatment.

3.3.2 Use of Virtual Reality in the Cloud

The solutions in the second category target using cloud and edge computing to support virtual reality applications. Examples of such solutions are discussed in [55], [56] and [57].

Urrea et al in [55] develop a simulator that recreates, in virtual reality, a team of Selectively Compliance Assembly Robot Arms (SCARA). The robots rely on a cloud server to communicate and work as a team in a collaborative manner to fulfill the task of stacking rectangular objects coordinated through a strategy. The execution of the task is based on a leader/follower configuration. The leader begins with performing a computed trajectory constantly reporting its position to the remote server. This information is sent back to the follower by the remote server. This allows the follower to follow the leader in the tasks that he is performing. The application in this work combines a variety of technologies, and implements communication routines for the purpose of communication among robots. The authors of this paper demonstrated the advantages of leveraging cloud resources for usage into a multi-robot system. In addition, the multi-robot system's performance was tested by means of the developed application. In this work, A high-level interface is defined to communicate with the robots. Furthermore, a group of robots communicate to fulfill the aforementioned tasks. However, the network delay falls behind the few milliseconds defined in our requirements.

Meanwhile, Baco et al [56] propose an Internet of Things (IoT) architecture for a remote monitoring system for ancient buildings. This system uses VR to integrate different kinds of data that is collected either by a wireless sensor network or an Unmanned Aerial Vehicle (UAV). The collected data includes the environmental and mechanical data acquired by a wireless sensor network. The information provided by the UAV allows to inspect the critical structural damage, such as the patterns of cracks in the structural components of the building being monitored. This approach, according to the authors, opens new scenarios to support structural health monitoring (SHM) activities, as an operator can interact with real-time data retrieved from a Wireless Sensor Network (WSN) using the VR environment. Although the architecture relies on high level REST interfaces to communicate between the wireless sensors network and the service, this research work does not discuss the ultra-low

latency challenge, nor does it address if multi participant communication is possible.

Finally, the authors in [57] suggest an educational cloud-based VR system in the field of energy. The system provides students with the fundamentals of solar or Photovoltaic (PV) cells, solar (PV) modules, and solar (PV) array installation configurations. A variety of technologies were used to develop The VR system and host it on the cloud. To this end, the students/users can access the VR system using a standard web browser, which makes the VR system widely accessible. The VR system consisted of self-guided laboratory modules covering electrical engineering fundamentals of solar (PV) cells such as output power losses as a function of finger length, width, depth, and spacing. Additionally, series and parallel solar (PV) cell connections to create desired voltage and current output, and orientation/tilt considerations of solar (PV) array installations were covered. Physics models to incorporate realistic solar energy behavior were programmed in the VR system. Instant feedback was provided to students using live graphs that shows how parameters affected total power output. An adaptive formative assessment system that tested students' application of electrical engineering fundamentals along with design skills was implemented. The system was used in a first year engineering fundamentals undergraduate class to make solar energy education accessible to early state engineering students. The authors claim that the data collected from the cloud system and student survey indicate growth in student engagement, as well as students' knowledge of introductory solar and electrical engineering topics. As this system is designed for students, it allows multi-participant communication. However, the ultra-low latency challenge is not discussed. In addition, no high-level interfaces were defined to interact with the proposed system.

To conclude, most of the research works in this category address the high level interfaces requirement, as they seek to propose Cloud-based VR solutions. However, the latency requirement is either not addressed at all or does not come close to the desired few milliseconds required for our use case. Table 3.2 summarizes the evaluation of the related works of this category.

Papers	**Requirements**			
	Multimodal Information Exchange	**Ultra Low Latency**	**Multi-participant Communication**	**High Level Interfaces**
Urrea et al. [55]	Visual and Tactile	x	✓	✓
Bacco et al. [56]	Visual	x	x	✓
Abichandani et al. [57]	Visual	x	✓	x

Table 3.2: Summary of the Evaluation of the Related Work for Use of Virtual Reality in the Cloud.

3.3.3 Tactile Internet Applications

Tactile Internet-related solutions are more geared towards haptic communication and are more likely to meet the latency requirement. An example of those solutions are discussed in [4], [58] and [5].

In [4], for instance, the focus is on the integration of tactile technology with smart city. To this end, the authors propose a novel QoE-driven Tactile Internet architecture for smart city that contains 5 layers: sensing layer, transmission layer, storage layer, computing layer, and application layer. Fig. 3.3 illustrates the proposed architecture. The main task of the sensing layer is to collect big data from distributed sensors in the smart city. Taking into account the features of haptic information, haptic devices and sensors with actuators, those should be combined with the existing sensing system to build the new sensing layer. Therefore, according to the paper, the devices in the sensing layer gather audiovisual information as well as haptic information. This original information then goes through pre-treatment tasks such as cleaning, filtering and fusion, which allows it to be sent further for delivering and processing. In addition, the authors state that it is necessary to redesign the communication schemes in the transmission layer, as it is difficult for the current systems to meet the ultra reliability and ultra responsiveness requirements. They claim that each layer of this architecture supports techniques that follow the requirements of low latency, high reliability, and high quality of user experience. Finally, the paper shows in its simulation results that it can achieve high QoE performance under the premise of low computational complexity and high availability. Although this paper suggests a Tactile Internet architecture that seeks to meet the tactile internet requirements, no details are mentioned in the paper about multi participant communication, and they do not define any high-level interfaces for communication within their architecture.

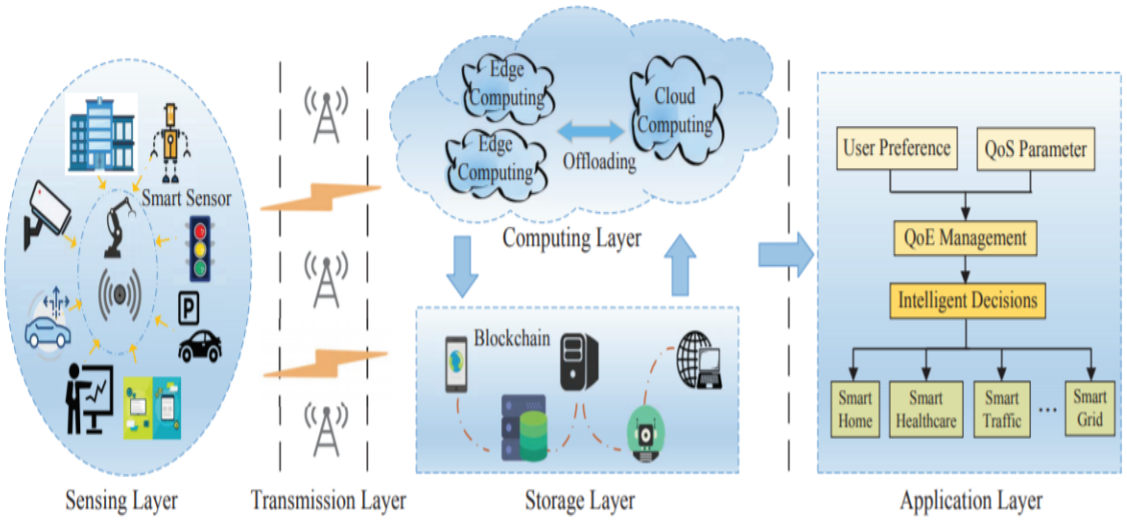


Figure 3.3: The proposed QoE-driven Tactile Internet architecture for smart city [4]

However, in [58], Mekikis et al discuss the key technologies to support the Tactile Internet characteristics in industrial environments. Based on this discussion, they proceed to implement a novel 5G NFV-enabled experimental platform. This platform relies mainly on SDN and NFV, as the authors claim those enable secure and dependable smart sensing and actuation in IoT and Industrial Internet of Things (IIoT) application scenarios. In addition, the platform is implemented as an end-to-end SDN/NFV architecture, complete with the local cloud, SDN networking and Field layers that demonstrate smart actuation, monitoring

and analytics functionalities. The platform also incorporates a variety of hardware. The IIoT services related to smart monitoring and actuation are implemented in the form of VNFs that can be automatically deployed and orchestrated by the cloud controller. Moreover, the implemented and deployed VNFs include smart monitoring and actuation, each deployed in a dedicated Tenant Network. Finally, the authors demonstrate in their measurements that they can attain a sub-millisecond end-to-end communication latency. For this research work, although it addresses the low latency requirement, the authors do not discuss any high-level interfaces, while the multi participant communication requirement is not applicable.

Last but not least, the authors in [5] try to address the open issues related to the lack of a tactile internet testbed, which, they claim, has made it difficult to establish a performance benchmark. In addition, the authors state that issues also exist in terms of the asynchronous efforts led by sub-groups belonging to different research disciplines, which have severely impeded the overall progress of tactile internet in their opinion. To this end, they take the opportunity in their research work to develop a common testbed for tactile internet applications, which they call Tactile Internet eXtensible Testbed (TIXT). According to the authors, this testbed is generic, modular, and extensible. The architecture of TIXT is typically divided into a master domain, a network domain as well as a controlled domain (Figure 3.4). Although the interfaces for the tactile devices are briefly discussed in this work, the low latency requirement is not addressed. In addition, the multi participant requirement is not applicable.

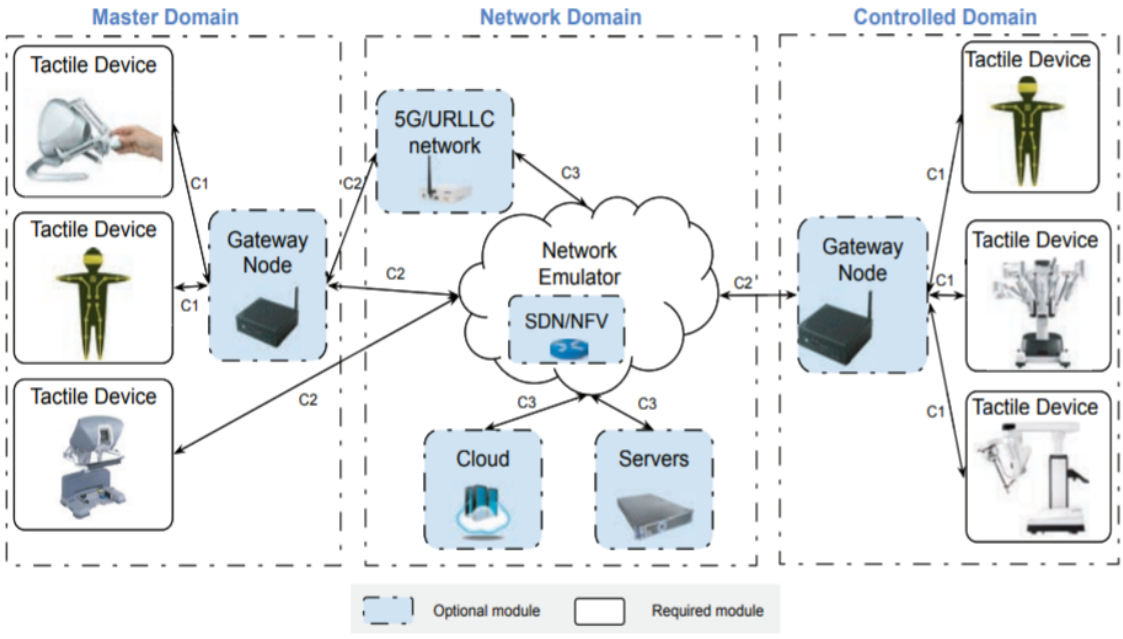


Figure 3.4: TIXT testbed architecture [5]

Ultimately, the papers in this category are more geared towards addressing low latency, as it is one of the main requirements of tactile internet applications. However, little to no attention is given to the high-level interfaces, and the multi participant requirement is not applicable in the vast majority of the papers. Table 3.3 summarizes the evaluation of the related works of this category.

Papers	Requirements			
	Multimodal Information Exchange	**Ultra Low Latency**	**Multi-participant Communication**	**High Level Interfaces**
Wei et al. [4]	Tactile	✓	x	x
Mekikis et al. [58]	Not Applicable	✓	Not Applicable	✓
Gokhale et al. [5]	Not Applicable	x	Not Applicable	✓

Table 3.3: Summary of the Evaluation of the Related Work for Tactile Internet Applications.

3.3.4 Mobile Ad-hoc Clouds

As for the research work addressing mobile ad-hoc clouds, the vast majority of papers address either the architectural aspect (e.g. [6], [59]), or algorithmic challenges such as the MAC formation [60], load distribution/tasks offloading [61], [62], and security issues [51]. To the best of our knowledge, only the authors at [63] discuss using the mobile ad-hoc cloud for a specific use case (i.e. automatic video surveillance).

In the work related to architecture, the authors at [6] address the problem of security attacks and malware affecting mobile devices. They claim that, although mobile security suites defend devices against malware and other intrusions, they require extensive resources not continuously available on mobile terminals. This, according to the paper, affects the relevance, efficiency and sustainability of security services in mobile devices. To this end, the authors propose an ad-hoc mobile edge cloud that takes advantage of WiFi Direct as means of achieving connectivity, sharing resources, and integrating security services among nearby mobile devices. The proposed framework is illustrated in Fig. 3.5. The proposed scheme embeds a multi-objective resource-aware optimization model and genetic based solution that provides smart offloading decision, based on dynamic profiling of contextual and statistical data from the ad-hoc mobile edge cloud devices. The carried experiments in the paper illustrate the relevance and efficiency of exchanging security services while maintaining their sustainability with or without the availability of internet connection. Moreover, the paper's results provide optimal offloading decision and distribution of security services while significantly reducing energy consumption, execution time, and number of selected computational nodes, without sacrificing security. Although the mobile ad-hoc cloud is incorporated in the proposed framework of this paper, high-level interfaces are not defined, and an ultra-low latency is not attained.

The work in [59] suggests an architecture and a protocol for mobile ad-hoc cloud. The paper presents a deep study and a comparison of the new distributed computing paradigms. The proposed solution allows using the resources of mobile terminals in a mobile ad-hoc network to construct a virtual cloud meeting the mobile community resources provider's needs. In addition, a proof of concept study of a proposed protocol for the deployment and the management of a resource sharing architecture is presented. This architecture is mainly composed of interconnected mobile nodes. The authors also implemented the proposed solution and tested it over small sized spontaneous networks. From the experiments, the results obtained detailed measurements of the time required for the architecture's setup and the customers or providers nodes joining. This, according to the authors, permits to

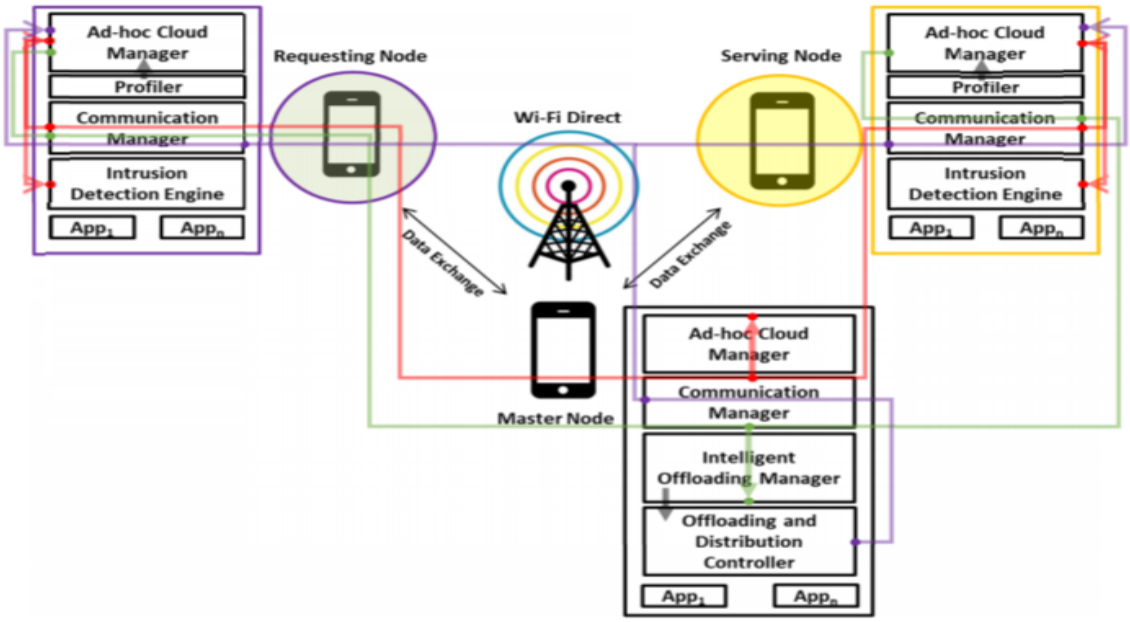


Figure 3.5: Framework Architecture for Ad-hoc Mobile Edge Cloud [6]

prove the protocol's feasibility in a real distributed environment with acceptable times. In a similar way to the previous work, although the mobile ad-hoc cloud was incorporated in the architecture of this paper, the authors do not address the ultra-latency challenge or define any high-level interfaces in their work.

In the use case specific work [63] , Shah et al propose a mobile ad-hoc cloud computing and networking infrastructure to support the execution of a mobile automated video surveillance system, in which multiple mobile devices, interconnected through a WiFi direct based mobile ad-hoc network are combined to create a virtual supercomputing node. In addition, the paper proposes an energy efficient resource allocation scheme for allocation of real time automated video surveillance tasks. To enable communication between mobile devices, a WiFi direct based mobile ad-hoc cloud networking infrastructure has been developed. More specifically, a routing layer has been developed to support communication between WiFi direct devices in a group and multi-hop communication between devices across the group. The proposed system has been implemented on a group of WiFi direct enabled Samsung mobile devices. In contrast to the previous research works, this work discusses the latency requirement. However, an ultra-low latency is not achieved. In addition, high-level interfaces are not defined.

To conclude, although the papers in this category incorporate the mobile ad-hoc cloud in their architectures, the low latency requirement is either not addressed or not achieved. In addition, high level interfaces are not discussed, and the multi participant requirement is equally not applicable in this category. Table 3.4 summarizes the evaluation of the related works of this category.

Papers	Requirements			
	Multimodal Information Exchange	**Ultra Low Latency**	**Multi-participant Communication**	**High Level Interfaces**
Dbouk et al. [6]	x	x	Not Applicable	Not Applicable
Zaghdoudi et al. [59]	x	x	Not Applicable	✓
Shah et al. [63]	x	x	Not Applicable	✓

Table 3.4: Summary of the Evaluation of the Related Work for Mobile Ad-hoc Clouds.

3.4 Conclusion

In this chapter, we presented a detailed description of the remote phobia treatment use case. Then, the set of requirements were extracted based on this use case. Afterwards, we reviewed and evaluated the state of the art based on the requirements. Finally, we concluded that none of the works presented in the state-of-the-art meet all of our requirements.

Chapter 4

Proposed Architecture

In the previous chapter, we derived the requirements for the remote phobia treatment tactile internet application. In this chapter, we aim to propose an architecture based on these requirements. This chapter begins with a presentation of the overall architecture. Afterwards, we present the detailed architecture, where the application components, the functional entities and the interfaces are discussed in the second, third and fourth sections respectively. Then, in the fifth section, we present four illustrative scenarios that include the application components, the functional entities as well as the establishment of a conferencing session. Finally, we show how the proposed architecture meets the predefined requirements before concluding this chapter in the last section.

4.1 General Overview of the Architecture

Figure 4.1 depicts the proposed architecture for this case study with a possible placement of the application components. The architectural components consist of the application components and the functional entities. The remote phobia treatment application includes the following components: *the Therapy Application*, *the Conferencing Server*, *the Conferencing Client*, *the Zone Detector*, *the Feedback Generator*, *the VR Component*, *the Haptic Device Manager*, *the Haptic Device Repository* and finally t*the Haptic Devices*. As for the functional entities, they include the following components: *the Deployment Engine*, *the Domains Repository*, *the Edge/Mobile Ad-hoc Cloud Execution Engine* as well as *the Edge/Mobile Ad-hoc Cloud Zone Head*.

As illustrated in the figure, the proposed architecture incorporates a *Cloud Layer*, an *Edge Layer*, as well as a *Mobile Ad-hoc Cloud Layer*. The cloud layer hosts the computation-heavy application components. Those components do not influence the latency and are simultaneously used by the therapist and the patient(s). The cloud layer also hosts the deployment engine and the domains repository. The edge layer serves as a potential placement layer for one or several application components, depending on the availability and resources of the mobile devices in the mobile ad-hoc cloud layer.

In an ideal scenario, all of the application components are deployed in the mobile ad-hoc cloud layer. However, an insufficiency of resources in the mobile ad-hoc cloud may force the deployment engine to opt for a hybrid placement of the application components between the edge and the mobile ad-hoc cloud. In addition, a zone head and an execution engine are placed in each edge and mobile ad-hoc cloud layers. The functional entities are responsible for finding the appropriate placement for the application components among the 3 layers,

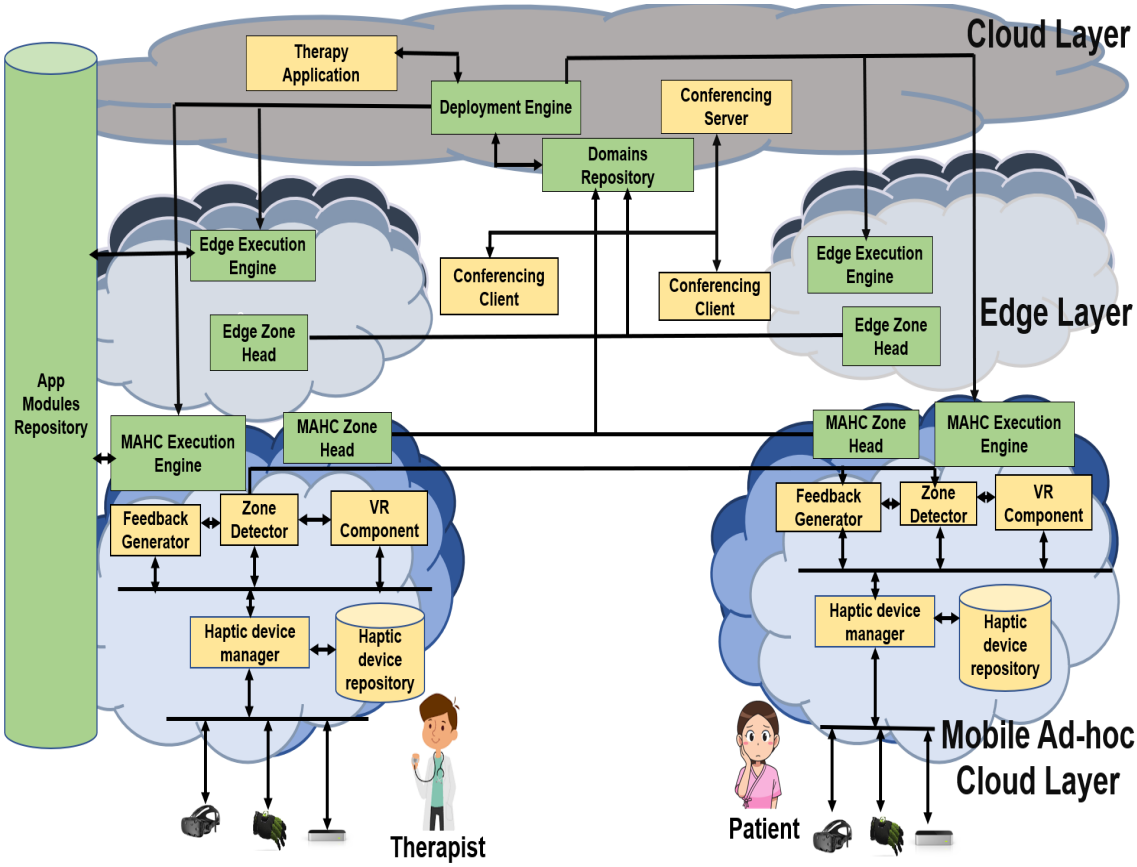


Figure 4.1: Proposed Architecture for Remote Phobia Treatment.

depending on the desired trade-off between cost and latency. A set of tactile devices are also required for the therapist and patient(s) to exchange visual, auditory and tactile information during the therapy session. In our case study, the tactile devices include a VR headset, a hand tracking device as well as a haptic glove to exchange tactile sensations. In Figure 4.1, the application components are in yellow and the functional entities in green.

The application components are described next, followed by a detailed presentation of the functional entities. Then, the different interfaces in the architecture are explained. A set of illustrative scenarios are presented last.

4.2 Application Components

As we mentioned previously, the application components include: the therapy application, the conferencing server, the conferencing client, the zone detector, the feedback generator, the VR component, the haptic device manager, the haptic device repository and the haptic devices. It is worth mentioning that the therapy application and the conferencing server are permanently deployed in the cloud layer, while the conferencing client is permanently placed in the edge. Those components are kept at fixed locations as they require more computational resources compared to the other components, especially for the conferencing server and the therapy application. As for the remaining components, they can be deployed across any of the layers depending on the desired trade-off between cost and latency. Those

components are dynamically deployable as they directly influence latency. Next, we proceed to describe each of the application components.

4.2.1 Therapy Application

This component is the first point of contact with the end-users. It allows the therapist to launch a new therapy session and allows the patients to join the session. It also collects the appropriate information from the end-users (e.g. location) and forwards it to the deployment engine. This information allows the deployment engine to generate the deployment plan (discussed later). Furthermore, it coordinates the visual view of the virtual environment between the therapy session participants.

4.2.2 Conferencing Server

The conferencing server provides an audio-video-text communication channel between the doctor and the patient(s). A conferencing session is created among all the participants at the beginning of the therapy session and before any haptic data is exchanged. This allows the therapist to guide his patient(s) throughout the therapy session, giving them vocal and textual instructions to encourage them to overcome their fears.

4.2.3 Conferencing Client

The conferencing client is provided at the vicinity of each participant in the therapy session to allow them to reach the conferencing server. The interaction between the conferencing server and the conferencing client is meant to facilitate the services that take place before the beginning of the therapy session such as authentication.

4.2.4 Zone Detector

This component subscribes to the hand movement tracking device, via the haptic device manager, and gets notified about the changes in terms of the zone where the hand is located. Each notification includes the new spatial coordinates of the tracked therapist's hand. These coordinates are used to determine the current position of play. In other words, the spatial coordinates indicate the zone of the virtual environment where the therapist's hand is located. Subsequently, this spatial information determines what kind of tactile feedback should be generated at the haptic glove level. For this purpose, the zone detector translates the position to the appropriate formats supported by the VR component and the feedback generator, then forwards the spatial information to these components. For example, at the beginning of a therapy session, when the therapist is still helping the patient(s) to get accustomed to the therapy process, the therapist's hand is still located far from the feared object (i.e. the spider). Therefore, the spatial information sent to the feedback generator does not trigger any tactile sensation on either participants' haptic gloves.

4.2.5 Feedback Generator

After receiving the spatial information from the zone detector, this component determines the appropriate haptic feedback to be generated, based on the zone of the subject/insect that was touched. This is achieved as the feedback generator maps each zone in the virtual environment where the therapy session takes place to its corresponding tactile feedback.

To ensure generating the same tactile sensations at both participants' sides (i.e. therapist and patient), the zone detector on the therapist's side communicates with the feedback generator at the patient's side (in addition to exchanging information with the feedback generator at the therapist's side). This means that both feedback generators receive the same spatial information throughout the therapy session, which leads to generating the same tactile sensations simultaneously at the haptic glove level of each participant.

4.2.6 VR Component

In a similar way to the feedback generator, the VR component leverages the spatial information received from the zone detector to periodically synchronize the virtual environment view in which the therapy session is being conducted. This is achieved via communication with each VR component, which exchanges information with its corresponding VR headset. This ensures that both participants receive similar visual feedback from the virtual environment.

4.2.7 Haptic Device Manager

The haptic device manager serves as the contact point between the application's components and the underlying end-user devices. It abstracts the devices' vendor-related details and offers a uniform interface to the higher-level functionalities to communicate with these devices. Furthermore, it provides a publish/discovery mechanism to allow the end-devices to publish their capabilities and the higher-level functionalities to find the devices of interest. The different publications are saved in the haptic device repository.

4.2.8 Haptic Device Repository

This component serves as a repository for the different haptic devices information. These information includes the device URI, the device type (e.g. VR headset, tracking device, haptic device) as well as the devices' capabilities (e.g. number of actuators in a haptic device). To publish this information, it is initially sent by each device to the haptic device manager before the start of a therapy session. The haptic device manager proceeds subsequently to store this information at the haptic device repository.

4.2.9 Haptic Devices

The haptic devices include the required devices to conduct a therapy session. Typically, those include a VR headset for visual perception of the virtual environment and a haptic glove to receive tactile sensations, and a tracking device to detect the spatial information related to a specific body part (i.e. hand in this use case). As previously described, the zone detector, the feedback generator and the VR component communicate with the hand tracking device, the haptic glove and the VR headset respectively via the haptic device manager.

4.3 Functional Entities

The functional entities include the following architectural components: the deployment engine, the domains repository, the edge execution engine, the edge zone head, the mobile ad-hoc cloud execution engine, the mobile ad-hoc cloud zone head and finally the application

modules repository. As the functional entities are responsible for finding the appropriate placement for the application components depending on the latency/cost trade-off, their placements are predetermined and fixed across the 3 layers. In what follows, a detailed description of each of those functional entities is given.

4.3.1 Deployment Engine

This functional entity is at the heart of the architecture deployment adaptability. As the therapist proceeds to initiate a new therapy session, a request is received at the deployment engine level to start a new therapy session. This leads the deployment engine to get the end-users' locations, which allows it to consult the domains repository to check resources' availability in the end-users' vicinity. Afterwards, the deployment engine, based on the end users' location information and resources' availability, decides on the optimal deployment model for the application components.

As an example, if both the therapist and the patient are in the same resource-sufficient mobile ad-hoc cloud domain, all the dynamically deployable application components (i.e. zone detector, feedback generator and VR component) can be instantiated and deployed on the very same mobile ad-hoc cloud. This will keep the latency at its minimum, and therefore enhance the user-interactivity. The output of this step is a deployment plan that will be communicated to the related edge/mobile ad-hoc cloud domain for execution. The deployment plan simply states where each application component should be deployed based on the end users' locations and the resources available. However, the generated deployment plan might be revisited and the deployment updated when needed. If there is a change in resources' availability or in the end-users' locations (e.g. if the patient has moved to a different edge/mobile ad-hoc cloud domain), for instance, some of those application components might need to be migrated.

4.3.2 Domains Repository

The domains repository serves as a central location where the information about the available edge and mobile ad-hoc cloud domains is stored. Such information includes the geographical location, the number of nodes (we assume each node can host at least one of the application components), as well as the overall capacity of the specific domain. Information related to the overall capacity of an edge/mobile ad-hoc cloud domain includes, for instance, the available bandwidth and the maximum number of application components that can be deployed on that specific domain.

4.3.3 Edge/Mobile Ad-hoc Cloud Execution Engine

The edge/mobile ad-hoc cloud execution engines act as delegates of the deployment engine at their respective layers. Their main responsibility is to identify the actual nodes to host the application components specified in the deployment plan, and then deploy and instantiate the components. The nodes are selected based on the application components resources' requirements and the available nodes' resources.

4.3.4 Edge/Mobile Ad-hoc Cloud Zone Head

The list of nodes and their respective properties are collected and maintained locally in each domain by the domain's zone head. The edge/mobile ad-hoc cloud zone heads are

also responsible for publishing a summary of such information to the domains repository. This information includes the domain's total number of nodes, its geographical location as well as the domain's overall computational capacity. After the execution engines identify the application components and select the nodes on which they should be deployed, they retrieve the appropriate application components from the application modules repository and instantiate them according to the placement defined in the generated deployment plan.

4.3.5 Application Modules Repository

The application modules repository serves as a repository that stores the application components. It spans the three layers, as the dynamically deployable application components might be placed across the three layers depending on the generated deployment plan by the deployment engine. This is also needed for smooth interaction between the deployment engine, the edge/mobile ad-hoc cloud execution engines and the application components. If different versions of the same application component are needed for different layers (e.g. because of the underlying required platform or operating system), these are all added to the repository with the related details and the execution engines select the most appropriate one.

4.4 Interfaces

To solve the problem of the heterogeneity of the end users' tactile devices, we specified previously in our requirements the need to provide high level uniform interfaces. Ultimately, we have identified the need for seven interfaces. First, for the purpose of interaction between each application component and its corresponding tactile device (via the haptic device manager), an interface at the haptic device manager level and another interface at the tactile devices level need to be exposed. Second, because the haptic device manager acts as a liaison between the end users' tactile devices and the application components, an interface needs to be exposed for this communication. Third, the application components interact with each other during the therapy session. Namely, as the zone detector communicates the spatial data to the feedback generator and the VR component, each of those components need to expose an interface in order to establish a uniform communication channel. Finally, because the conferencing server also needs to handle the details of the different participants during a therapy session, an interface should be exposed at the conferencing server as well.

To design those interfaces, we relied on a REST based-approach [64]. We chose REST because it is based on existing web technologies, it provides a uniform interface, and because RESTful web services are lightweight and can be implemented in various types of devices, including small footprint devices, such as cameras and virtual reality headsets that might be needed for a phobia-treatment application. This section describes those interfaces, starting with the interfaces exposed by the haptic device manager. Afterwards, we describe the end user devices' interface. Then, the interfaces for the zone detector, the feedback generator and the VR component are presented. Finally, we describe the interface exposed by the conferencing server.

4.4.1 Haptic Device Manager Interface

The haptic device manager interface allows the tactile devices to register with the device manager and publish their URIs and capabilities. We can differentiate between two types

of tactile devices: the sensor devices and the actuator devices. The sensor devices capture specific events and notify the application components that have subscribed to such events. The hand tracking device, for instance, tracks the hand movement and notifies the zone detector (via the haptic device manager). As those notifications are initially sent to the haptic device manager, this allows the application components to reach and interact with their corresponding devices via the haptic device manager. The detailed design of this interface is given in Table 4.1

Resource	**Description & Functionalities**	**URI & Operations**	**Resource Representation in the Request**	**Resource Representation in the Response**
List of end-devices' registrations (at the haptic device manager level)	A record of all the end users' devices (e.g. hand tracking device) registered for use within the architecture	URI:**/registrations** **POST:** create a new registration	**- device type & URI:** haptic device, VR headset, etc. **- capabilities:** e.g. number of actuators in the haptic glove	The URI of the newly created registration resource **/registrations/ {registrationID}**
A specific end-device registration (at the haptic device manager level)	Registration of a specific device (allows devices to show their availability to the application components and publish their capabilities).	URI: **/registrations/ {registrationID}** **DELETE:** unregister a specific device. **PUT:** change information related to a specific device (e.g. in case of device upgrade).	**- capabilities:** (for PUT operation)	None in case of success OR error description in case of error
List of notifications (At the haptic device manager level)	Record of notifications sent by devices regarding requested changes.	URI: **/notifications** **POST:** send a change notification.	**detected changes:** e.g. the new zone where the therapist's hand is located.	- notification id - therapy session id
A specific notification (At the haptic device manager level)	A specific notification received from a specific device.	URI: **/notifications/ {notificationID}** **GET:** check the notification's information.	None.	- notification details.

Table 4.1: Summary of Haptic Device Manager Interface for Tactile Devices.

In addition, to match each tactile device to its corresponding application component, an interface at the haptic device manager level needs to be exposed for the application components. This interface allows the application components to register for notifications from the tactile devices. Moreover, it provides another functionality to allow each application component to search for the appropriate device. This interface is described in Table 4.2

Resource	**Description & Functionalities**	**URI & Operations**	**Resource Representation in the Request**	**Resource Representation in the Response**
List of subscriptions for notifications (at the haptic device manager level)	Keeps track of all subscriptions issued by an application component (e.g. zone detectors)	URI:**/subscriptions** **POST:** create a new subscription	**- Subscribing application component URI** (e.g. zone detector URI). **- subscription parameters:** (e.g. notifications' frequency).	The URI of the newly created subscroption resource **/subscriptions/ {subscriptionID}**
A specific subscription for notifications (at the haptic device manager level)	Allows a specific subscription from a specific application component	URI: **/subscriptions/ {subscriptionID}** **DELETE:** unsubscribe **PUT:** update an ongoing subscription; e.g. change the notification periodicity.	**- None:** in case of DELETE **- New subscription parameters:** in case of PUT	None in case of success OR error description in case of error.
List of tactile devices (At the haptic device manager level)	Search for a specific tactile device among the registered devices.	URI: **/subscriptions /devices** **GET:** check the registered devices.	**None**	- device id - device type - description

Table 4.2: Summary of Haptic Device Manager Interface for the Application Components.

4.4.2 End User Tactile Devices' Interface

The end user tactile devices should provide a subscription functionality, which allows the haptic device manager to forward the information received from devices to the appropriate application components. For this purpose, those devices offer two registration related resources to allow the application components to subscribe to updates/notifications (via the haptic device manager). In this sense, this interface is partially similar to the previous interface exposed by the haptic device manager for the application components. The details of this interface are given in Table 4.3.

4.4.3 Zone Detector Interface

The zone detector needs to communicate the spatial data to the feedback generator and the VR component. To this end, the VR component and feedback generator need to register with the zone detector. Therefore, the zone detector provides two separate resources, *list of registrations* and *specific registration*, similar to those of provided by the haptic device manager to the tactile devices. In addition, as the zone detector may also receive notifications from the hand tracking device via the haptic device manager, an additional resource needs to provided for this purpose, hence the *list of notifications* resource. This

Resource	**Description & Functionalities**	**URI & Operations**	**Resource Representation in the Request**	**Resource Representation in the Response**
List of subscriptions for notifications (at the end-user's tactile device level)	Keeps track of all subscriptions issued by an application component (e.g. zone detectors)	URI:**/subscriptions** **POST:** create a new subscription	**- Subscribing application component URI** (e.g. zone detector URI). **- subscription parameters:** (e.g. notifications' frequency).	The URI of the newly created subscription resource **/subscriptions/ {subscriptionID}**
A specific subscription for notifications (at the end-user's tactile device level)	Allows a specific subscription from a specific application component	URI: **/subscriptions/ {subscriptionID}** **DELETE:** unsubscribe **PUT:** update an ongoing subscription; e.g. change the notification periodicity.	**- None:** in case of DELETE **- New subscription parameters:** in case of PUT	None in case of success OR error description in case of error.

Table 4.3: Summary of Tactile Devices' Interface .

interface is described in Table 4.4.

4.4.4 Feedback Generator and VR Component Interfaces

Because the haptic device manager facilitates the communication of these 2 components with their corresponding tactile devices (i.e. haptic glove and VR headset respectively), they solely need to expose an interface to allow the zone detector to send them notifications about the new zone where the hand is located. Consequently, to receive notifications, the VR components and feedback generators expose a *list of notifications* resource. This interface is described in Table 4.5. Finally, we should also mention that the aforementioned application components (zone detector, feedback generator and VR componet) expose a common interface to the haptic device manager, which allows the latter to forward the incoming registrations from the tactile devices to their corresponding application components. To this end, this interface is similar to the interface exposed by the haptic device manager to the tactile devices.

4.4.5 Conferencing Server Interface

As for the conferencing sever interface, it should allow the conferencing client at each participant's side to either create a therapy room where he can be joined by other participants of the same therapy session, or simply join an already existing therapy room. Therefore, this interface needs to expose a *list of participants* resource, where each participant can register as a new participant in the conferencing server, as well as a *list of rooms* resource that allows a patient either to check before choosing the specific therapy room where he is supposed to participate, or to create a new room where he can be joined

Resource	**Description & Functionalities**	**URI & Operations**	**Resource Representation in the Request**	**Resource Representation in the Response**
List of registrations (at the zone detector's level)	A record of all feedback generators and the VR components registered within the specific zone detector.	URI:**/registrations** **POST:** create a new registration for the feedback generator or the VR component.	**- component URI** **- frequency of updates**	The URI of the newly created registration resource **/registrations/ {registrationID}**
A specific component registration (at the zone detector's level)	Allows a specific feedback generator or VR component to unregister	URI: **/registrations/ {registrationID}** **DELETE:** unregister a specific feedback generator or VR component.	**- None**	None in case of success OR error description in case of error.
List of notifications (at zone detector level)	A record of all the notifications sent from the haptic device manager.	URI:**/notifications** **POST:** send a new notification from the haptic device manager.	**- timestamp** **- spatial coordinates/zone.**	The URI of the newly created notification resource **/notifications/ {notificationID}**

Table 4.4: Summary of Zone Detector's Interface.

by a therapist and potentially other patients in the case of a collective therapy session. The summary of this interface is illustrated in Table 4.6.

Resource	**Description & Functionalities**	**URI & Operations**	**Resource Representation in the Request**	**Resource Representation in the Response**
List of notifications (at the feedback generator/VR component level)	A record of all the notifications sent from the zone detector.	URI:**/notifications** **POST:** send a new notification from the zone detector.	**- timestamp** **- adapted spatial coordinates/zone.**	The URI of the newly created notification resource **/notifications/ {notificationID}**

Table 4.5: Summary of Feedback Generator/VR Component Interface.

4.5 Illustrative Scenarios

To get a good grasp of the different architectural components' interactions, we illustrate a set of different scenarios. The first scenario describes the interactions between the functional entities in order to choose the appropriate placement of the application components. As for the second scenario, it describes the application components' interaction after their deployments, before and during an individual (one to one) therapy session. The third section describes the same scenario but for a collective (one to many) therapy session. the fourth and final scenario illustrates the interactions leading to the establishment of a

Resource	Description & Functionalities	URI & Operations	Resource Representation in the Request	Resource Representation in the Response
List of conferencing rooms (at the conferencing server's level)	A record of all conferencing rooms used for therapy.	URI:**/rooms** **POST:** create a new conferencing room for therapy. **DELETE:** remove an existing conferencing room after the end of a therapy session.	**- room name**	The URI of the newly created room resource **/rooms/{roomID}**
List of participants in a specific room (at the conferencing server's level)	Registration of a specific participant (therapist or patient) in a specific conferencing room.	URI: **/rooms/ {roomID}/ participants** **POST:** register a specific participant in a specific conferencing room. **DELETE:** unregister a specific participant from a specific conferencing room.	**- Participant username**	The URI of the newly created participant resource. **/rooms/{roomID}/ participants/ {participantID}**

Table 4.6: Summary of the Conferencing Sever Interface.

conferencing session between the therapist and the patient(s). Each of those scenarios is described and illustrated with a sequence of interactions in what follows.

4.5.1 Functional Entities Interactions

Figure 4.2 summarizes the sequence of the key interactions between the functional entities, leading to the deployment of the application components and subsequently the initiation of a new therapy session.

At first, the mobile ad-hoc cloud and edge zone heads publish the information related to their specific domains to the domains repository (step 1). We assume that each edge/mobile ad-hoc cloud domain has a group owner node, which is running the zone head functional entity. All the nodes in a given domain communicate their information to the group owner node. The latter stores the information locally and publishes an aggregated version to the domains repository. When the therapy application receives a request to initiate a new session, it forwards this request to the deployment engine (step 2). The deployment engine then proceeds to fetch the relevant information about the available edge/mobile ad-hoc cloud domains in the vicinity of the session's participants (step 3 and 4). This information, as previously mentioned, includes the location and the number of application components that can be hosted. Afterwards, the deployment engine computes the best deployment plan (step 5), and requests the selected edge/mobile ad-hoc cloud execution engines to instantiate the appropriate application components (step 6). The execution engine downloads the relevant application components from the application modules repository (step 7) and instantiates them into the chosen nodes (step 8). Finally, a notification of successful deployment is sent

from the edge/mobile ad-hoc cloud execution engine (depending on where the components were deployed) to the deployment engine (step 9).

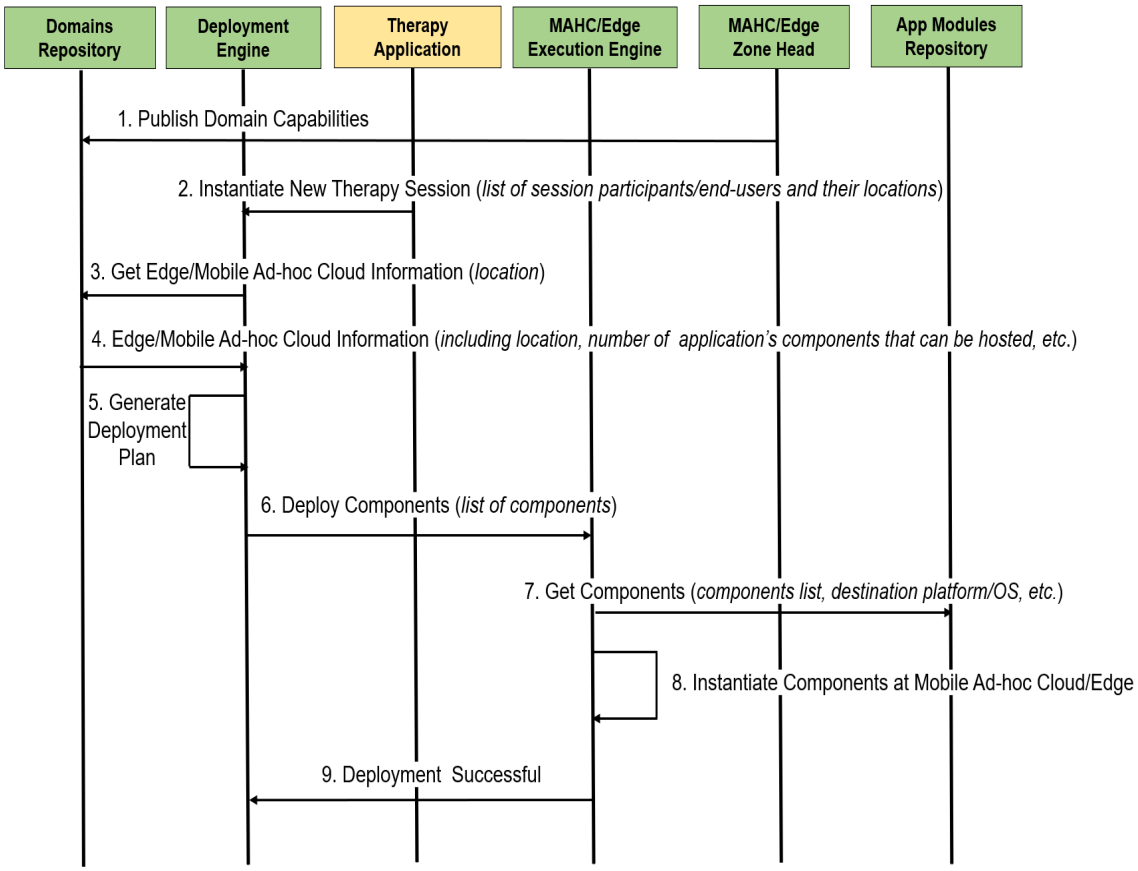


Figure 4.2: Sequence of Interactions among the Functional Entities to Initiate a New Therapy Session.

4.5.2 Application Components Interactions for an Individual Therapy Session

In this section, we consider a one-to-one spider phobia treatment use case as an illustrative scenario, where the therapist and the patient are at separate edge domains. Figure 4.3 illustrates the interactions for this scenario. For the sake of simplicity, we explain and illustrate partially the devices to components communication. The general scenario of a therapy session takes place as follows: First, the end users' tactile devices at each end (therapist's end and patient's end) register with the haptic device manager (steps 1 and 2). We assume that the tactile devices are already configured and put online. Afterwards, those registrations requests are sent by the haptic device manager to the appropriate application components (step 3). To start a new therapy session, the therapist launches the therapy application and adds the participants (we assume that all the participants are already online). This will trigger: (1) the zone detector (and the other application components) to register with the haptic device manager to receive location updates from the hand-tracking device (and the other tactile devices) (steps 4 and 5); (2) the haptic device manager to

forward this registration to the hand tracking device side (and other tactile devices side) (step. 6); and (3) the VR component and feedback generator to register with the zone detector (steps 7 and 8). When the therapist starts moving his/her hand, the hand-tracking device detects the movement and sends the new hand coordinates to the haptic device manager, which will then forward them to the zone detector (steps 9 and 10). The latter calculates and adapts the spatial information and notifies the VR component and feedback generator about the update (steps 11 and 12). These devices (VR headset and haptic glove) will then render the appropriate effect for both actors. For the virtual environment, this effect will translate to a new updated VR view. As for the haptic feedback, the glove will generate the appropriate tactile sensation corresponding to the zone where the therapist's hand has moved (steps 13 and 14).

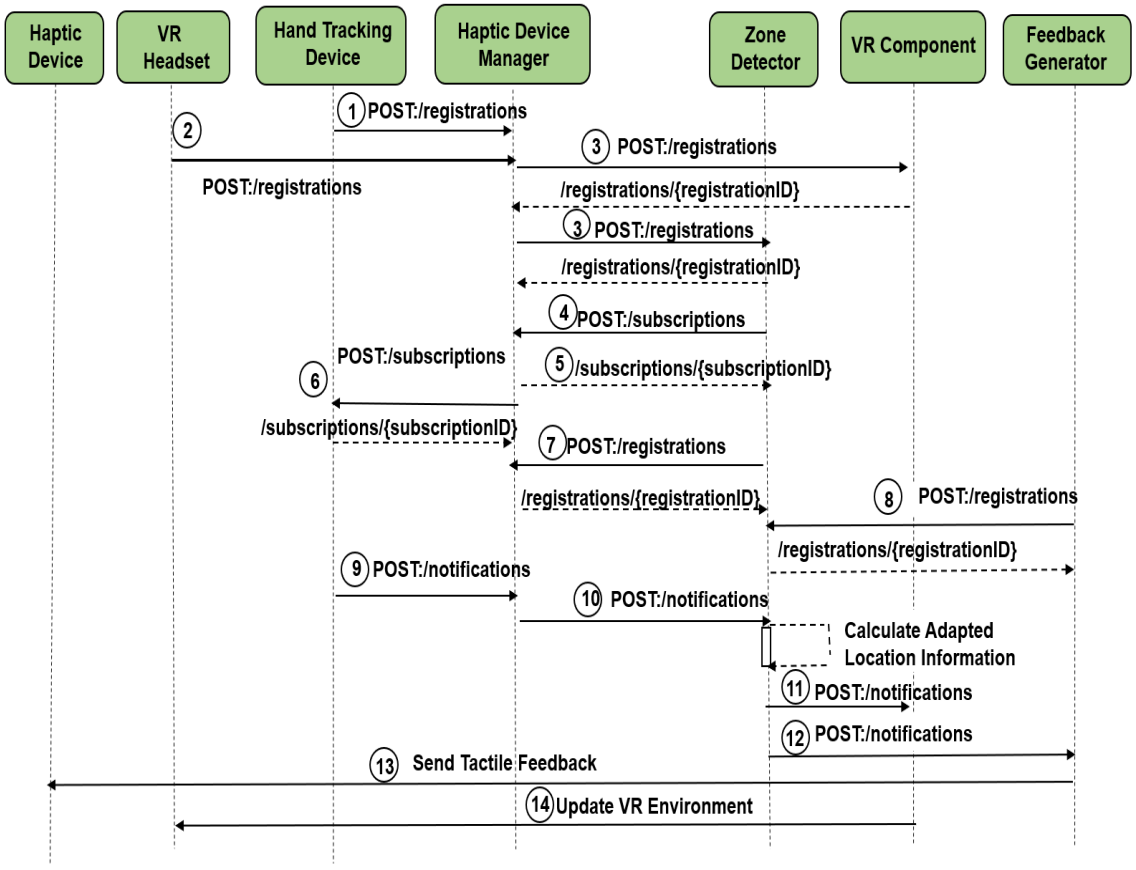


Figure 4.3: Sequence of Interactions among the Application Components before and during a New Individual Therapy Session.

4.5.3 Application Components Interactions for a Collective Therapy Session

Although the general scenario for a collective therapy session is similar to the individual therapy session scenario, the application components number increases proportionally to the number of participants involved in the therapy session. To this end, the occurring

interactions between the application components before and during the therapy session remain similar to what was already illustrated in Fig 4.3.

4.5.4 Conferencing Session Interactions

To ensure the efficiency of the therapy session, it is equally important to allow the therapist and the patient to exchange visual and auditory information. This functionality is provided via the conferencing server in our architecture. The conferencing server is reachable in our application via the therapy application, which does not only provide the virtual environment where the therapist and the patient(s) interact, but it does also serve as an intermediary between the participants and the conferencing server. As soon as the participants access the conferencing server, the latter requests the participants to enter a username as well as the conferencing room they wish to join. We assume the conferencing room where the therapist interacts with his patient(s) is provided before the start of the therapy session. The sequence of interactions illustrating the establishment of a conferencing session is shown in Fig. 4.4.

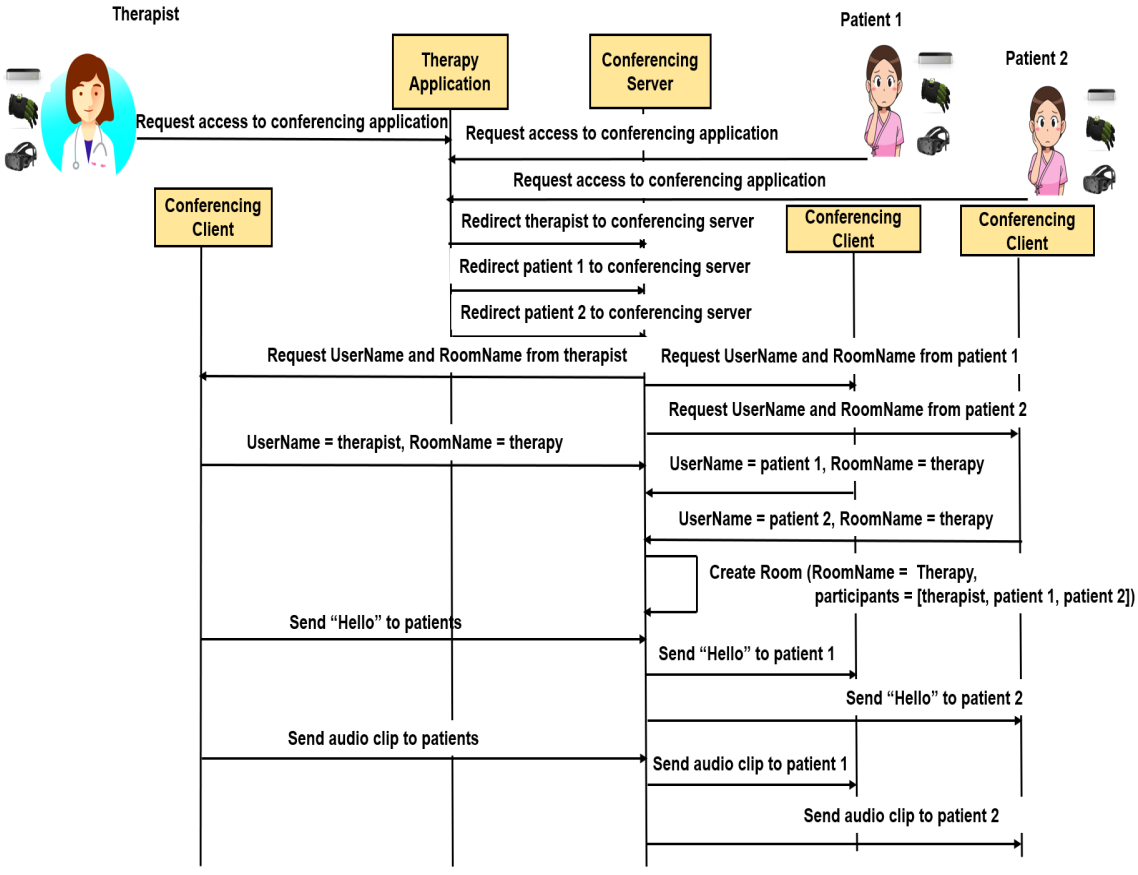


Figure 4.4: Sequence of Interactions to Establish a Conferencing Session.

4.6 Evaluation of the Proposed Architecture Against the Requirements

In this subsection, we proceed to evaluate the proposed architecture against the requirements derived from the use case in the previous chapter.

First, the proposed architecture incorporates a set of tactile devices, including a VR headset, a hand tracking device as well as a haptic glove. In addition, a conferencing server is among the application components, allowing the therapist and the patient(s) to establish conference calls before and during the therapy session.

Therefore, the proposed architecture allows exchange of tactile sensations/information via the haptic glove, as well as visual and auditory communication during conference calls.

Second, the architecture relies on the deployment engine and the deployment plan that it generates to find the appropriate placement for the application components (with the help of the other functional entities). Moreover, the architecture leverages the edge and mobile ad-hoc cloud layers to deploy the components as close as possible to the end user.

Thus, an end to end latency in the order of a few milliseconds can be obtained (obtained later in the experimental results), which satisfies the second requirement of the architecture.

Third, the functional entities' main responsibility is to fetch the information related to each participant and instantiate the application components in his/her vicinity.

To this end, either an individual or a collective therapy session can be conducted, as the deployment engine can simply download the application components from the application modules repository and instantiate them close to each participant in the therapy session, which satisfies the third requirement.

Finally, a discussion related to the heterogeneous nature of the tactile devices led to introducing the haptic device manager component, which acts as an intermediary between the tactile devices and their corresponding components. This component provides a high level REST-based interface for the devices. In addition, a set of other interfaces were also defined, including interfaces at the tactile devices level.

Those interfaces were provided with a detailed description of their functionalities. This means that the fourth and final requirement related to providing high level uniform interfaces was also satisfied.

4.7 Conclusion

In this chapter, we presented the proposed architecture, with a detailed description of each architectural component across the 3 layers. A set of high level interfaces were also designed and explained. Afterwards, various illustrating scenarios were explained and illustrated. In the last section of this chapter, we evaluated the proposed architecture against the

requirements that we derived previously from the use case. As a result, we deduced that our architecture fully satisfies those requirements. In the next chapter, the implemented proof of concept prototype of the proposed architecture will be described and discussed. We will also discuss the obtained experimental results and the insights gained from these results.

Chapter 5

Validation of the Architecture

In this chapter, we present comprehensively the proof of concept prototype architecture. Afterwards, a brief description of the implemented prototype is given. Then, we describe the prototype's setup and the performance metrics considered to evaluate it. Finally, we discuss the results of the various experiments and discuss the gained insights from these results. A concluding section summarizes this chapter at the end.

5.1 Prototype Architecture

A proof of concept prototype of the architecture that includes the three layers was implemented. The scenario described in the use case and state of the art chapter was implemented for group therapy. This section follows a bottom up approach to present the details of the prototype and the technologies used in its implementation, starting with the end users' tactile devices. We also discuss how the mobile ad-hoc cloud layer and the application components deployment process was realized. Fig. 5.1 illustrates the prototype architecture with a specific application components' placement, where the VR component, the zone detector and the feedback generator are deployed on the mobile ad-hoc cloud, while the conferencing server and the therapy web application are placed in the cloud. In our prototype, we assumed that the mobile ad-hoc cloud had already been formed. The implemented functionalities and architectural components are described in what follows. The application components are described first. Then, we proceed to describe the functional entities.

5.1.1 Application Components

In this section, we begin with a description of the tactile devices. Then, the implemented application components are presented and explained.

5.1.1.1 End User Tactile Devices

In this subsection, we describe the end users' tactile devices used in the prototype. As previously mentioned, 3 types of devices were used: VR headset for VR rendering, haptic glove for haptic communication and finally a hand movement tracking device to determine the zone where a participant's hand is located in the virtual environment.

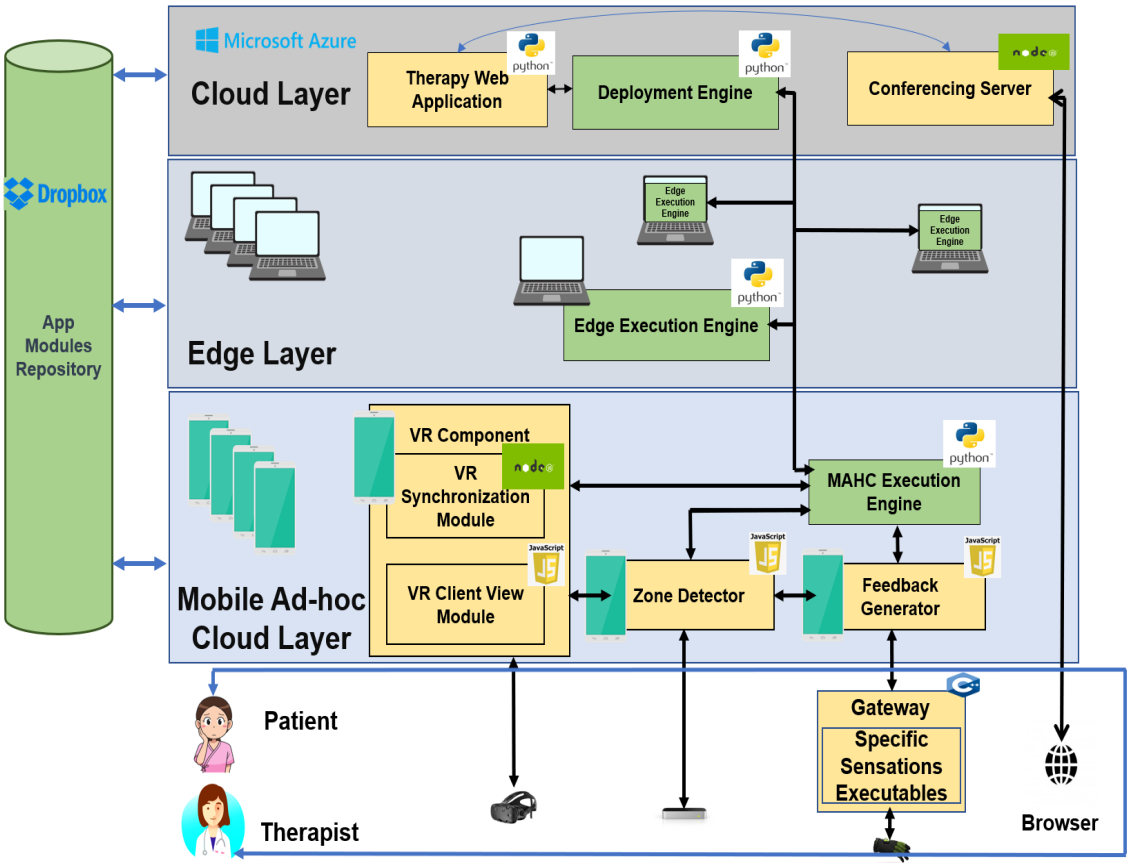


Figure 5.1: Prototype Architecture.

A. VR Rendering

We used HTC Vive headset for VR rendering. The HTC Vive is a virtual reality headset developed by HTC and Valve Corporation. The headset uses "room scale" tracking technology, allowing the user to move in 3D space and use motion-tracked handheld controllers to interact with the environment. HTC Vive supports a variety of technologies including C++, C, Javascript and Unity for VR games [65].

B. Haptic Communication

Gloveone haptic gloves were used as wearable haptic devices to render haptic sensations. Gloveone is a wearable vibrotactile device that was designed by the spanish company Neurodigital Technologies in 2016. It has 10 vibrators, including 5 on each finger while the remaining 5 are located on the palm. It supports USB connection as well as Bluetooth connection. In addition to the supporting software (NDSuite) that comes with those gloves, they are directly programmable via a C++ and C APIs. They also have a specific Unity SDK for games and VR applications [14].

C. Hand Movement Tracking

Leap Motion device was used for hand movement tracking. Leap Motion is a hand tracking device that was designed by the Leap Motion Company. It is compatible with a variety of

Figure 5.2: HTC Vive VR Headset.

VR headsets including HTC Vive and Oculus Rift. The Leap Motion system recognizes and tracks hands, fingers and finger-like tools. The device operates in an intimate proximity with high precision and tracking frame rate and reports discrete positions, gestures, and motion. The Leap Motion controller uses optical sensors, which have a field of view of about 150 degrees, and infrared light. Leap Motion can be either connected to a computer or a VR headset via a USB cable. In terms of supporting software, this device has a specific Unity SDK for VR and games, as well as a Javascript Library to integrate the device in web applications/games (LeapJS) [66].

Figure 5.3: Gloveone Haptic Gloves.

5.1.1.2 Feedback Generator

The feedback generator was implemented using JavaScript. It communicates with the haptic gloves via a local gateway running on the localhost of each participant, as the used haptic gloves do not have a web API and cannot be accessed remotely. The gateway was implemented based on Node.js and C++. In addition, the feedback generator relies on web sockets to establish a communication channel with the zone detector, where the spatial data related to location of the participant's hand is being sent.

5.1.1.3 Zone Detector

This software module uses LeapJS API to communicate with the hand tracking device and detect the hand position. Afterwards, the hand coordinates are sent to the VR client view module. Moreover, the feedback generator is notified when the hand is at a zone that should generate a specific feedback.

5.1.1.4 VR Component

The VR component is implemented in two parts. First, a Node.js-based module responsible for VR synchronization among the participants was implemented, consequently ensuring that all the connected participants see the same virtual objects in the shared virtual environment. Second, we implemented a client view module responsible for rendering the current VR view to the participants when they login to the application on the browser. The latter uses the WebVR API to communicate with the VR headset and render the VR scene to the end users through the headset. WebVR is an open specification that provides interfaces to various VR headsets, allowing the development of VR applications running on a browser. The synchronization module also handles the connection related issues, and notifies the connected participants when a new participant joins the therapy session.

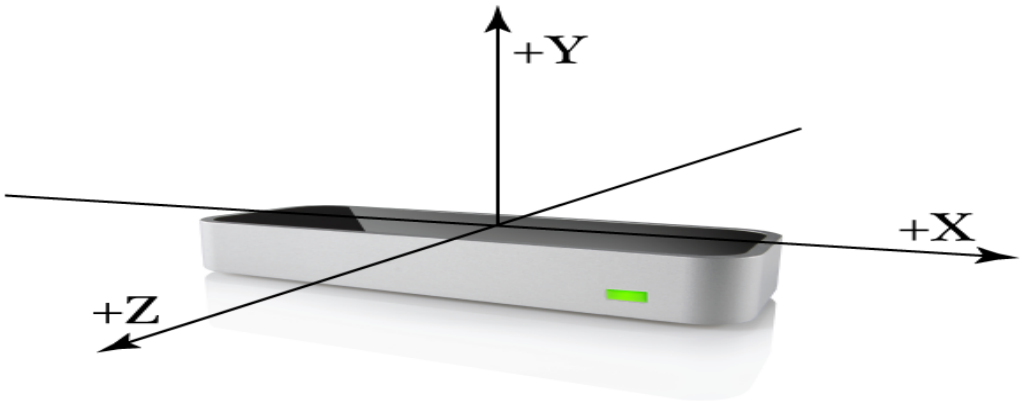


Figure 5.4: Leap Motion Hand Tracking Device.

5.1.1.5 Conferencing Server

This component was implemented as a separate Node.js server that can be accessed from the therapy web application. It was deployed in the cloud and Microsoft Azure was used as a cloud platform to host this server [67]. The conferencing server uses web

real time communication (WebRTC) for visual, auditory and text exchange between the participants. WebRTC is an open-source set of protocols and APIs for peer-to-peer real time communication, which can run directly from a web browser.

5.1.1.6 Therapy Web Application

The therapy web application, which was also deployed on the cloud using Microsoft Azure, was implemented as a web server using the python-based web framework Flask. The therapy web application's main responsibility is to receive the request to start a therapy session from the therapist. In addition, it communicates with the VR synchronization module to allow addition and removal of virtual objects from the VR scene. It also allows the participants to access the conferencing application. All those functionalities can be triggered using the web-based user interface provided by the therapy web application, when the participants access the application on the browser.

5.1.1.7 Haptic Device Manager

The functionalities of the haptic device manager were implemented as part of the feedback generator, zone detector, and VR client view modules. This integrated implementation was mainly due to the limitations of the used devices, especially the haptic glove that does not offer a Web API. As for the Haptic Device Repository, it was not needed in our implementation as we the application components are communicating with an individual device each.

5.1.2 Functional Entities

In this section, we describe the functional entities and their respective roles in the deployment and placement of the application components described in the previous section. We assume that the discovery and publishing of the nodes in the edge and mobile ad-hoc layers has already taken place. Therefore, the zone heads in both layers and the domains repository were not implemented.

5.1.2.1 Deployment Engine

The deployment engine was implemented as a python web server running in the cloud, using python web framework Flask. Microsoft Azure was used as a cloud platform to host this web server. As the information repository and the edge/mobile ad-hoc cloud zone heads were not implemented, the IP addresses of the edge and mobile ad-hoc cloud devices were hard-coded in the deployment engine. To this end, the generated deployment plan specifies the layer where to deploy each of the application components. To be more specific, as soon as the deployment engine, which is running on the cloud, receives a request from the therapy application to start a new therapy session, it issues a request to the execution engine at both layers (edge and mobile ad-hoc cloud) to instantiate the specified application components. To specify the layer where each component should be deployed, a JSON file containing the components that should be deployed/instantiated is sent with the request. The JSON file has a basic format of a single key value pair: 'Components': [List of Components to be deployed]. If the list is empty, no component should be deployed in the layer of the execution engine receiving the request.

5.1.2.2 Edge/Mobile Ad-hoc Cloud Execution Engine

As for the edge/mobile ad-hoc cloud execution engines in the bottom 2 layers, they were also implemented as web servers using python web-framework Flask. For simplicity and accessibility purposes, the edge and mobile ad-hoc cloud setup was under the same local private network, as each device was assigned a static IP address within the network (2 static IP addresses for the 2 mobile phones forming the mobile ad-hoc cloud, and 2 static IP addresses for the 2 laptops used as edge servers). Each aforementioned device runs an instance of the execution engine locally, which makes it reachable at the same IP address of the device. While running the execution engine on the laptops used as edge nodes was straightforward, we opted for the Linux Deploy android application (available at Google Play Store) to enable running the execution engine and the application components on the mobile devices [68]. This Android application gives the possibility to run virtual Linux environments with graphical and command line interfaces on top of Android operating systems. Figures 5.5 ((a)) and 5.5 ((b)) illustrate the usage of Linux Deploy inside a Samsung J7 mobile device for our prototype. When an execution engine in a specific layer receives the request from the deployment engine, it proceeds to download the application components from the application modules repository, and then, depending on the list of components named in the JSON file, the execution engine on each device instantiates the downloaded components on the local laptop/mobile phone.

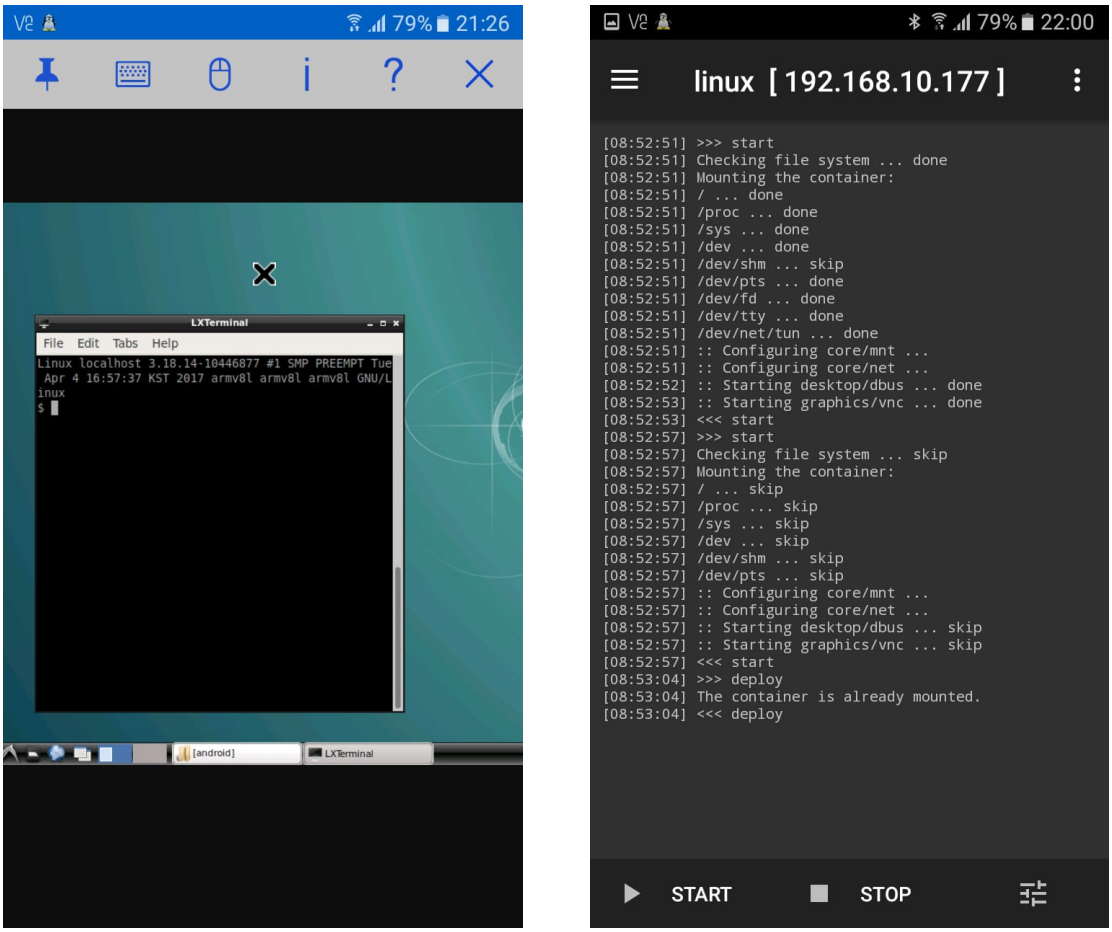
5.1.2.3 Application Modules Repository

Dropbox was used as the application modules repository [69]. Although the components are downloaded initially from the cloud, a copy of those components is kept at the edge nodes and mobile ad-hoc cloud devices whenever a new therapy session needs to be established between the therapist and other patients.

5.2 Description of the Implemented Prototype

In this section, we proceed to describe the implemented prototype from a functional perspective. To this end, we will present the various functionalities that it provides with a special focus on the application components. After the deployment of the application components (either across different layers or in a single layer), and the patient(s) joining the therapist in virtual environment, the therapist can finally proceed to communicate with the patient using the conferencing service provided via the therapy web application, as illustrated in Fig 5.6. Afterwards, the therapist and the patient(s) can start interacting inside the virtual environment, starting with the therapist adding a butterfly to the virtual environment, and interacting with it following the steps previously described, before moving to add a spider and help the patient to interact with it to overcome his phobia.

Figure 5.7 illustrates the virtual environment containing both a spider and a butterfly during a therapy session. As shown in the figure, the Add Spider/Butterfly buttons allow the therapist to enhance the initial (empty) virtual environment with virtual insects, exposing the patient(s) visually to the insects before encouraging him/her to interact with them. The Delete Spider/Butterfly buttons, in contrast, allow the therapist to remove a virtual insect from the virtual environment, in case the patient(s) show that they are not ready yet to be exposed to those insects. The therapist can equally use this button if he feels that the presence of one of those virtual insects is no longer required. As we mentioned previously,



((a)) Debian Distribution on top of Linux Deploy ((b)) Linux Deploy Environment

Figure 5.5: Linux Deploy Usage on a Samsung J7 Mobile Phone.

the Create/Join Conference button allows the therapist to create a new conferencing room where he can communicate with his patient(s), and the patient(s) to join him there once the room is created.

5.3 Performance Measurement

In this section, we begin with a description of the performance metrics. Then, we present the considered performance use cases for conducting the measurements, before proceeding to describe the experiment setup. Afterwards, the obtained results are presented and analysed. Finally, we conclude the section with a discussion of the gained insights from the obtained results.

5.3.1 Performance Metrics

Two performance latency-related metrics were selected for measurement:

First, we evaluated the performance in terms of "the therapy application to participant" latency. We define this latency as the time difference between the timestamp of the hand's

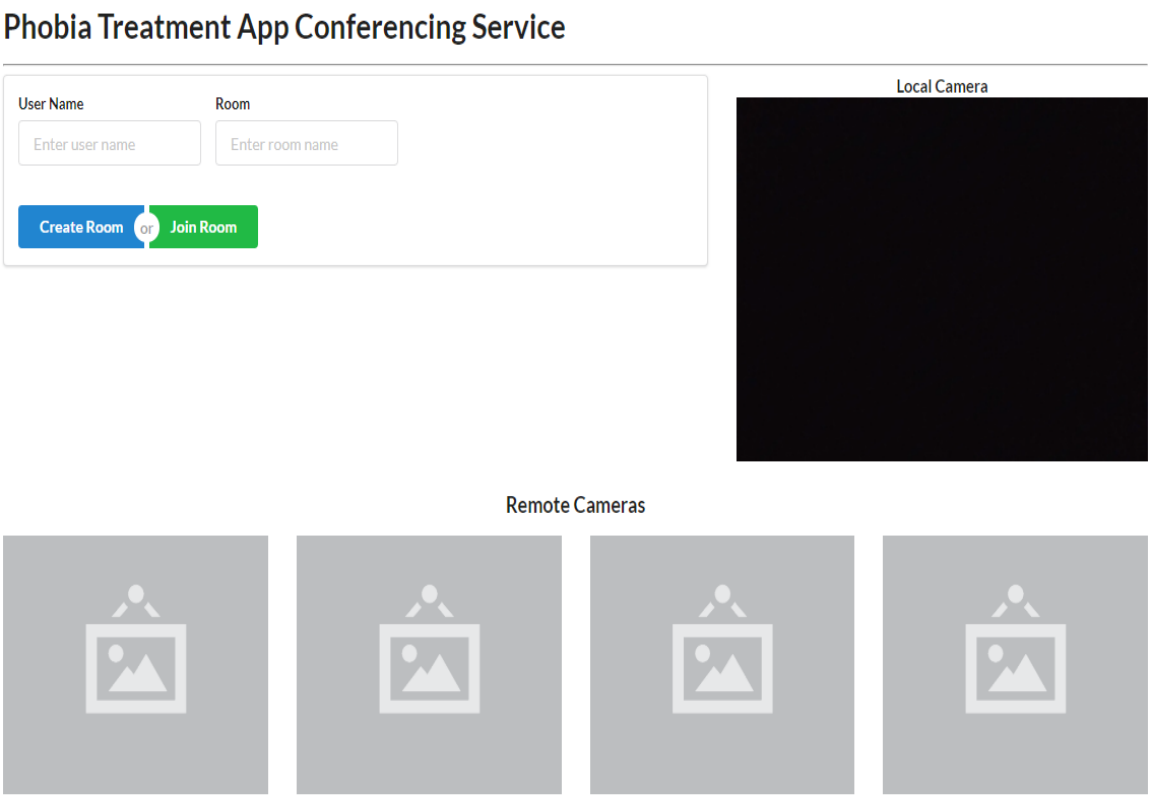


Figure 5.6: Phobia Treatment Application Conferencing Service.

collision with a virtual object (e.g. the spider) in the VR environment, and the reception of the resulting haptic sensation at the participant's (i.e. therapist or patient) glove. This latency is used mainly to measure the required time delay to communicate with the haptic device. We also measured this latency for the purpose of exploring whether it can impact negatively the end to end latency.

Second, the prototyped architecture was also measured in terms of the end to end latency. The end-to-end latency is defined as the time difference between the reception of a tactile sensation at the therapist's glove (from the feedback generator after the collision of his hand with a virtual object) and the reception of the same sensation at the patient's glove (from the therapist's side). Measurements for this latency were made to explore whether the prototyped architecture satisfies the ultra low latency requirement, as well as observing how this latency is impacted by the variation of the application components placement among the 3 layers (among other criteria discussed later).

5.3.2 Performance Use Cases

The performance use cases are chosen to showcase the impact on latency changes according to three criteria: the distance between the therapist and the patient (criterion 1), the underlying networking infrastructure and machine characteristics (criterion 2), and the deployment pattern (criterion 3). To take measurements for the first criterion, the therapist is kept in the same edge domain while the patient is moved between four locations, Montreal/Local, Vancouver, Calgary and Waterloo. SAVI testbed was used for those locations using machines with the same characteristics. For the second criterion, the

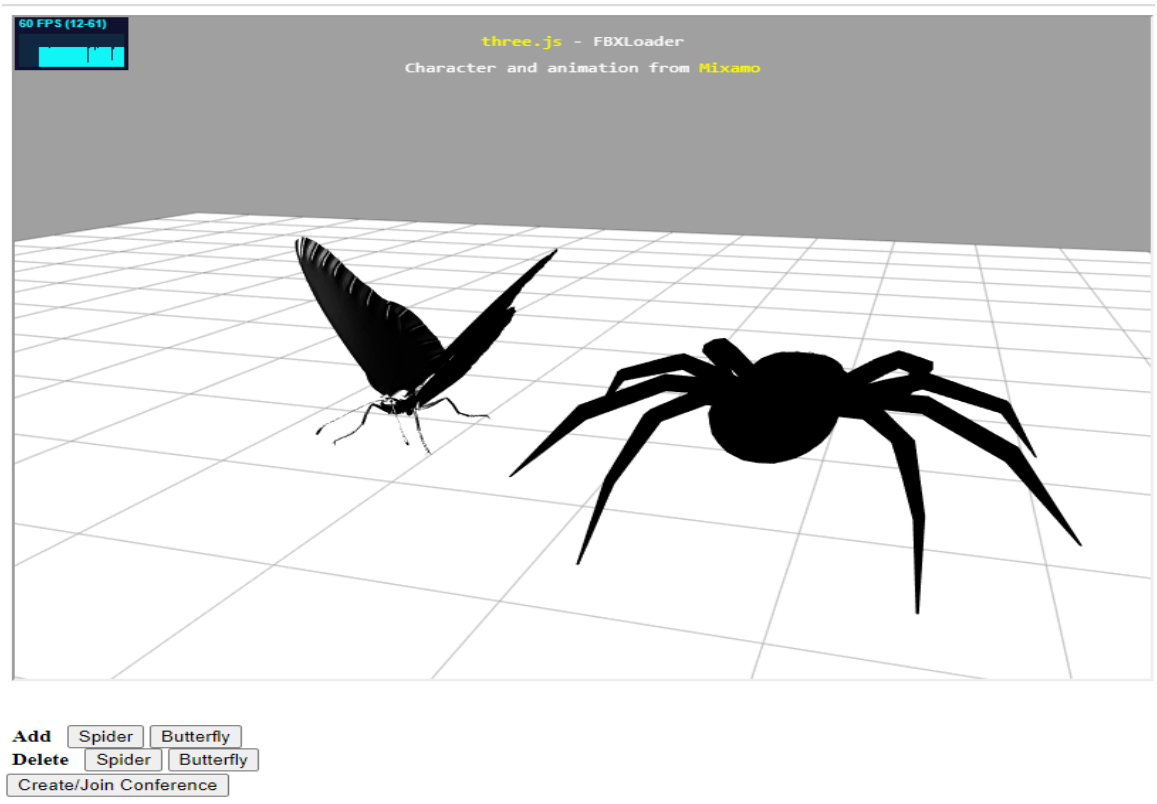


Figure 5.7: Phobia Treatment Web Therapy Application.

therapist is again kept in the same location, but the patient is run in different datacenters under different cloud providers and with different machines characteristics. For criterion 3, the therapist and patient are in the same edge domain, while their respective application components are deployed following the five patterns in Table 5.1.

5.3.3 Experiment Setup

Due to the difficulty of synchronizing several (local and remote) machines, and other issues related to getting the exact timestamp when the hand has moved to a zone that must trigger a feedback, the application components were adapted to allow measuring the end to end latency. Indeed, the hand tracking device updates the hand position 60 times per second, and not every update corresponds to a new zone or to a zone that can generate a feedback (the new zone might not contain any virtual object). Therefore, an extra functionality was added to the therapy application to allow the participants to simulate a hand move in the virtual environment.

To measure the end to end latency, a new hand move is triggered from the therapist's machine via the browser. The measurement then starts from the time the therapist feels the according feedback. This simulates the real-life scenario, where the therapist touches the spider, feels the feedback, then the same feedback is shared with the patient. In spite of these slight complications, measurements for both latencies were made. For this purpose, 8 machines are used for the prototype. The location, characteristics, and architectural components run in each machine are described in Table 5.2. The table also shows the details related to the different architectural components' placement among those machines.

Pattern	Description
P1	All the dynamically deployable application components are running on the mobile ad-hoc cloud.
P2	The zone detector and the feedback generator are running on the edge and the rest of the components on the mobile ad-hoc cloud.
P3	The zone detector and the feedback generator are running on the mobile ad-hoc cloud and the rest of the components on the edge.
P4	All the dynamically deployable application components are running on the edge.
P5	All the dynamically deployable application components are running on the cloud.

Table 5.1: Criterion 3 Deployment Patterns.

5.3.4 Results and Analysis

Figures 5.9, 5.10 and 5.11 show the obtained results. For the first criterion (Figure 5.8), the focus is on the variation of the distance between the therapist and the patient. To this end, both latencies were measured in this case. SAVI testbed was used to make these

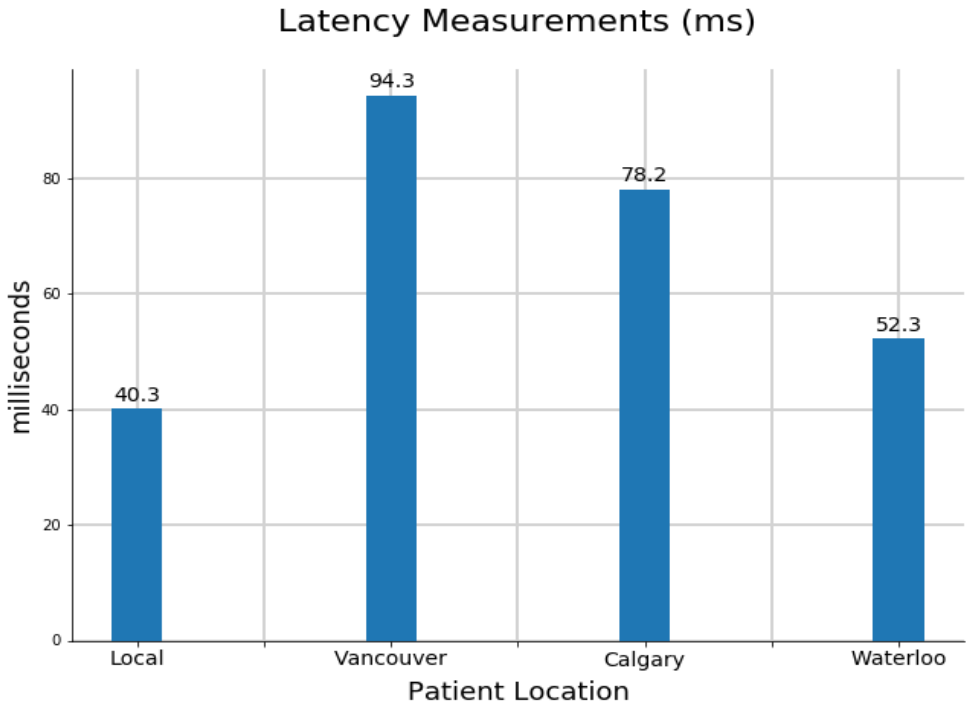


Figure 5.8: End to End Latency Variation in terms of Distance between the Therapist and the Patient (criterion 1).

measurements. Unsurprisingly, the latency increases with the distance and it fluctuates between 40ms and 95ms.

For the second criterion, machines 1 and 2 are used for the local setup. Machines 1, 3 and 4 are used for the other locations. The therapy web application is running in machine 5. The obtained results in Fig.5.10 show that the end to-end-latency is the highest in the local setup (minimum distance between the therapist and the patient) and the smallest in the Toronto setup (the maximum distance). This can be explained by the high-quality networking infrastructure used in Microsoft Azure, and the limited bandwidth of the local network. The "therapy application to participant" latency shows barely any difference regarding where the application web server is deployed. This is mainly because the feedback generator is running on the browser at the client side.

The last use-case investigates the impact of different deployment patterns (shown in Table 5.1) on both latencies, and the added value of placing the application components closer to the participants (i.e. at the mobile ad-hoc cloud layer). We were mainly interested in separating the zone detector and the feedback generator from the remaining application components, as these are the key components responsible for rendering tactile sensations. In a similar way to the previous scenario, the "therapy application to participant" latency does not show a significant difference (Fig. 5.10). In contrast, the end to end latency has significantly improved in P1, where all the components are deployed in the mobile ad-hoc cloud layer. Unsurprisingly, the highest latency was obtained when all the application

Machine Number	Location/Provider	Role	Characteristics
Machine 1	Montreal/ Home Private Network (edge domain 1).	**Therapist:** feedback generator + zone detector + VR component + conferencing client + edge execution engine.	**- OS:** Windows 10 Home 64 bit **- CPU:** Intel Core i5-8300H 2.30 GHz **- Memory:** 8 GB **- Browser:** Google Chrome (conferencing client)
Machine 2	Montreal/ Home Private Network (edge domain 1).	**Patient 1:** feedback generator + zone detector + VR component + conferencing client + edge execution engine.	**- OS:** Windows 10 Home 64 bit **- CPU:** Intel Core i7-8550U, 1.80 GHz. **- Memory:** 16 GB **- Browser:** Google Chrome (conferencing client)
Machine 3	Montreal/ Amazon Web Services (edge domain 2).	**Patient 2:** feedback generator + zone detector + VR component + conferencing client + edge execution engine.	**- OS:** Ubuntu Server 16.04 LTS 64 bit **- CPU:** Intel Xeon CPU E5-2686 v4, 2.3 GHz. **- Memory:** 8 GB. **- Browser:** Firefox (conferencing client)
Machine 4	Toronto/ Microsoft Azure (edge domain 3).	**Patient 3:** feedback generator + zone detector + VR component + conferencing client + edge execution engine.	**- OS:** Windows 10 Home 64 bit **- CPU:** Intel Xeon CPU E5-2673 v3, 2.4 GHz. **- Memory:** 8 GB. **- Browser:** Google Chrome (conferencing client)
Machine 5	Virginia/Microsoft Azure (Cloud Domain).	**Therapy Web Application.**	**- OS:** Windows 10 Pro 64 bit. **- CPU:** Intel Xeon CPU E5-2673 v3, 2.4 GHz. **- Memory:** 8 GB.
Machine 6	Virginia/Microsoft Azure (Cloud Domain).	**Conferencing Server + Deployment Engine.**	**- OS:** Windows 10 Pro 64 bit. **- CPU:** Intel Xeon CPU E5-2673 v3, 2.4 GHz. **- Memory:** 8 GB.
Machine 7	Samsung Galaxy J7 (Mobile Ad-hoc Cloud Domain 1).	**Therapist:** feedback generator + zone detector + VR component + conferencing client + mobile ad-hoc cloud execution engine.	**- OS:** Android 6.0.1 (Marshmallow). **- CPU:** Octa-core 1.6 GHz Cortex-A53. **- Memory:** 2 GB. **Browser:** Google Chrome for Android.
Machine 8	Samsung Galaxy J7 (Mobile Ad-hoc Cloud Domain 2).	**Patient:** feedback generator + zone detector + VR component + conferencing client + mobile ad-hoc cloud execution engine.	**- OS:** Android 6.0.1 (Marshmallow). **- CPU:** Octa-core 1.6 GHz Cortex-A53. **- Memory:** 2 GB. **Browser:** Google Chrome for Android.

Table 5.2: Prototype Setup Details.

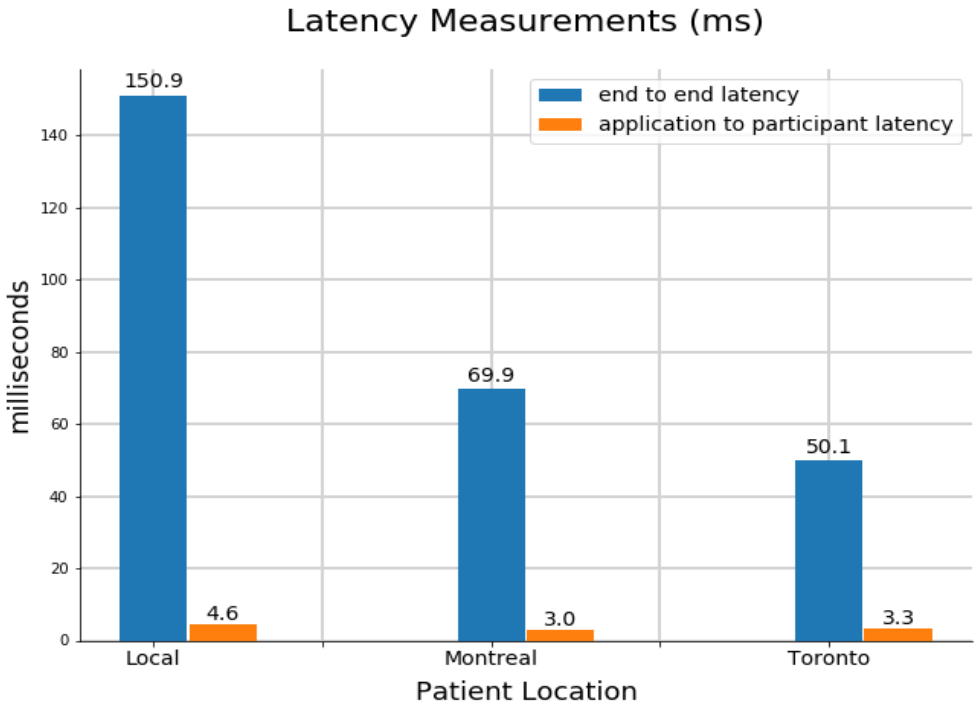


Figure 5.9: Performance Evaluation for both Latencies' Variation in terms of Machine Characteristics and Underlying Networking Infrastructure (criterion 2).

components were deployed on the cloud (P5). Another interesting result is that deploying all the application components in the edge (P4) results in better latency compared to the latency obtained when splitting the application components between the edge and the mobile ad-hoc cloud (P2). This is very likely because all the application components are placed near each other, decreasing therefore the inter-components' communication delays. Also, deploying the zone detector and the feedback generator in the mobile ad-hoc cloud (P3) results in a better performance than deploying them on the edge (P2).

5.3.5 Gained Insights

First, as illustrated in Fig. 5.10, the closer the dynamically deployable application components are to the end user tactile devices, the lowest is the latency. This is applicable to all the dynamically deployable application components, but more particularly to the zone detector and the feedback generator. Second, the closer these application components are to each other, the better the performance. This is a valuable input to selecting the best deployment strategy, especially if all the layers are able to deploy all the application components. Finally, although cloud deployment can be cheaper in terms of costs, the latency obtained is far from the few milliseconds required by tactile internet applications. We should note that those measurements were only conducted for a one to one therapy session. In this sense, the architecture needs to be validated further for the one to many therapy session, in order for any definitive conclusions to be drawn.

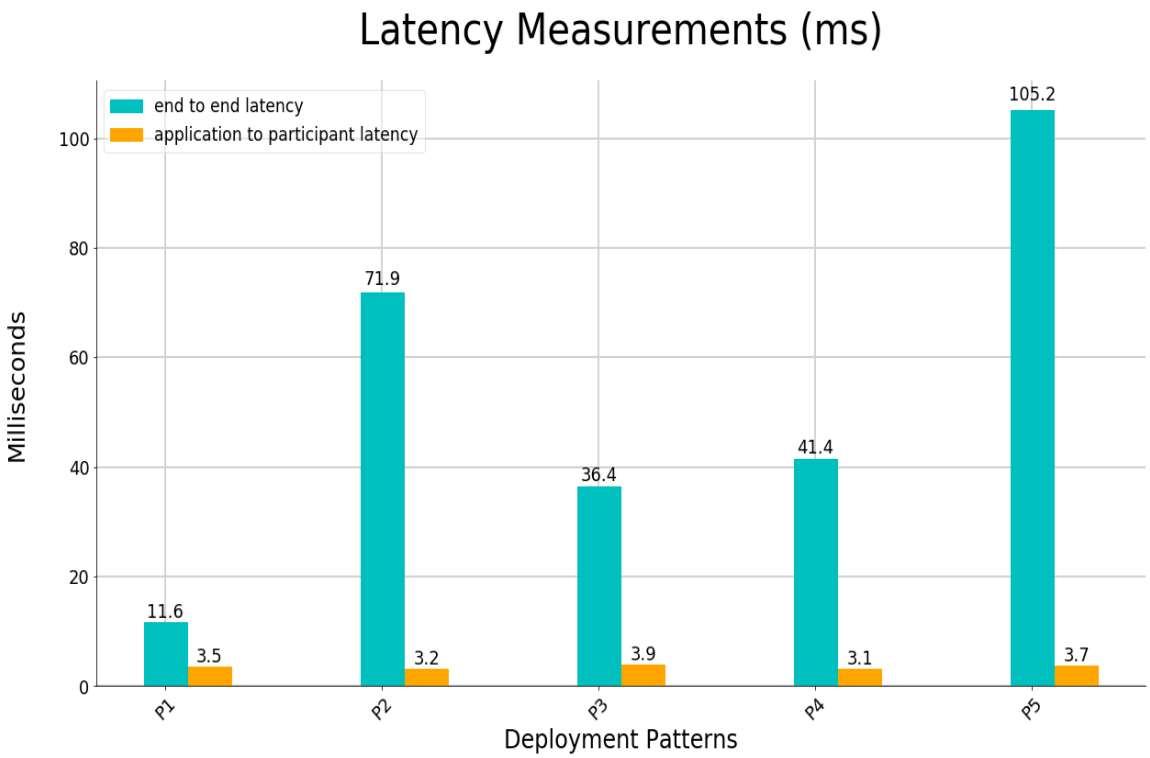


Figure 5.10: Performance Evaluation for both Latencies' Variation in terms of Application Components Deployment Patterns among the 3 Layers (criterion 3).

5.4 Conclusion

In this chapter, we presented the proof of concept prototype and the technologies used to implement it. Afterwards, we proceeded in the performance measurement section to describe the performance metrics as well as the considered performance use cases. Then, we presented the experiment setup, and the obtained results were shown and analysed. Finally, the gained insights from the obtained results were discussed.

In the next chapter, the thesis is concluded with a specific focus on its summary and the future work to be done.

Chapter 6

Conclusion

In this chapter, we begin with a highlight of the contributions of this thesis. Afterwards, we focus on the possible future research directions.

6.1 Contributions Summary

Over the last few decades, a lot of progress has been made in the treatment of phobias. Nevertheless, remote phobia treatment remains far from being an easy challenge. In addition to the complications that may arise from the phobia treatment procedure itself, new challenges emerge in the case where phobia therapy sessions are conducted remotely.

Although Tactile Internet and its critical requirements, namely in terms of the networking end to end latency, can help in that regard, achieving the few milliseconds latency is not a trivial task. Moreover, conducting therapy sessions remotely puts even more emphasis on the need to be able to communicate in a multimodal manner, which should allow the therapist to guide his patient using visual, auditory as well as tactile information during the therapy session. Last but not least, for such therapy to take place over the internet, a set of tactile devices, including VR equipment and haptic gloves, should be incorporated. Therefore, the heterogeneous nature of such devices and leveraging them for the purpose of remote phobia treatment is another challenge that needs to be addressed.

To obtain a good grasp of those challenges and evaluate whether other challenges might need to be taken into consideration, the remote phobia treatment use case was comprehensively explained and presented in a step by step fashion. In addition to the three aforementioned requirements, which include an ultra low latency to avoid "cyber-sickness", multimodal information exchange, and uniform interfaces for the heterogeneous devices, another requirement was determined, which is related to allowing the therapist to conduct individual "one to one" therapy sessions as well as collective "one to many" therapy sessions. Indeed, some patients' phobia may be more severe than their counterparts'. Therefore, those severe phobia patients may require special care from the therapist in an individual therapy session.

After the identification of those requirements, we proceeded to evaluate the existing work in the literature, combining between what was achieved in the field of psychology in that regard, with existing VR-based solutions and tactile internet applications, in

addition to the work related to the mobile ad-hoc clouds. We came, eventually, to the conclusion that none of the solutions in those categories satisfies all the requirements that we determined.Afterwards, we moved to presenting our proposed architecture for remote phobia treatment as a tactile internet application, evaluating in the process whether it satisfies the predetermined requirements or not. After introducing the various architectural components in the architecture, we presented the interfaces that should be exposed to allow uniform inter-components communication as well as components-devices communication, before finishing this chapter with a set of illustrative scenarios.

In terms of requirements, the proposed architecture incorporates a set of functional entities and 3 layers, including cloud, edge and mobile ad-hoc cloud. To this end, the functional entities collaborate, before the start of a therapy session, to generate a deployment plan that results in obtaining an ultra low latency. Ideally, such latency is obtained with a full mobile ad-hoc cloud application components' deployment. However, in some cases, such deployment cannot be achieved either due to insufficient resources or because the deployment costs can prove to be too expensive. In this case, a hybrid deployment between cloud and/or edge and/or mobile ad-hoc cloud is opted for instead. Moreover, the proposed architecture allows exchange of visual, auditory and tactile information, as a set of tactile devices, including a VR headset and a haptic glove and a hand tracking device are leveraged in the architecture. Furthermore, as the functional entities are responsible for deployment and instantiating the application components, it is possible for a therapist to conduct an individual as well as a collective therapy session. In addition, following a REST-based approach, a set of uniform interfaces were designed to facilitate interactions between the various software and hardware entities within our architecture. In this sense, we came to the result that all of the requirements were satisfied. We finished this section by showcasing a set of illustrative scenarios, which demonstrate various interactions within the proposed architecture. Finally, although the IEEE 1918.1 standard for tactile internet architectures was not explicitly followed in our proposed architecture, the similarities between the standard and our architecture are sufficiently clear from a functional perspective.

To validate the proposed architecture, we started with an exhaustive presentation of the implemented prototype's architecture and the different technologies used for this implementation. We also used this section to present the tactile devices used, which included the HTC Vive VR headset, the Gloveone Haptic Glove and the Leap Motion tracking device for hand movement tracking. Afterwards, a brief description of the implemented prototype was given, ranging from the therapy web application to the conferencing server used for communication during a therapy session.

Finally, a couple of performance metrics were defined, which consisted of the end to end latency and the therapy application to participant latency. The obtained results were shown and analysed. To obtain these results, we start with a description of the performance use cases that we relied on for our performance evaluation, as well as the criteria chosen for evaluation. In this regard, using the 2 aforementioned metrics, we evaluated our prototype in regards to both latencies variation against the physical distance between the therapist and the patient. In addition, the prototype was evaluated in terms of end to end latency variation against different underlying infrastructure/machine capabilities. Last but not least, we tried a set of different placements for the application components to gain an insight about the

best potential placement of the application components across the 3 layers that results in an ultra low latency. To this end, we came to the conclusion that a full mobile ad-hoc cloud placement results in the lowest latency amongst all the placements tried. However, such deployment is only possible if the costs can be covered. Otherwise, a hybrid placement between cloud/edge/mobile ad-hoc cloud might be best depending on the desired trade-off between latency and costs. For this purpose, the deployment engine can benefit from a component placement algorithm, such as the one discussed in [70], which can only help further to find the optimal trade-off between latency and costs.

6.2 Future Research Directions

In this work, we relied mainly on the deployment aspect of the application components to achieve an ultra low latency. As previously mentioned, this might come at the expense of deployment costs. In future work, we intend to deploy our tactile internet application on top of a 5G infrastructure, and explore whether the end to end latency can be lowered further. In 5G, network softwarization techniques, such as SDN and NFV, can also help in reducing latency. In that regard, deployment of the application components as Virtual Network Functions (VNFs) might give promising results. In addition, network slicing can equally be explored to enhance this work, especially that it is already mentioned in the context of Tactile Internet, namely for the remote robotic surgery use case. Finally, compression techniques are often discussed in the tactile internet applications context. Those techniques include, for instance, bringing novel approaches in terms of haptic codecs to reduce kinesthetic data, or achieve compression of tactile signals. However, to the best of our knowledge, we are yet to witness a concrete contribution in this sense. To this end, this could be an equally intriguing direction where this work can be reoriented in a potential future extension.

References

[1] A. Aijaz, Z. Dawy, N. Pappas, M. Simsek, S. Oteafy, and O. Holland, "Toward a Tactile Internet Reference Architecture: Vision and Progress of the IEEE P1918.1 Standard," *arXiv:1807.11915 [cs]*, July 2018. arXiv: 1807.11915.

[2] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, and G. Fettweis, "5G-Enabled Tactile Internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, pp. 460–473, Mar. 2016.

[3] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges," *IEEE Communications Surveys Tutorials*, vol. 20, no. 1, pp. 416–464, 2018.

[4] X. Wei, Q. Duan, and L. Zhou, "A QoE-Driven Tactile Internet Architecture for Smart City," *IEEE Network*, vol. 34, pp. 130–136, Jan. 2020.

[5] V. Gokhale, K. Kroep, V. S. Rao, J. Verburg, and R. Yechangunja, "TIXT: An Extensible Testbed for Tactile Internet Communication," *IEEE Internet of Things Magazine*, vol. 3, pp. 32–37, Mar. 2020.

[6] T. Dbouk, A. Mourad, H. Otrok, H. Tout, and C. Talhi, "A Novel Ad-Hoc Mobile Edge Cloud Offering Security Services Through Intelligent Resource-Aware Offloading," *IEEE Transactions on Network and Service Management*, vol. 16, pp. 1665–1680, Dec. 2019.

[7] Y. Choy, A. J. Fyer, and J. D. Lipsitz, "Treatment of specific phobia in adults," *Clinical Psychology Review*, vol. 27, pp. 266–286, Apr. 2007.

[8] C. Botella, J. Fernández-Álvarez, V. Guillén, A. García-Palacios, and R. Baños, "Recent Progress in Virtual Reality Exposure Therapy for Phobias: A Systematic Review," *Current Psychiatry Reports*, vol. 19, p. 42, July 2017.

[9] Y. Jebbar, F. Belqasmi, R. Glitho, and O. Alfandi, "A Fog-Based Architecture for Remote Phobia Treatment," in *2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 271–278, Dec. 2019. ISSN: 2330-2186.

[10] "ITU-T Technology Watch Report (August 2014) - The Tactile Internet."

[11] G. P. Fettweis, "The Tactile Internet: Applications and Challenges," *IEEE Vehicular Technology Magazine*, vol. 9, pp. 64–70, Mar. 2014.

[12] "Falcon Haptic Device > Anarkik3D." Library Catalog: anarkik3d.co.uk.

[13] "Touch," June 2016. Library Catalog: www.3dsystems.com.

[14] "Avatar VR - The best haptic glove for training in VR." www.avatarvr.es.

[15] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Communications of the ACM*, vol. 53, pp. 50–58, Apr. 2010.

[16] T. Dillon, C. Wu, and E. Chang, "Cloud Computing: Issues and Challenges," in *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, (Perth, Australia), pp. 27–33, IEEE, 2010.

[17] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, pp. 637–646, Oct. 2016.

[18] D. Zhang, D. Zhang, H. Xiong, C.-H. Hsu, and A. V. Vasilakos, "BASA: building mobile Ad-Hoc social networks on top of android," *IEEE Network*, vol. 28, pp. 4–9, Jan. 2014.

[19] B. Zaghdoudi, H. K.-B. Ayed, and I. Riabi, "Ad Hoc Cloud as a Service: A Protocol for Setting up an Ad hoc Cloud over MANETs," *Procedia Computer Science*, vol. 56, pp. 573–579, Jan. 2015.

[20] I. Yaqoob, E. Ahmed, A. Gani, S. Mokhtar, M. Imran, and S. Guizani, "Mobile ad hoc cloud: A survey," *Wireless Communications and Mobile Computing*, vol. 16, no. 16, pp. 2572–2589, 2016.

[21] R. C. Kessler, P. Berglund, O. Demler, R. Jin, K. R. Merikangas, and E. E. Walters, "Lifetime Prevalence and Age-of-Onset Distributions of DSM-IV Disorders in the National Comorbidity Survey Replication," *Archives of General Psychiatry*, vol. 62, pp. 593–602, June 2005.

[22] I. M. Marks, "Imipramine and Brief Therapist-Aided Exposure in Agoraphobics Having Self-exposure Homework," *Archives of General Psychiatry*, vol. 40, p. 153, Feb. 1983.

[23] V. E. Caballo, *International Handbook of Cognitive and Behavioural Treatments for Psychological Disorders*. Elsevier, Nov. 1998.

[24] K. Meyerbröker and P. M. G. Emmelkamp, "Virtual reality exposure therapy in anxiety disorders: a systematic review of process-and-outcome studies," *Depression and Anxiety*, vol. 27, pp. 933–944, Oct. 2010.

[25] L.-G. Öst and K. Hugdahl, "Acquisition of phobias and anxiety response patterns in clinical patients," *Behaviour Research and Therapy*, vol. 19, pp. 439–447, Jan. 1981.

[26] A. Jerremalm, L. Jansson, and L. G. Ost, "Individual response patterns and the effects of different behavioral methods in the treatment of dental phobia," *Behaviour Research and Therapy*, vol. 24, no. 5, pp. 587–596, 1986.

[27] L.-G. Öst, "Fading vs systematic desensitization in the treatment of snake and spider phobia," *Behaviour Research and Therapy*, vol. 16, pp. 379–389, Jan. 1978.

[28] A. Aijaz, M. Dohler, A. H. Aghvami, V. Friderikos, and M. Frodigh, "Realizing the Tactile Internet: Haptic Communications over Next Generation 5G Cellular Networks," *IEEE Wireless Communications*, vol. 24, pp. 82–89, Apr. 2017. arXiv: 1510.02826.

[29] Z. Shi, H. Zou, R. M, L. Chen, H. S, and M. Hj, "Effects of Packet Loss and Latency on the Temporal Discrimination of Visual-Haptic Events.," *IEEE Transactions on Haptics*, vol. 3, pp. 28–36, Oct. 2009.

[30] D. L. M, M. Tk, and E. Mo, "Recalibration of Multisensory Simultaneity: Cross-Modal Transfer Coincides With a Change in Perceptual Latency," Nov. 2009. ISSN: 1534-7362 Issue: 12 Volume: 9.

[31] K. M. Stanney, D. A. Graeber, R. S. Kennedy, D. A. Graeber, and R. S. Kennedy, "Virtual Environment Usage Protocols," Dec. 2005. Pages: 401-418.

[32] R. Chaudhari, C. Schuwerk, M. Danaei, and E. Steinbach, "Perceptual and Bitrate-Scalable Coding of Haptic Surface Texture Signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, pp. 462–473, Apr. 2015.

[33] M. Dohler, T. Mahmoodi, M. A. Lema, M. Condoluci, F. Sardis, K. Antonakoglou, and H. Aghvami, "Internet of skills, where robotics meets AI, 5G and the Tactile Internet," in *2017 European Conference on Networks and Communications (EuCNC)*, (Oulu, Finland), pp. 1–5, IEEE, June 2017.

[34] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Communications Magazine*, vol. 54, pp. 146–153, July 2016.

[35] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing management & prioritization in 5G mobile systems," 2016.

[36] N. Bizanis and F. A. Kuipers, "SDN and Virtualization Solutions for the Internet of Things: A Survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

[37] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," p. 7.

[38] "A break in the clouds: towards a cloud definition: ACM SIGCOMM Computer Communication Review: Vol 39, No 1."

[39] I. Sriram and A. Khajeh-Hosseini, "Research Agenda in Cloud Technologies," *ArXiv*, 2010.

[40] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, pp. 7–18, May 2010.

[41] R. Buyya, J. Broberg, and A. Goscinski, "Cloud Computing Principles and Paradigms," p. 674.

[42] N. R. Herbst, S. Kounev, and R. Reussner, "Elasticity in Cloud Computing: What It Is, and What It Is Not," pp. 23–27, 2013.

[43] F. Liu, G. Tang, Y. Li, Z. Cai, X. Zhang, and T. Zhou, "A Survey on Edge Computing Systems and Tools," *Proceedings of the IEEE*, vol. 107, pp. 1537–1562, Aug. 2019.

[44] M. Satyanarayanan, "Pervasive computing: vision and challenges," *IEEE Personal Communications*, vol. 8, pp. 10–17, Aug. 2001.

[45] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H.-I. Yang, "The case for cyber foraging," in *Proceedings of the 10th workshop on ACM SIGOPS European workshop*, EW 10, (Saint-Emilion, France), pp. 87–92, Association for Computing Machinery, July 2002.

[46] M. Yannuzzi, R. Milito, R. Serral-Gracià, D. Montero, and M. Nemirovsky, "Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing," in *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pp. 325–329, Dec. 2014. ISSN: 2378-4873.

[47] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog Computing: Platform and Applications," in *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*, pp. 73–78, Nov. 2015.

[48] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services - MobiSys '14*, (Bretton Woods, New Hampshire, USA), pp. 68–81, ACM Press, 2014.

[49] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: elastic execution between mobile device and cloud," in *Proceedings of the sixth conference on Computer systems*, EuroSys '11, (Salzburg, Austria), pp. 301–314, Association for Computing Machinery, Apr. 2011.

[50] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative Mobile Edge Computing in 5G Networks: New Paradigms, Scenarios, and Challenges," *IEEE Communications Magazine*, vol. 55, pp. 54–61, Apr. 2017.

[51] T. Dbouk, A. Mourad, H. Otrok, and C. Talhi, "Towards ad-hoc cloud based approach for mobile intrusion detection," in *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, (New York, NY), pp. 1–8, IEEE, Oct. 2016.

[52] C. Suso-Ribera, J. Fernández-Álvarez, A. García-Palacios, H. G. Hoffman, J. Bretón-López, R. M. Baños, S. Quero, and C. Botella, "Virtual Reality, Augmented Reality, and In Vivo Exposure Therapy: A Preliminary Comparison of Treatment Efficacy in Small Animal Phobia," *Cyberpsychology, Behavior, and Social Networking*, vol. 22, pp. 31–38, Oct. 2018.

[53] J. Almeida, D. Suárez, F. Tapia, and G. Guerrero, "Use of Virtual Reality Using Render Semi-realistic as an Alternative Medium for the Treatment of Phobias. Case Study: Arachnophobia," in *Applied Informatics* (H. Florez, C. Diaz, and J. Chavarriaga, eds.), Communications in Computer and Information Science, (Cham), pp. 144–154, Springer International Publishing, 2018.

[54] J. Kritikos, S. Poulopoulou, C. Zoitaki, M. Douloudi, and D. Koutsouris, "Full Body Immersive Virtual Reality System with Motion Recognition Camera Targeting the

Treatment of Spider Phobia," in *Pervasive Computing Paradigms for Mental Health* (P. Cipresso, S. Serino, and D. Villani, eds.), Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, (Cham), pp. 216–230, Springer International Publishing, 2019.

[55] C. Urrea and R. Matteoda, "Development of a virtual reality simulator for a strategy for coordinating cooperative manipulator robots using cloud computing," *Robotics and Autonomous Systems*, vol. 126, p. 103447, Apr. 2020.

[56] M. Bacco, P. Barsocchi, P. Cassará, D. Germanese, A. Gotta, G. R. Leone, D. Moroni, M. A. Pascali, and M. Tampucci, "Monitoring Ancient Buildings: Real Deployment of an IoT System Enhanced by UAVs and Virtual Reality," *IEEE Access*, vol. 8, pp. 50131–50148, 2020.

[57] P. Abichandani, W. Mcintyre, W. Fligor, and D. Lobo, "Solar Energy Education Through a Cloud-Based Desktop Virtual Reality System," *IEEE Access*, vol. 7, pp. 147081–147093, 2019.

[58] P.-V. Mekikis, K. Ramantas, A. Antonopoulos, E. Kartsakli, L. Sanabria-Russo, J. Serra, D. Pubill, and C. Verikoukis, "NFV-Enabled Experimental Platform for 5G Tactile Internet Support in Industrial Environments," *IEEE Transactions on Industrial Informatics*, vol. 16, pp. 1895–1903, Mar. 2020.

[59] B. Zaghdoudi, H. K.-B. Ayed, and I. Gnichi, "A protocol for setting up ad hoc mobile clouds over spontaneous MANETs: A proof of concept," in *2016 Cloudification of the Internet of Things (CIoT)*, (Paris, France), pp. 1–6, IEEE, Nov. 2016.

[60] C. Li, Z. Liye, T. Hengliang, and L. Youlong, "Mobile user behavior based topology formation and optimization in ad hoc mobile cloud," *Journal of Systems and Software*, vol. 148, pp. 132–147, Feb. 2019.

[61] T. Truong-Huu, C.-K. Tham, and D. Niyato, "A Stochastic Workload Distribution Approach for an Ad Hoc Mobile Cloud," in *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*, (Singapore, Singapore), pp. 174–181, IEEE, Dec. 2014.

[62] B. Li, Y. Pei, H. Wu, and B. Shen, "Heuristics to allocate high-performance cloudlets for computation offloading in mobile ad hoc clouds," *The Journal of Supercomputing*, vol. 71, pp. 3009–3036, Aug. 2015.

[63] S. C. Shah, "A Mobile Ad hoc Cloud Computing and Networking Infrastructure for Automated Video Surveillance System," *Journal of Computer Science*, vol. 13, pp. 767–780, Dec. 2017. arXiv: 1810.07338.

[64] F. Belqasmi, R. Glitho, and C. Fu, "RESTful web services for service provisioning in next-generation networks: a survey," *IEEE Communications Magazine*, vol. 49, pp. 66–73, Dec. 2011.

[65] "VIVE™ | Discover Virtual Reality Beyond Imagination." www.vive.com.

[66] "Digital worlds that feel human | Ultraleap." www.ultraleap.com.

[67] "Cloud Computing Services | Microsoft Azure." azure.microsoft.com.

[68] "Linux Deploy – Apps on Google Play." play.google.com.

[69] "Dropbox." www.dropbox.com.

[70] C. Mouradian, S. Kianpisheh, M. Abu-Lebdeh, F. Ebrahimnezhad, N. T. Jahromi, and R. H. Glitho, "Application Component Placement in NFV-Based Hybrid Cloud/Fog Systems With Mobile Fog Nodes," *IEEE Journal on Selected Areas in Communications*, vol. 37, pp. 1130–1143, May 2019.