Proxemic Interactions with Mobile Devices

Paulo César Pérez Daza

Proxemic Interactions with Mobile Devices

Paulo César Pérez Daza

Rapporteur:	Sophie Dupuy-Chessa	Université Grenoble Alpe, France
Rapporteur:	Xavier Le Pallec	Université de Lille, France
Directeur:	Philippe Rosse	Université de Pau et des Pays de l'Adour, France
Co-directrice:	Nadine Couture	ESTIA Institute of Technology, France
Co-directrice:	Yudith Cardinale	Université Simón Bolívar, Venezuela
Examinateur:	Dalmau Marc	Université de Pau et des Pays de l'Adour, France
Examinateur:	Emmanuel Dubois	Université de Toulouse, France
Invité:	Dominique Masson	Société DEV 1.0, France

Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour
(LIUPPA)
UNIVERSITÉ DE PAU ET DES PAYS DE L'ADOUR

This thesis is submitted for the degree of
Doctor of Computer Sciences

Anglet March 2021

Acknowledgements

First and foremost, my gratitude goes to the Lord my God: I thank Him for all His blessings. I give thanks to my loving parents, my mother on Earth, and my father in Heaven. My father always believed that education is the greatest wealth.

I would first like to offer my deep and sincere gratitude to Professor Philippe Roose for allowing me to follow my highest degree at Pau University and for finding the necessary resources for this research. He encouraged me and gave me his invaluable advice during the hardest times to complete the research successfully. His positive attitude has taught me to find solutions to the challenges in this thesis.

I would also like to thank my co-supervisors, Professor Nadine Couture and Professor Yudith Cardinale. Nadine Couture allowed me to integrate into the ESTIA Institute of Technology to share my research experiences with a multicultural environment. She did a careful review to improve the quality of work in the Human-Computer Interaction field and in the methodological aspect of this dissertation. I am extending my thanks to Professor Yudith Cardinale for providing invaluable guidance throughout the different publications.

In addition, I would like to thank Dr. Dalmau Marc for his valuable guidance throughout my studies. He provided me with the technical support necessary to choose the right direction and complete my dissertation. I would also like to thank Mr. Dominique Masson for his friendship and empathy when I arrived in France. I am extending my heartfelt thanks to the Informatics Department for acceptance during my work in the IUT Bayonne.

I give many thanks to my wife and son for their love, understanding, prayers, and continuing support to complete this research work. Also, I express my thanks to my sister, brother, sisters-in-law, brother-in-law, lovely mother-in-law, dear nephews, and niece for their support and valuable prayers.

Finally, my thanks go to all the people who have supported me in completing the research work directly or indirectly.

Abstract

Smart environments are currently overpowering traditional Human-Computer Interaction (HCI) approaches for people's everyday life applications. Proxemic interaction is an emerging area for improving HCI experiences in such environments, causing the so-called proxemic environments to emerge. Proxemic interactions establish how the five dimensions (i.e., Distance, Identity, Location, Movement, and Orientation – DILMO) can be used to implement interactions between people and digital devices. Current studies in this area are focused on developing proxemic applications with a specific task and toolkits that allow developers to obtain DILMO information from a wide range of sensors. However, there exists a notable lack of general approaches capable of supporting the whole implementation process from the modelling of proxemic environments to represent general proxemics behaviours and finalizing with the development of mobile applications. To help the integration of proxemic capabilities in HCI, we propose an approach for modelling proxemic environments based on a graphical Domain-Specific Language (DSL). The DSL allows designers to express proxemic interactions for modelling proxemic environments and supports the development process.

We also provide an API in a framework for mobile devices based on Android operating system. This API implements high-level primitives (DILMO) permitting to provide proxemic interactions on widespread mobile devices. We have developed an API that is feasible for developing proxemic mobile applications.

We applied the proxemic interactions in order to develop an architecture that can support mobile applications in the health sector. We propose interpersonal distances and proxemic dimensions (i.e., Distance, Identity, and Orientation - DIMO) to implement HCI with mobile devices that encourage touchless interactions. Our goal was to promote mobile apps' development with proxemic HCI, supported in a proposed architecture, to stop the spreading of nosocomial infection. To illustrate our proposal's usability, we developed two prototypes of applications for mobile devices as a proof-of-concept, using several combinations of proxemic DILMO dimensions to model proxemics HCI that allowed flexible interaction between people and mobile devices.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Nowadays, we are surrounded by smart device technologies in different aspects of our daily life. Recent advances on mobile technology boost the development of a wide variety of applications to support users in their activities (e.g., sports, health care, on-line services, business) or to help them in unexpected or dangerous situations (e.g., first aid, natural disasters) [11, 24]. The use of mobile technologies in our ordinary life is increasing in a way without precedent. People can interact with different contexts through electronic devices (e.g., mobile phones, tablets, wearable technologies, and smartwatches) to accomplish their common tasks. Many of these tasks require a specific Human-Computer Interaction (HCI). In the last decade, researchers have demonstrated that the concept of proxemics can support HCI in several uses such as remote controls and interaction for smart environments. Proxemic interaction describes how people use interpersonal distances to interact with digital devices applying the five physical proxemic dimensions [9, 41]: Distance, Identity, Location, Movement, and Orientation (DILMO).

Proxemic interaction is derived from the social theory of proxemics proposed in 1963 by the anthropologist Edward T. Hall [46]. Hall describes how individuals perceive their personal space relative to the distance among themselves and others. Social relationships are essential in the life of human beings and can be expressed as how people allow contact and interaction among each other in a physical space. Thus, people interactions are based on physical distances or face orientation of others. Both factors describe the level of engagement among people to establish communication. Similarly, there are relationships among people and digital objects in a cyber environment, in which the physical distance is the main factor that determines the interactions as response of such digital objects.

Edward T. Hall established the proxemics as a concept used mainly to describe the human use of space. He pointed out proxemics as "the interrelated observations and theories of how the humans use of space as a specialized elaboration of culture" [46]. His research illustrates

the proxemic behavior of people using a system notation that is based on communication factors among people such as: eye contact, body language (i.e., posture), tone of voice, gesture, and facial expression.

This social science theory has inspired researchers to create seamless interactions between users and digital surfaces and devices in which their physical affect the interactive behaviour. Additionally, current studies in this area focus on developing proxemic applications for specific functionalities [9, 14, 43, 59], such as controlling digital devices in smart environments for implementing interactive proxemic applications (e.g., media players, video analysis, and remote control applications). Some other works propose frameworks and toolkits in order to help developers to build systems based on proxemic interactions [16, 64]. Also, proxemic applications have been introduced for improving the daily life of blind people [12, 70, 76].

Nowadays, the vast majority of smartphones and mobile devices are equipped with powerful hardware capabilities. These capabilities allow devices to process and obtain proxemic information from the measurement of the DILMO parameters; for example, by using sensors and cameras embedded in a smartphone, it is possible to estimate physical distances between objects or people by means of the smart-mobile camera. In turn, we can implement proxemic-based mobile applications aiming to facilitate users' contact and interaction with other people and devices within proxemic environments, composed of indoor and outdoor spaces.

This current trend of using proxemic interaction with the aim to improve mobile HCI, has raised the need for frameworks and tools to assist the development of such proxemic applications based on mobile devices and wearable technologies. However, there exists a lack of more general approaches able to support the whole implementation process (starting from the design phase) of proxemic applications for smart environments. In such a way, end-users, without programming knowledge, can easily design such proxemic applications, taking into account how to make better use of social distance based on theory of proxemics and implement their designs in mobile devices. Thus, the need for standardization and a more powerful, flexible, and general approach emerges. We use the term "Flexible interaction" as used by Groenbak et al in [42], for designing flexible cross-device systems based on proxemic interactions.

Accordingly, to overcome those limitations, we propose a framework to support the development of proxemic applications and the creation of proxemic environments with mobile devices and wearable technologies. More precisely, the main goals of this research are:

- To demonstrate that proxemics can improve people's daily lives using mobile devices based on proxemic interactions.

- To propose the modelling of proxemic environments based on graphical notation with the purpose of designing and implementing proxemic applications. Thus, non-expert users can easily design proxemic environments and represent proxemic behaviors, which allow developers to implement the proposed design.
- To create a framework-like toolkit, along with a graphical Domain Specific Language (DSL) and an API in order to allow developers to build proxemic mobile applications based on a combination of the DILMO dimensions, which determine interactions either between a person and a device or among devices.

1.1 Contributions

Responding the proposed objectives, the main contributions of this study can be summarized as follows:

- A mobile application based on proxemic interactions to help rescuers identify an unconscious person using social interaction, in which the unconscious person's device can share information without explicit interaction (see Chapter 4).
- A graphical DSL, with graphic symbols and formal notation that allows non-specialist modeling of proxemic interactions and their integration into proxemic applications for a better HCI. We develop a prototype of the DSL, which simulates proxemic environments, including entities, proxemic zones, DILMO combinations, and conditions to define entity's behaviors and reactions (see Chapter 5).
- An API toolkit aiming to help developers to build mobile applications that can be implemented in mobile devices by means of exclusively employing sensors available in such devices, in order to obtain proxemic information (see Chapter 6).
- An integral framework, with the DSL and the API, that provides an approach for defining proxemic zones, as well as managing DILMO proxemic dimensions and technical components required for creating proxemic environments based on the use of both mobile appliances (see Chapter 6).
- Mobile applications developed with the proposed framework. We offer mobile applications that allow people to use social distances without compromising users' intimate space to control a video player, to control a musical notes player, and to reduce the spread of nosocomial infections in healthcare workers (HCWs), according to several combinations of DILMO dimensions (see Chapter 7).

1.2 Publications

From the results of the contributions of this research, the following publications were produced and support the material presented in this thesis:

- Pérez, P., Roose, P., Cardinale, Y., Dalmau, M., Masson, D., Couture, N. (2020) *Mobile Proxemic Application Development for Smart Environments*. In Proceedings of the 18th ACM International Conference on Advances in Mobile Computing & Multimedia(MoMM 2020), Chiang Mai, Thailand, November 3- December 2, pp. 94 -103.

- Pérez, P., Roose, P., Cardinale, Y., Dalmau, M., Masson, D., Couture, N. (2020) *Proxemic Environments Modelling Based on a Graphical Domain-Specific Language*. In Proceedings of the 17th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2020), Antalya, Turkey, November 2-5, pp. 1-8.

- Pérez, P., Roose, P., Cardinale, Y., Dalmau, M., Masson. (2020) *Proxemic Interactions in Mobile Devices to Avoid the Spreading of Infections*. In Proceedings of the 16th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2020), Thessaloniki, Greece, October 12-14. pp. 48-55.

- Pérez, P., Roose, P., Cardinale, Y., Dalmau, M., Masson, D, Couture, N. (2020) *Proxemic Environments: A Framework for Developing Mobile Applications based on Proxemic Interaction*. In Proceedings of the 15th IEEE Conference on Computer Science and Information Systems (FedCSIS 2020), Sofia, Bulgaria, September 6-9, pp. 653- 656.

- Pérez, P., Roose, P., Dalmau, M., Cardinale, Y., Masson, D, Couture, N. (2020) *Modélisation graphique des environnements proxémiques basée sur un DSL*. In Proceedings of the INFormatique des ORganisations et Systèmes d'Information et de Décision (INFORSID 2020), Dijon, France, Jun 2-4, pp. 99-114.

- Pérez, P., Roose, P., Dalmau, M.,Couture, N. Cardinale, Y., Masson, D. (2018) *Proxemics for first aid to unconscious injured person*. In Proceedings of the 30th ACM Conference on l'Interaction Homme-Machine (L'IHM 2018) Brest, France, October 23-26, pp, 156-162.

1.3 Structure of the Dissertation

The structure of this dissertation is given as follows:

- In Chapter 2, we describe the theory of proxemics and how this social theory has been adopted in ubiquitous computing (ubicomp) environments for improving the HCI. We describes fundamental concepts of HCI and proxemic interaction.
- In Chapter 3, we outline related work based on proxemic interaction. We illustrate the process of developing proxemic systems, including tools for design and frameworks architecture that allows developers to built proxemic applications. Besides, we make reference to technologies for implementing proxemic systems.
- In Chapter 4, we explore the proxemic concept to manage effectively social distance implementing a first aid mobile application that identifies unconscious people, generates alerts, and shares medical identification in a secure way. In this context, we develop a First Aid Mobile Application (FAMA) that can emit emergency information to other smartphones based on proxemic interactions.
- In Chapter 5, we describe our proposed approach for modelling proxemic environments based on a DSL, as well as the formal notation that assist designers to represent proxemic behaviors of entities that are part of the proxemic environment. Moreover, we show the suitability and functionality of our proposed graphical DSL by means of the implementation and the description of a prototype exemplifying the design of proxemic behaviors in different scenarios.
- In Chapter 6, we proposed a framework to implement mobile applications, which allows adapting combinations of DILMO proxemic dimensions to the required scenario. We illustrate our framework that allows developers to build proxemic mobile applications supporting the process of proxemic information.
- In Chapter 7, we describe the use of interpersonal distances and DILMO proxemic dimensions for implementing HCI with mobile devices that reduces their touchability. We propose an architecture, to stop spreading of nosocomial infection. To show the usability and suitability of our proposal, we present two prototypes of apps for mobile devices as proof-of-concept.
- Chapter 8 summarizes and concludes this thesis work, by stating the key contributions. Finally, ideas and implications concerning future work and its directions are considered.

Chapter 2

Proxemic Interactions in Human-computer interaction: preliminaries

To study the proxemic interaction in the context of computer applications is important to know and understand the social theory of proxemics and how it has been adopted to improve HCI. This chapter is dedicated to explain these aspects.

2.1 Theory of Proxemics

Social distances are present in our daily existence. Theory of proxemics is a concept used mainly to describe the human use of space. Edward T. Hall proposed the first definition, who pointed out proxemics as "the interrelated observations and theories of humans use of space as a specialized elaboration of culture" [46].

He presented how people perceive, interpret, and use space, specially related to distance among people [35]. Theory of proxemics describes how people from different cultures speak not only diverse languages but also inhabit different sensory worlds. For example, Hall's theory demonstrated that human cultures use differences physical distance of communication. In some cultures, people make more use of olfaction and touch senses than others. In this respect, the distance has an indispensable role in proxemics in order to establish a region around the person that serves to maintain proper spacing among individuals.

According to theory of proxemics, the interaction zones have been classified as four proxemic zones, as shown in Figure 2.1. The four distance zones have been collected from

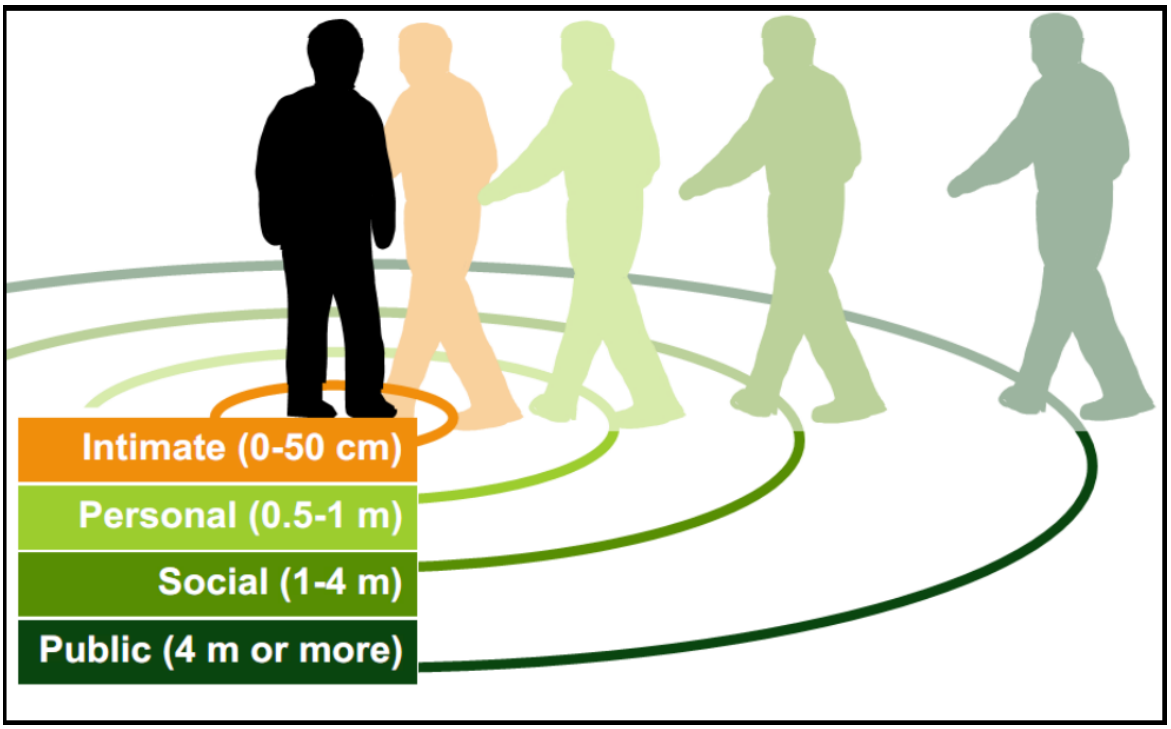


Fig. 2.1 Edward Hall's proxemic theory: interpersonal distances of people, showing radius in meters [46]

observations and studies of the human behaviour in different environments of the world. Below, we describe the properties of each proxemic zone:

- **Intimate (0-50 cm)**: This space is reserved for close relationships between people. The physical contact of both persons is possible in this zone. Usually, people can access to the zone if the other person allows it (Figure 2.2(a)). However, there are exceptions according to environments, such as public transportation, lifts where a person's intimate zone can be compromised without his permission.

- **Personal (0.5 cm-1 m)**: In this zone, people could have a natural interaction with other people, and it is barely possible to reach contact with their arms in which physical domination is limited (Figure 2.2(b)). Beyond it, a person can not freely "get his hands on" someone else.

- **Social space (1-4 m)**: This area can be related to space where the people could maintain communication without touching each other (e.g., a meeting table). In this zone people keep physical distancing among individuals. For example, a business meeting can describe the zone in which people have to speak louder to address others in order to catch their attention (Figure 2.2(c)).

- **Public (>4 m)**: It describes the distribution of people in urban spaces such as concert hall, public meeting, cinema, where the people's attention is focused on the moderator (Figure 2.2(d)). The person's identity is unknown among individuals that share the same space.

Fig. 2.2 Proxemic zones: (a) intimate[1], (b) personal[2], (c) social[3], and (d) public[4].

Theory of proxemics also describes how people use the space and their territories for specific activities such as sleeping, eating, and working. The territory is in all respects of the word, an extension of the organism, which has been characterized by visual, vocal, and olfactory signs [45]. Hall also describes that any culture's characteristic produces a set of patterns such as postural, touch code, and voice loudness relative to proxemic behavior in the environment [45]. He created a notation system to describe such behaviours. Proxemic behaviour details if a person is standing, sitting, squatting, or lying down in the physical space.

Hall classified the territory as a fixed-feature space and a semifixed-feature space, which we describe below.

Fixed-Feature Space

Edward Hall describes how people design their urban area in the physical space according to a cultural context; for example, houses, office buildings. He emphasizes the lack of congruence in the design of civil infrastructure, which must be designed based on users' physical characteristics such as height and body volume. He highlights that characteristics arrangement of the rooms is different among cultures. According to this theory, people's behavior is influenced by fixed-feature space characteristics.

[1]Royalty-free stock photo ID: 506837092
[2]Royalty-free stock photo ID: 479653195
[3]Royalty-free stock photo ID: 66163672
[4]Royalty-free stock photo ID: 1517239640

Semifixed-Feature Space

Semi fixed-feature space indicates the arrangement of objects in the physical space like home, conference room, office. Generally, these types of areas may include furniture, plants, screens, paintings, decorations. Edward Hall details how the arrangement of objects can influence on human relationships. For example, "the tables at a French sidewalk cafe, tend to bring people together". On Figure 2.3, we illustrate two examples of seating arrangements in the space which can affect interaction and communication among people.

Fig. 2.3 Seating arrangements affecting people interaction: (a) Chairs arranged in a circle encourage communication[1]; (b) Chairs arranged in a row limit interactions among people[2];

Next, we explain how theory of proxemics has been implemented in ubiquitous computing for improving HCI.

2.2 Human-Computer Interactions

Nowadays, we are surrounded by technology. The HCI is present in diverse aspects of our daily life using digital devices, such as mobile phones, tablets, game consoles, vending machines, laptops, among many others, that require different interaction modalities. The interaction modalities have been described as the communication channels between humans and computers (i.e., a set of transformations of raw data from input devices). It is generated by input/output peripherals [58]. HCI is mostly supported by computer vision, which is the field of computer science that provides new ways of interaction and new applications [2]. In the next section, we describe technologies that assist computing environments based on HCI.

[1]Royalty-free stock photo ID: 1023809758
[2]Royalty-free stock photo ID: 1302594496

2.2.1 Movement Recognition

Computer vision in the 21st century is widely used in HCI for movement recognition, allowing users to interact with an invisible computer in the environment.[2]. Vision-based HCI allows obtaining a more extensive range of input capabilities by using computer vision techniques in order to process data from cameras. Visual-based HCI is associated with movement recognition technology, which is produced by sophisticated sensors (e.g., camera with embedded vision processor, depth camera with Microsoft Kinect sensor) [50, 95]. Motion capture is the method of recording actors' movement through skeletal tracking in which the human body is represented by several joints representing body parts such as head, neck, shoulders, and arms [48, 53, 95]. Computer vision is implemented to support face detection, proximity between the camera and object [48, 50, 53]. Microsoft Kinect sensor is an input device that allows users to interact with electronic devices based on hand gesture recognition (touchless interaction) [37]. The current Kinect sensor technology provides flexible, naturalistic interfaces that support HCI. Although conventional input devices such as the keyboard, mouse, and touch screen already involve bodily actions, new interaction technologies utilizing human movements that provide more flexible, naturalistic interfaces and support the ubiquitous computing paradigm.

In the next section, we describe the computing ubiquitous computing that has been incorporated into our environment to help us in our daily lives.

2.2.2 Ubiquitous Computing

Mark Weiser coined the first definition of ubiquitous computing (ubicomp) in 1988 when he was a principal scientist of the Computer Science Laboratory at Xerox PARC [93]. Ubiquitous computing assumes the disappearance of technologies into the everyday smart environment, creating them invisible to a user [63]. Ubiquitous computing has included different services and smart technologies (Internet, operating system, sensors, microprocessors, interfaces, networks, and mobile protocols), which allow people interacting with the environment in a more natural and more personalized way. His approach was focused on the construction of new computing artifacts for use in everyday life. He took his inspiration from objects from home and office, particularly objects that transmit or capture information. He classified the devices that were part of ubiquitous computing environment according to three sizes:

- Wall-size interactive surface similar to the whiteboard office which is a big size and characterizes semifixed-feature spaces.

- Office desk and desk accessories, such as note pad, desk lamp, pends, etc., that use an individual person on his daily work.
- Portable devices like personal digital assistants in a way that each person can have a least one device in one office.

Ubiquitous computing environment has explored a group of devices intending to evaluate the autonomy of computing artefacts for use in everyday life. Figure 2.4 illustrates the first ubiquitous computing environment composed by Liveboard lard-sized and personal digital devices, such as note-book and digital assistant ParcTab, that were proposed by Xerox Palo Alto Research Center Ubicomp shows how the new technologies could be more advantageous

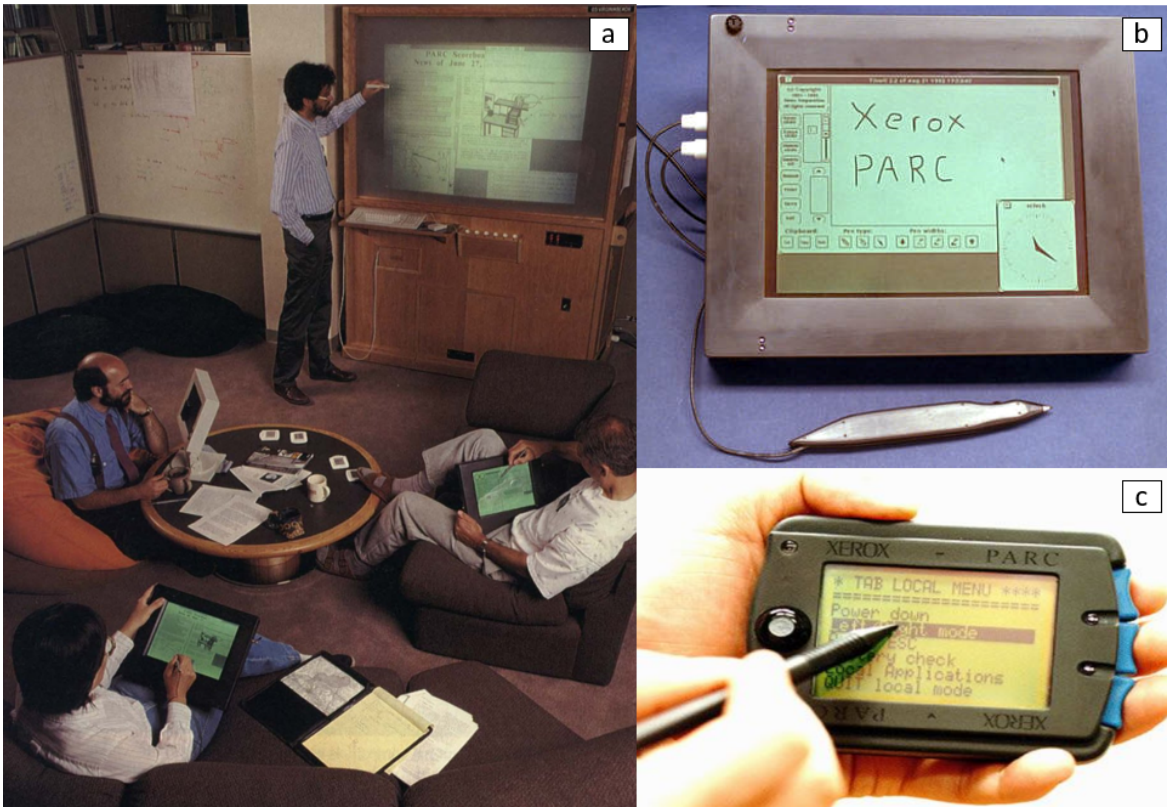


Fig. 2.4 Ubiquitous Computing environment Xerox Parc: (a) The standing man in the room uses Liveboard *Yard-sized* [93]; (b) It is a note-book device ParPad [93]; (c) Personal digital assistant ParcTab *Inch-sized* [93].

to the user when users interact with multiple wireless devices interconnected.

Ubiquitous computing provides new ways to explore HCI in which one user has many computing devices and can be employed according to the context required by the user. At this point, proxemic interaction emerges as a strategy to mediate people's interaction in ubicomp environments and facilitate communication between the user and digital devices.

Once the concept of ubiquitous computing is clear, in the following section, we describe proxemic interaction and how it has been implemented in HCI for ubicomp environments.

2.3 Proxemic Interactions

Sensing technologies are continually evolving by advancements in technology allowing new techniques for HCI. The earliest interaction between a human and a machine was purely physical before the introduction of computing systems [73]. Over the past two decades, HCI emerged as an area of computer science that studies the user interfaces between humans and machines [17]. HCI evoked many challenging problems, such as direct manipulation interfaces, user interface management systems, and collaborative work. It is a discipline that has been characterized as the visible part that impacts human lives for supporting computer use. The basic interactions were based on direct manipulation, where visible objects on the screen are directly manipulated with a peripheral device [72]. Peripheral interaction describes how people can perceive information from and physically interact with computing technology [8]. Context-Aware interaction describes a system state as implicit interaction according to inputs of itself [79].

In this regard, implicit HCI captures environmental information that allows implementing context-aware of digital devices [79]. Since the last decade, proxemics in HCI has developed the social theory based on physical distance between people and digital devices. Proxemic interaction has been implemented to enhance HCI in ubiquitous environments [59], where the physical distances and contextual factors have been used as a user's interface among the users and ubicomp devices [77]. Proxemic interaction is focused on improving social interaction through intuitive implicit communication and fully integrated into everyday activities [59]. HCI methods need to be designed to adapt to the interaction of different contexts in which combine two or more modes of input. Therefore, proxemic interaction raises awareness of the environment to indicate situations and establish communication channels between users and devices, allowing interaction without physical contact. In this regard, proxemic interaction is becoming an influential approach to implement HCI in ubiquitous computing [13, 64], where the proximity has been used as a user's interface.

Theory of proxemics describes how individuals perceive their personal space relative to the distance between themselves and others, which are based on physical, social, and cultural contextual factors that influence and regulate interpersonal interactions [64, 94]. In order to know how the factors should be applied designing proxemic-aware ubicomp systems Greenberg identified five dimensions [41]: Distance, Identity, Location, Movement, and Orientation (we call them DILMO as an abbreviation), which are associated with people,

digital devices, and non-digital things. Based on these concepts, Ballendat illustrates a ubiquitous computing ecology based on proxemic interactions that describe how people interact with digital devices (see Figure 2.5).

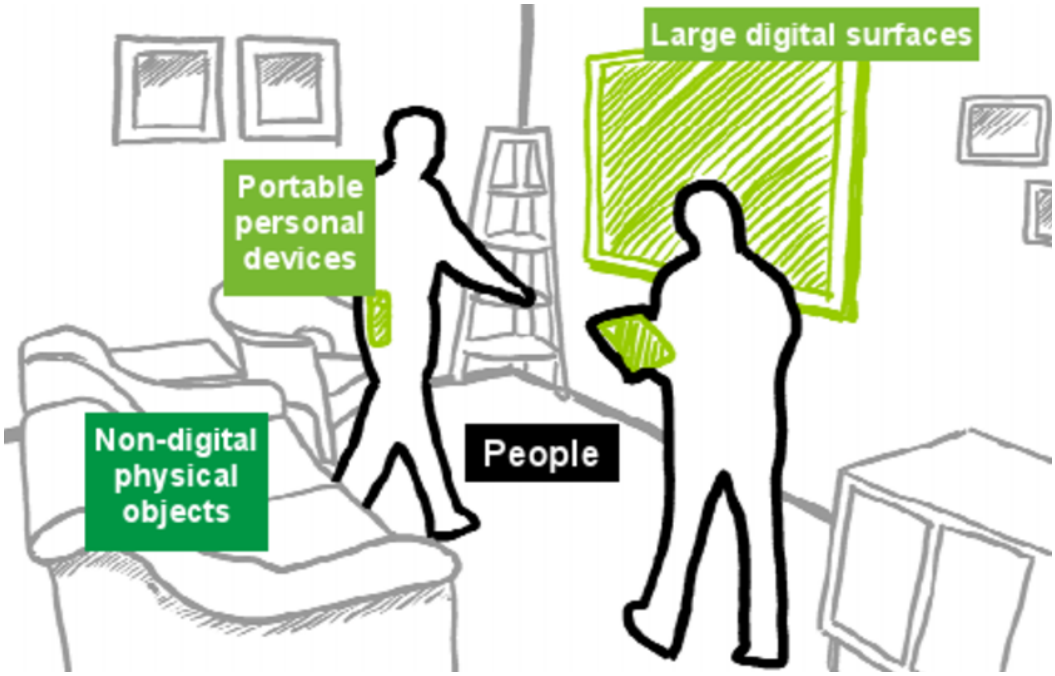


Fig. 2.5 Proxemic interactions associate people to digital devices; digital devices to digital devices; and non-digital physical objects to both people and digital devices [9].

Proxemic interactions allow enhancing connectivity and interaction possibilities when in proximity to people, other devices, or objects for building context-aware systems [65]. Proxemic interaction empowers the user to control digital devices, reducing physical contact with hardware without losing semantic interactions [9, 59, 64]. However, the proxemic interactions are not widely implemented in developing mobile applications. The adoption of mobile technologies in our daily life is growing in a way without precedent. Users can interact with diverse contexts through electronic devices (e.g., personal mobile phones, tablets, and wearable technologies) to accomplish their daily activities. Many of these activities require a specific HCI. In our dissertation, we implement proxemic interactions on mobile devices for improving the user experience. In the following, we describe the DILMO proxemic dimensions that have been proposed by Greenberg [41].

2.3.1 Proxemic DILMO Dimensions

Proxemic DILMO dimensions allow determining relationships among people and entities (digital devices and non-digital devices). Each DILMO dimension can also be analysed in

a variety of ways according to the measures that can vary by accuracy and the values they return (i.e., discrete or continuous). In the following, we describe the concept of DILMO dimensions, proposed by Greenberg [41].

2.3.2 Distance

Distance is a physical measure of separation between two entities, according to how they interact [41, 66] (see Figure 2.6(a)). The distance is used as a discrete parameter with the purpose to assign a proxemic zone based on Hall's theory. The zone allows the users to interact with the display or devices in different proximity [88]. Typically, short distances allow high-level interactions and the user could select multiple options to interact with the surface (see Figure 2.6 (a), while long distances reduce interaction between the user and a device (see Figure 2.6 (b)).

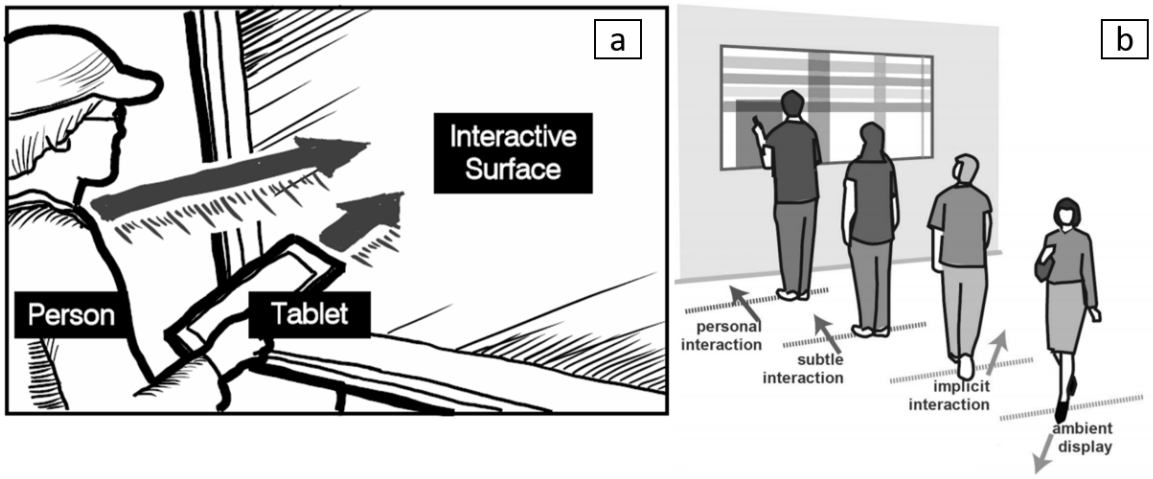


Fig. 2.6 Distance: (a) distance between devices [64]; (b) distance between device and human [88].

2.3.3 Identity

Identity is a term that mainly describes the individuality or role of a person or a particular object that distinguishes one entity from another one in a space [41, 64]. The identity is used for controlling spatial interactions between a person's handheld device and all contiguous appliances in order to generate an effective appliance control interface [59] (see Figure 2.7).

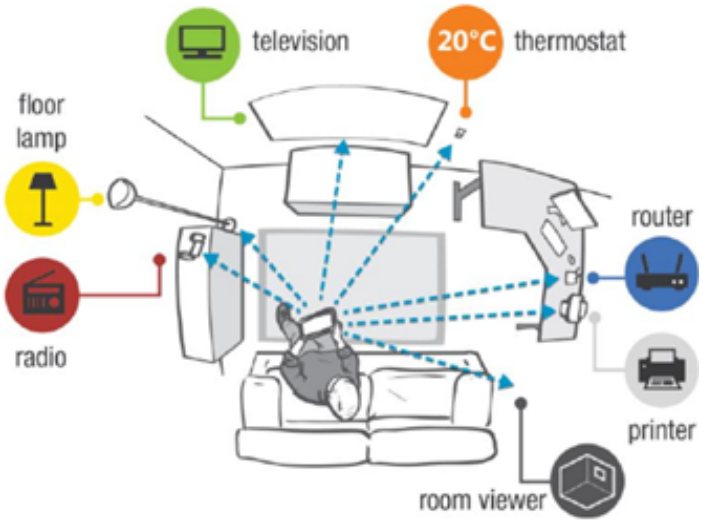


Fig. 2.7 Identities in proxemic interaction: Appliances control base on user's identity [59].

2.3.4 Location

Location defines the physical context in which the entities reside. The location allows relationships of entities with objects which are categorized as fixed (e.g., room layout, doors, and windows) and semi-fixed objects that are changeable, such as chairs, desks, lamps [41, 59] (see Figure 2.8). It is an important factor because other measures may depend on the contextual location. Location provides the entity's positions in the space, that can be assessed at any time.

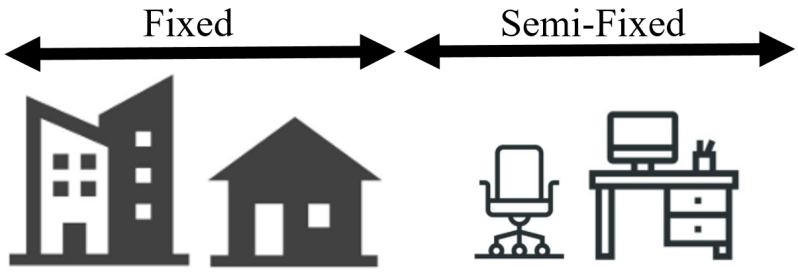


Fig. 2.8 Fixed and semi-fixed feature space.

2.3.5 Movement

Movement is defined as entity's change of positions over the time [41, 59]. Figure 2.9 shows when a user walks towards the screen or approaches it the content of the screen is adjusted according to the speed user's movements. The movement includes the directionality allowing interaction between the user and the application.

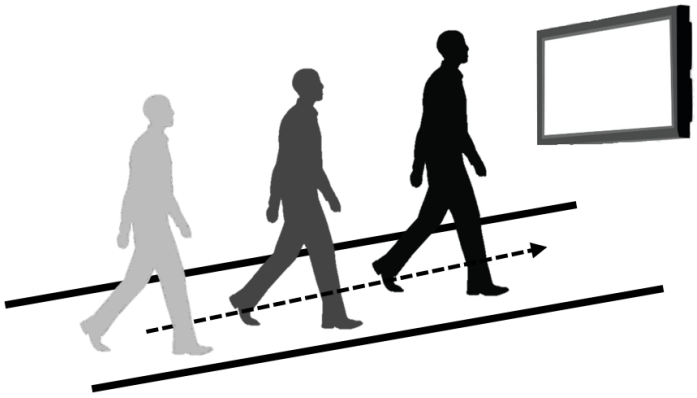


Fig. 2.9 User's movement towards the screen.

2.3.6 Orientation

Orientation provides the information related to the direction in which entities are facing between each other (see Figure 2.10). It is only possible if an entity has a "front face" and the entity can be detected in the visual field of another entity. Orientation can be continuous (e.g., the pitch/roll/ yaw angle of one object relative to another) or discrete (e.g., facing toward or away from the other object) [41].

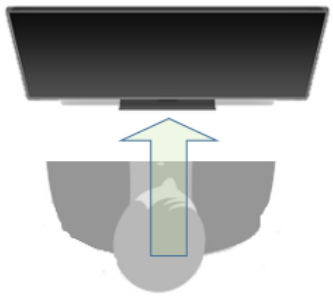


Fig. 2.10 User is facing to the screen

2.4 Conclusion

This chapter explained the theory of proxemics, proposed by the anthropologist Edward T Hall, which describes the human use of space and the people's behavior in a cultural context. Hall demonstrated the importance of his approach for effective communication among people using four distance zones (intimate, personal, social, and public). Furthermore, he characterized the territory as fixed-feature space and semi fixed-feature, where locations allow people to do activities of daily life. According to the theory of proxemics, human

behavior can be affected by the design of fixed-feature space or semi fixed-feature space, that govern every human interaction.

We explained how the HCI allows information transmitting through communication channels between humans and computers and how they have been implemented in ubicomp environments for improving the HCI. In this sense, the social theory of proxemics has inspired researches to design features to provide fluid interactions between digital devices and people. Proxemic interaction contributes to more natural experiences for HCI. Through this study, we have explored the concept of proxemic interactions and each proxemic dimensions, called DILMO (Distance, Identity, Location, Movement, and Orientation), whose measurements define the behaviors of entities in ubicomp environments.

In the next chapter, we continue describing how proxemic interaction has been implemented by previous studies and tools that help the development of proxemic systems.

Chapter 3

Related Work

In this chapter, we first present a review of studies that show how DILMO proxemic dimensions have been addressed with the purpose of implementing interactions among digital devices and among devices and people. Then, we describe studies that support the software engineering for developing proxemic systems – i.e., tools and frameworks for the design and implementation of proxemic applications. Finally, we introduce technologies that provide a wide range of possibilities for sensing proxemic information.

3.1 Systems based on DILMO Proxemic Dimensions

Proxemic interaction is a remarkable interaction technique that allows users to control digital devices in a natural way [9, 59, 64]. Previous proposals have implemented multiple proxemic applications based on DILMO and focused on developing proxemic applications for specific functionalities. From this review, we want to highlight that these proposals manage all DILMO proxemic dimensions or a subset of them.

Regarding the **proxemic dimensions**, the majority of works use all **DILMO** dimensions, while other works use just a subset of them. DILMO proxemic environments, in which all dimensions are considered are described in [9, 10, 14, 16, 32, 39, 54, 59, 64, 67, 70, 76]. The work presented in [16], illustrates how the proxemic dimensions can support interaction among entities (people and objects), with a context-aware framework. The approach introduced in [76] proposes the use of the body-tracking capabilities of kinect sensors to obtain the distance and react according to the location, movements, and orientation.

DIMO proxemic environments are considered in the studies presented in [12, 90]. The work applied computer vision and OptiTrack camera to measure the mobile device's distances between the user and device, allowing interaction with a proximity-aware mobile navigation. application[12]. Proxemic Peddler [90] is a display that reacts to users' movement monitoring

the passerby distance and orientation with respect to the display at all times. Additionally, **DLMO** proxemic environments are presented in [43, 42] in the context of cross-device games and cross-device interactions. Games' actions are defined according to distance, location, movements, and orientation of devices manipulated by children [43]; while in [42] cross-device interactions are conducted by these dimensions with the purpose to share information among digital devices. In [21], a proxemic environment is described in the context of an application that allows the recognition of materials based on low-cost mobile thermal camera integrated into a smartphone. This application measures the physical distance between the camera and the material and recognizes a specific texture (identity) a. The application provides a tool that records the location of potholes in a road based on **DIL**. Works based on **DILM** environments are presented in [31, 81]. The Multi-Room Music System proposed in [81] is a mobile app based on proxemic interactions that lets the user hearing the same songs playlist, while he changes his distance and location through the speaker arrangement in the house.In [31], entities have been implicitly managed distance, movement, and location dimensions have been used to implement interaction between user and screen based on interaction zones. Proxemic environments reacting to distance, identity, and orientation (i.e., **DIO** proxemic environments) are presented in [33, 44, 94]. Multi-View distinct [33] views from a single display compared to the angle of orientation between two users. This work used computer vision technology that allows user identification. In [44] shows the use of a built-in compass in smart-phones to support pairing them based on the orientation. Distance and location **DL** of people are considered in [85] to define different actions in an interactive display. Distance and location are also used for adapting visualizations on displays based on the users' distance relative to the screen and generate multiple views of the information displayed.

Table 3.1 summarizes the comparison of the described studies according to how DILMO dimensions have been addressed. Distance dimensions has been frequently used to implement interaction between a user and a screen where the physical distance has been implemented to define the proxemic zones. In all of these applications, the interaction objects (i.e., people and devices) are considered entities; when the Identity dimension is implemented, these entities are explicitly identified in a particular role. Orientation provides information related to the direction in which an entity is facing. It allows identifying the front of an entity. Previous proposals have demonstrated how a user's orientation related to a display can improve user interaction. Location and movement dimensions are less used in the previous work. The location has been associated with the position of an object in the environment. The movement allows applications to adjusted screen views according to the user's movements.

Table 3.1 Systems based on DILMO proxemic dimensions

Reference	D	I	L	M	O
Ballendat et al.[9]	■	■	■	■	■
Bhagya et al.[10]	■	■	■	■	■
Brock et al.[12]	■	■		■	■
Brudy et al.[14]	■	■	■	■	■
Cardenas et al.[16]	■	■	■	■	■
Cho et al.[21]	■	■	■		
Dingler et al.[31]	■	■	■	■	
Dostal et al.(a)[32]	■	■	■	■	■
Dostal et al.(b)[33]	■	■			■
Garcia-Macias et al.[39]	■	■	■	■	■
Grønbæk et al.(b)[42]	■		■	■	■
Grønbæk et al.(a)[43]	■		■	■	■
Grønbæk et al.(c)[44]	■	■			■
Kim et al.[54]	■	■	■	■	■
Ledo et al.[59]	■	■	■	■	■
Marquardt et al. [64]	■	■	■	■	■
Mentis et al.[67]	■	■	■	■	■
Mojgan et al.[70]	■	■	■	■	■
Rector et al.[76]	■	■	■	■	■
Sørensen et al.[81]	■	■	■	■	
Vermeu et al.[85]	■		■		
Wang et al.[90]	■	■		■	■
Wolf et al.[94]	■	■			■

3.2 Software Engineering for the Development of Proxemic Interactions

Software Engineering allow designers and developers to focus on developing useful and accurate software. It is a discipline that organizes the systems development, separated into states to improve the design and implementation phases. Software engineering supports developing complex systems in a systematic way, including the development and building of computer systems software on a second hand. In order to increase the productivity of software engineering, tools to support the different phases of the software development process have been proposed, such Domain-Specific Languages (DSL) [87], tools to support the design of applications, and frameworks or the development. Next sections provide a brief review of these three aspects.

3.2.1 Domain-Specific Languages

DSL is an executable specification language that provides the appropriate notations for a specific problem domain [84]. It can be viewed as specification languages, as well as programming languages. DSLs can be supported by a compiler which generates applications. A DSL can be textual or graphical that allow modeling a particular system. DSLs are a code structure designed to make it easy for humans to understand particular domain [36]. DSLs are focused on four key elements to solve a problem for particular domains:

- **Computer programming language**: Humans use a DSL to instruct a computer to do something.
- **Language nature**: DSL is a programming language that makes sense when the expressiveness comes from composed expressions and individual expressions.
- **Limited expressiveness**: A general-purpose programming language is to provide support abstract structures in a specific domain that helps to build aspects of the computer system.
- **Domain focus**: A limited language is useful only if it has a clear focus on a small domain.

Hundreds of DSLs are currently used for different areas such as financial products, software architectures, web computing, robot control, and digital hardware design. The DSLs are used in software engineering to increase the productivity of the development process [87]. A DSL provides a collection of sentences in a textual or visual notation with a formally defined syntax and semantics. The structure of the language sentences must be based on a meta-model with abstract mathematical semantics. Therefore, the properties and behavior of a DSL model should be predictable [87].

In the context of ubicomp environments, we did not found many studies proposing DSL to support the development of ubicomp applications and we did not find any DSL helping to modelling proxemic applications. Only the wok proposed in [40] shows the modeling of a system that describe the user requirements about the different services based on DSL for specifying pervasive systems [40]. Thus, there is a lack of DSL that allows designer to describe proxemic interactions, in order to improve HCI in ubicomp environments.

In previous studies were limited to the implementation of applications for specific task which have been described in section 3.1. Concerning design tools, only two of them aim to design proxemic interactions [32, 54] which will be described in next section 3.2.2 . There is also two frameworks in order to develop applications based on proxemic interactions [16, 64] that will be described in section 3.2.3.

3.2.2 Tools for Designing Proxemic Applications

Software design is the process to convert the user requirements into some suitable form, which supports the developer in software coding and implementation. Table 3.2 summarizes the tools for supporting the design of proxemic environments.

Table 3.2 Tools for design proxemic enviroments

Design tool	Graphical notation	Proxemic modelling	Hardware requirement
SpiderEyes [32]	no	yes	yes
Ministudio [54]	yes	yes	yes

SpiderEyes [32] is a system that offers a visual tool that allows designers to create collaborative proxemic applications for Wall-Sized Displays. This work develops proxemic applications for adjusting visualizations on displays. The visualization design tool is a web application that requires setting the values to distinguishes active users passing by in the background and foreground based on whether their visual attention is on display or not and only displays visualizations for active users (see Figure 3.1).

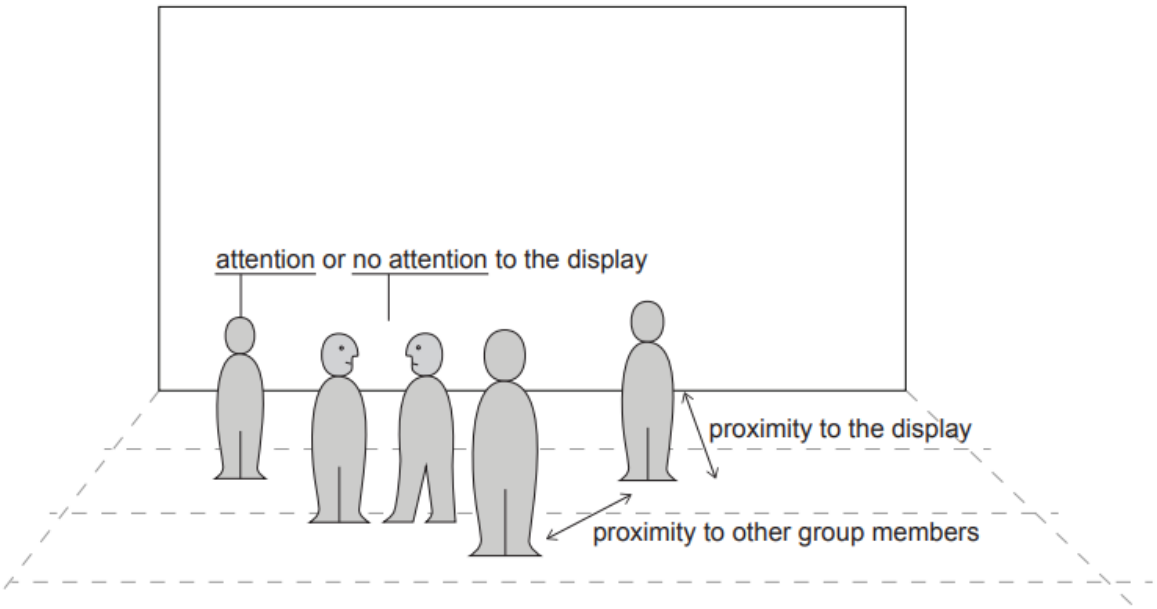


Fig. 3.1 Proxemic interaction to support collaborative scenarios 3.1.

This system shows how people's orientation and distance to the display can support collaborative activities around large wall-sized displays. Nevertheless, it does not propose a graphical design to enable end-users the general modelling of proxemic behavior.

Ministudio [54] is a tool for designers without technical implementation skills, which can be used to build miniature prototypes for proxemic interactions in the design phase. This tool illustrates how the proxemic dimensions can be used to define spatial relationships and interactions among people, devices, and objects using designers' familiar software and materials. Ministudio is based on tangible interaction interface, which uses miniature models on paper and projected images. The tool is composed of three functional components (see Figure 3.2):

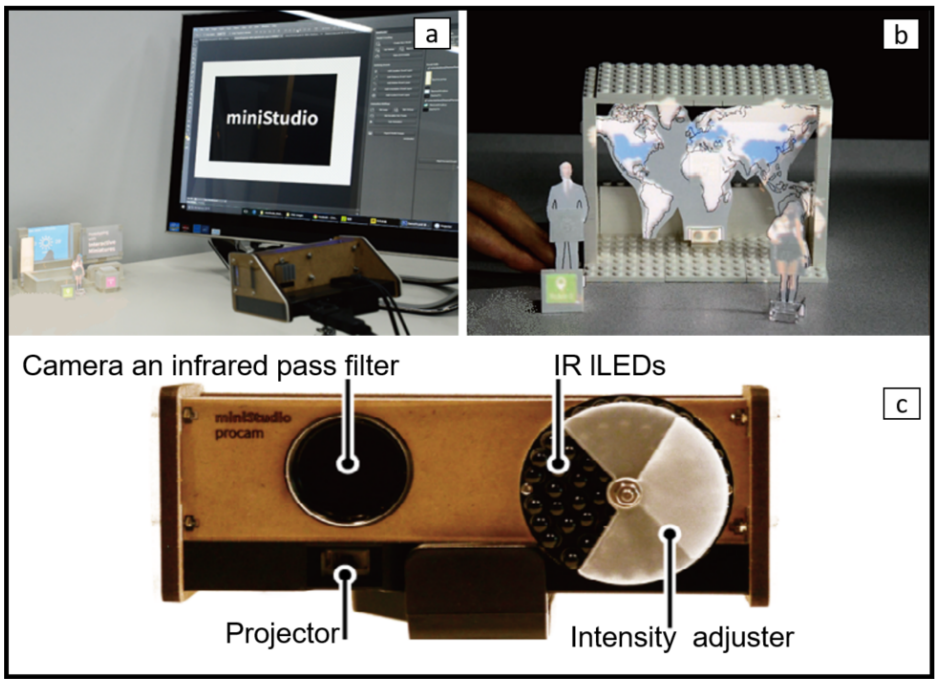


Fig. 3.2 MiniStudio Components: (a) a photoshop plug-in application; (b) tangible support materials including hidden AR stickers (c) a camera-projector system [54].

(**a**) Photoshop plug-in for generating miniatures and organizing interactions.

(**b**) Tangible support materials consisting of hidden augmented reality stickers and event icon.

(**c**) Camera-projector system for projecting images on physical miniatures.

This work shows that MiniStudio supports rapid designing of large and complex ideas with multiple connected components for prototyping ubicomp environments in the design

phase. However, the implementation of the tool needs specific hardware and material for creating miniature models such as Augmented Reality marker and camera-projector and it does not offer a graphical notation to allow the designer to model proxemic behaviors. Therefore, this tool is limited to be implemented on a large scale. Specific hardware requirements limit the use of SpiderEyes and Ministudio for the design of mobile proxemic applications

Next, we discuss frameworks proposed to build proxemic systems based on processing proxemic information from an ample range of sensors.

3.2.3 Frameworks to build Proxemic Systems

Some frameworks have been proposed to support the development process and complement the software engineering [16, 64]. In [64], a framework called Proximity Toolkit, used to discover novel proxemic-aware interaction techniques is proposed. The framework is a guide on how to apply proxemic interaction design for domestic ubiquitous computing environments. It is used to discover novel proxemic aware interaction techniques. The framework manages DILMO dimensions, which allow to determine relationships between entities and people. The framework architecture has four main components (see Figure 3.3): (a) a Proximity Toolkit server (b) a Tracking pug-in (c) a Visual Monitoring tool, and (d) an API. The Proximity Toolkit server is the central component in the distributed client-server architecture which allows multiple client devices to access the captured proxemic information. The Tracking plug-in contains two plugins: the marker based VICON and the KINECT sensor, that tracking of skeletal bodies and stream the raw input data of tracked entities to the server. The Visual Monitoring tool permits developers and designers to visualize tracked entities and their proxemic relationships among digital devices.

The API is a collection of libraries developed in C that use spatial information and relations among objects and space for processing proxemic information from ubicomp environment. This framework allows rapid prototyping of innovative interfaces processing proxemic information from sensors. However, the implementation of this framework requires a hardware architecture based on fixed devices (e.g., a Kinect Depth sensor and a client-server architecture) for allowing the server to process the proxemic information from appliances. This solution does not offer the mobility and portability required for implementing proxemic interactions on mobile devices or wearable technologies [6]. With the current Proximity Toolkit version, it is not possible to obtain proxemic information from the new smartphone's sensors capabilities and ensuring that users can use proxemic mobile applications on their smartphones in any place.

The work presented in [16], illustrates how the proxemic dimensions can support interaction among entities (people or objects), with a proposed context-aware framework.

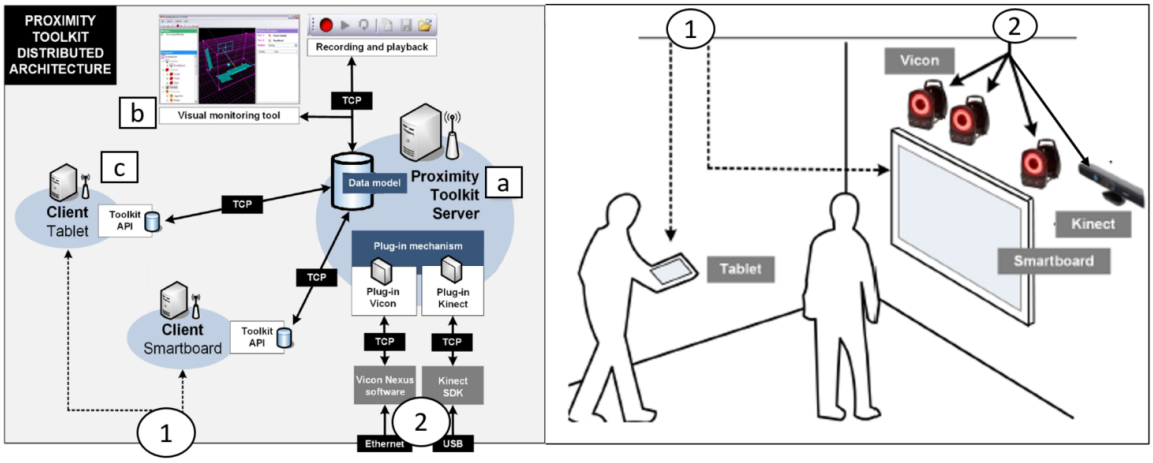


Fig. 3.3 Proximity Toolkit Distributed Architecture: (a) Proximity Toolkit server; (b) Tracking plug-in; (c) Visual Monitoring tool; and (d) Application Programming Interface (API) [64].

This framework used mobile sensors that provide portability for sensing DILMO proxemic dimensions. However, this framework needs an active internet connection to process proxemic information from mobile sensors. This framework does not offer methods that allow developers to implement proxemic interactions based on mobile computer vision. Furthermore, neither of the two current frameworks provides an API that enables developers to process proxemic information from mobile devices sensors. Both frameworks are limited to built proxemic mobile applications, while ProximiThings [16] cannot process DILMO dimensions without a server connection. Table 3.3 summarizes technologies used by previous frameworks and also describes constraints for building proxemic mobile applications. We described the hardware and software that previous frameworks have been used for sensing proxemic information.

Table 3.3 Proxemic Frameworks Technology

Framework	Technologies	Portability	Offline service	Mobile API
Toolkit [64]	Kinect Depth Camera Vicon motion capture	Low	yes	No
ProximiThings [16]	Infrared Sensors IR Accelerometer Gyroscope Magnetometer	High	No	No

All these works demonstrate the current interest for researchers to develop tools that support the design and implementation of proxemic applications. Nowadays, smartphones and mobile technologies are powerful and offer a wide range of possibilities to improve user interaction. There are about 2 billion people with smartphones, which represents one-quarter of the global population in the world, and this trend is on the rise [38]. It is therefore of great importance to provide tools for developing mobile apps and improve HCI on mobile devices.

3.3 Technologies for sensing DILMO Proxemic Dimentions

In this section, we describe the technologies used in ubiquitous environments for sensing each DILMO from a variety of sensors. We describe the technologies that have been used for the development of proxemic systems by previous studies. Table 3.4 summarizes the sensors used by proposals to obtain proxemic information listed in Table 3.1.

Table 3.4 List of sensors for obtaining DILMO proxemic dimensions

Sensor	D	I	L	M	O
Microsoft kinect (Computer Vision)	✔	✔	✔	✔	✔
Vicon/OptiTrac Motion Capture	✔		✔	✔	✔
Leap Motion				✔	
LV-MaxSonar-EZ1	✔				
SHARP GP2Y0A02YK0F	✔				
Bluetooth BLE	✔	✔			
Mobile Sensors (accelerometer, gyroscope, magnetometer)			✔	✔	✔
Mobile Thermal Imaging	✔	✔			

Distance is defined in Chapter 2.3.2. The work presented in [76] proposes the use of the body-tracking capabilities of **Microsoft Kinect** Sensors to obtain the distance. Authors demonstrate the suitability of their proposal in an application that measures the distance between blind people and paintings, according to which it provides different background music experiences. In [12, 67], computer vision is used for measuring the distance from the device hosting the program to the user. **Vicon tracking** system has been used to estimate distance between the user and a wall display in [9]. However, the Vicon tracking system is expensive and thus it is not a realistic mobile platform. In work used, [39] the user wearing the ultrasonic sensor **LV-MaxSonar-EZ1** that allows the detection of objects at distances between 30 cm to 6.45 meters. The **Sharp GP2Y0A02YK0F** is employed for measuring the distance between the visitors and an exhibit which is around of interaction zones [94].

Bluetooth Low Energy (BLE) technologies allow the device to estimate the proximity among entities by Received Signal Strength Indicator (RSSI) and Broadcasting Power value (TX power), as in [96, 92]. The BLE technology is currently used with Beacons which are small Bluetooth radio transmitters that broadcast BLE signal frequently employed in proximity marketing (see Figure 3.4). Furthermore, these are used to support human interaction. The work presented in [20], shows how blind people or visually impaired users equipped with smartphones can interact with beacons in order to receive assistance. In that work, the resolving of proximity detection (distance) takes great relevance. This study shows the advantage of implementing BLE Beacon.

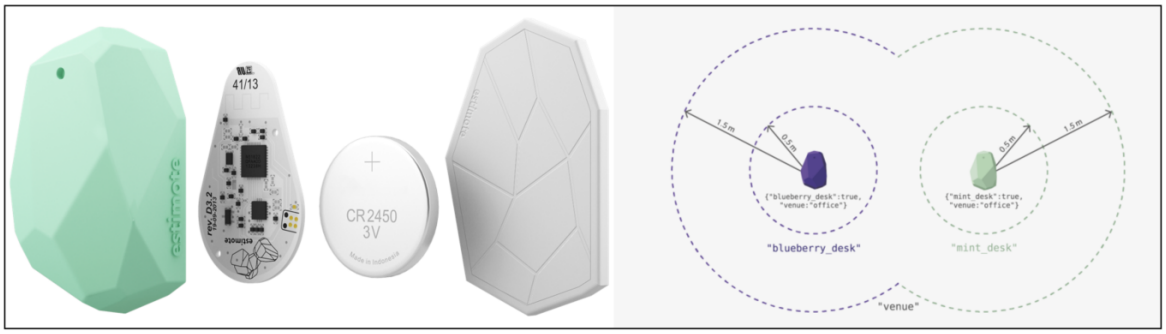


Fig. 3.4 BLE Beacons [1]

The work described in [28], implements beacon to rescue persons caught in an avalanche (skiers). Authors propose the use of two physical beacons: transmitter and receiver, one for victim and the other for the rescuer. Beacon transmits an electromagnetic signal, allowing achievement of the goal. This work has demonstrated how the beacon technology has contributed to save human lives. **Mobile Thermal Imaging** allows to estimate proximity which is based on the use of mobile thermal imaging and deep learning [21]. However this is not a standard sensor for all smartphone. **Mobile Computer Vision** is implemented to determine the distance between the user and the display using mobile device's camera [12].

Identity is defined in Chapter 2.3.3. SpiderEyes [32] uses a **Kinect Depth Camera** that allows identification of entities. In this particular regard, it was indispensable to have user identification. User's interaction is based on their identity and distances with the display. Kinect combines three kinds of information to recognize users' identity such as face, clothes, and body skeleton using an RGB camera which is based on the three primary colors red, green, and blue (RGB) [61]. The user identity is verified using a "truth-table". This table provides a final result based on a sequential process that start from face detection to height

[1] https://developer.estimote.com/lte-beacon/using-bluetooth/

estimation, in which face detection is the most reliable value. Table 3.5 [68] shows an example of a truth table for users authentication.

Table 3.5 A truth table for identifying a user

Characteristic	user 1	user 2	user 3	user 4
Face detection	Positive	Negative	Positive	Unknown
Clothing color	Unknown	Unknown	Negative	Negative
Height	Positive	Positive	Unknown	Unknown
Identification	✔	✖	✖	✖

Bluetooth BLE is a radio transmitter that broadcast BLE signals and Universally Unique Identifier (UUID) allowing identification of digital devices in the environment [96]. **Mobile Thermal Imagine** and a deep learning technique are used for sensing and recognizing the material [21]. Currently, **Mobile Computer Vision** is a powerful technology which is based on advanced image processing that allows to detect the human face.

Location is defined in Chapter 2.3.4. In the work presented in [76], location is used to detect events related to hands' tracking. For example, when a blind user explores a painting with his hands, the application uses the 3D coordinate system of the **Kinect depth camera** [69], to know the user's hand position in a specific region of the painting. In [43, 64], entities are associated with three-dimensional positions related to a fixed point that can be used for initial setting of smart environments. In such a way, it is possible to obtain the relative position among people and devices. **Vicon motion capture** system uses a reflective infrared markers to obtain the people's location and digital devices [9]. Multi-Room Music System [81] is an application that allows the user to hear the same songs playlist while changing his location. This application uses a comparison of received Wi-fi signal strength indicator (RSSI) measurements, handled by the smartphone application.

Movement is defined Chapter 2.3.5. It has been obtained using a **Kinect depth camera** and **Leap Motion** for capturing the movements of the hands with accuracy [31]. Motion capture uses data sensing from images to triangulate the 3D positions of a person or an object between one or more cameras. The **Vicon motion capturing system** [86] tracks infrared reflective markers to retrieve fine-grained information of the person's movements. Figure 3.5 shows the architecture of Vicon motion capturing system. The movement also allows gesture recognition through smartphones or wearable technologies by employing **Mobile Sensors** [6, 43], such as accelerometer, gyroscope, and magnetometer which are generally incorporated in modern mobile devices .

[1] https://www.vicon.com/hardware/

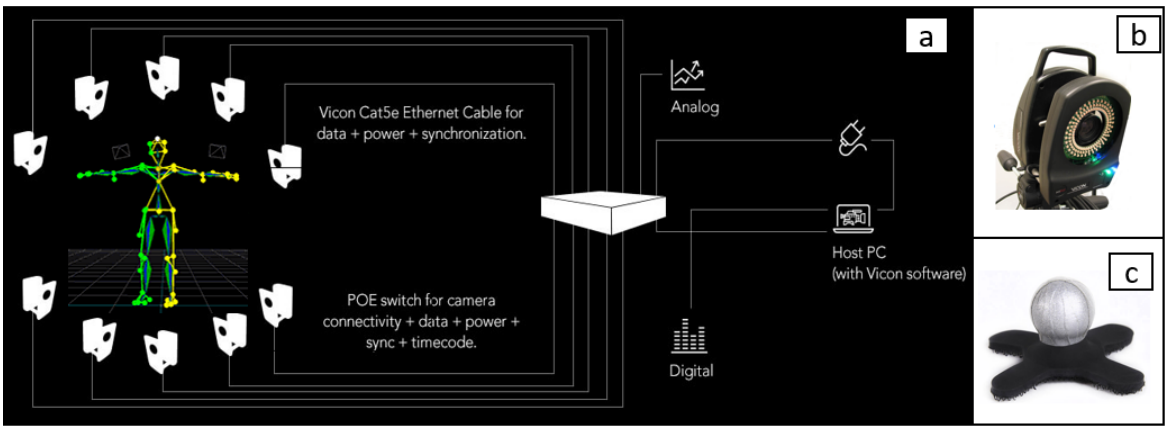


Fig. 3.5 Vicon motion capture systems[1]: (a) The Vicon 3D motion capture system; (b) Motion Capture Camera; (c) Vicon markers

Orientation is defined in Chapter 2.3.6. Previous works have demonstrated how a person's orientation related to a display can be used for improving user interaction [9, 43, 66, 71] using **Kinect depth camera** sensor and the marker-based **Vicon motion** capturing system to obtain the orientation between two entities. In [44], has used a built-in compass orientation in mobile devices to support pairing between mobile devices equipped with **Mobile sensors**.

Computer vision is frequently employed to obtain almost the whole DILMO proxemic dimensions using a Kinect depth camera. Kinect depth camera is powerful and low cost sensor that is frequently used in previous proposal for sensing proxemic interaction . Vicon motion capture technology is very accurate in seizing people's movement and objects in which users have to carry and wear Vicon marker sensors. Notwithstanding, theses sensor do not provide portability that allow developers to implement in mobile devices.

3.4 Conclusion

In this chapter, we have studied software engineering tools for the development of proxemic applications. We have described previous studies that have used DILMO dimensions or a subset of them. Previous solutions demonstrated the interest of researchers to develop tools that support the design and implementation of proxemic systems. However, our research found that there are needs from tools based on a symbology, in which designers without high-degree implementation skills can design proxemic systems.

From the analysis of previous proposals in Section 3.2.3, we note there exists a notable lack of general approaches able to support the design process to represent proxemics

behaviours. Each work proposes and uses a particular notation and tool; there is not a standard process neither a standard model to design and implement proxemic applications. We overcome these limitations by proposing a development approach based on a formal and graphical modelling supported by a DSL, that allows the designer to represent different proxemic behaviors in smart environments, based on different combination of DILMO dimensions. Thus, the design phase is supported by formal notations and graphical objects, to facilitate the design process.

Furthermore, the vast majority of mobile devices are equipped with robust hardware capabilities. These capabilities allow devices to process and obtain proxemic information on mobile devices; for example, by using sensors and cameras embedded in a smartphone. In turn, it is possible to implement proxemic based mobile applications that facilitate users' contact and interaction with other people and devices in indoor and outdoor spaces, which we call smart proxemic environments. Therefore, we set ourselves the challenge of creating a functional proxemic mobile application. In the next chapter, we illustrate the implementation of a proxemic mobile application that allows us to identified if there are needs of tools for the development of proxemic environments based on mobile devices.

Chapter 4

First Aid Mobile Application (FAMA): A Proxemic Application in Mobile Devices

To demonstrate the feasibility of proxemic interactions with mobile devices, we implemented a mobile application to solve a daily life problem. In this chapter, we describe both the application and the development process. We developed a First Aid Mobile Application (FAMA) as proof-of-concept using a subset of proxemic dimensions (i.e., Distance, Identity, and Movement) to model proxemic HCI that allows flexible interaction between people and devices. We describe the technical aspects and challenges related with the development of FAMA. We show the benefit of using proxemic interaction to implement HCI, in the context of emergency and rescue services, in terms of better performance and accessibility, compared with an existing first aid mobile application on the market.

4.1 Motivating Scenario

Nowadays, it exists a wide variety of first aid applications for Android and iOS devices to help users in unexpected or dangerous situations. However, their massive use is limited due to several aspects, such as the complicated user interaction or the wearing of special devices for medical identification. In order to find solutions to these problems, proxemic interaction seems to be an appropriate solution. We proposed a mobile application based on proxemic interactions and developed with Bluetooth Low Energy (BLE) Beacon technology. Furthermore, FAMA's development allowed us to achieve the first approach to implement proxemic applications with mobile devices for sensing proxemic information. The prototype shows that the use of proxemics allows users to faster identification and more efficient interaction.

4.2 First Aid Applications

In first aid situations, the unconscious persons commonly do not receive proper help until paramedics come on the scene. The medical identification is not available for people around the unconscious person, who could help meanwhile the paramedics arrive. In this regard, nowadays there exist a variety of first aid apps. Below we describe first mobile applications.

Today, we can find a variety of first aid apps for Android [4, 75]. Both applications [4] store emergency information for hospital staff, which can be accessed even when the is device locked. In such apps, the helper person can access emergency medical identification. However, these apps have the disadvantage that helper persons need to have a physical access to the injury person, in order to check the victim medical identification.

Another way to obtain the unconscious medical information is with the use of medical alert bracelets [1, 25] which are worn directly on the body by the user. These accessories allow sharing information directly or by interaction between the bracelet and the application on the smartphone. These solutions need additional accessories in which people must carry a bracelet. In contrast, in our proposal users only use the smartphone to provide aid and assistance. Below, we describe previous researches that allowed us to understand how to obtain the proximity between digital devices.

The work presented in [20], shows how blind people or visually impaired users equipped with smartphones can interact with beacons in order to receive assistance. In that work, the resolving of proximity detection (distance) takes great relevance. This study shows the advantage of implementing BLE beacon. The work described in [28], implements beacon to rescue persons caught in an avalanche (skiers). Authors propose the use of two physical beacons: transmitter and receiver, one for victim and the other for the rescuer. In a rescue situation, the victim and a rescue group need to work, both individually and collectively, the beacon transmits an electromagnetic signal, allowing achievement of the goal. This work has demonstrated how the beacon technology has contributed to save human lives. It shows the advantage of using proxemic interaction to implement HCI, in emergency and rescue services for better accessibility, compared with existing first aid mobile applications. However, the implementation of this solution requires sophisticated beacons based on electromagnetic signal which are not available on mobile devices. We proposed a FAMA based on proxemic interactions and developed using BLE Beacon technology to implement the application in mobile devices.

4.3 Problem

Previous proposals are focused on the identification of unconscious persons without taking into account the physical distance between people. In the context of first aid applications with unconscious persons, we have identified difficulties related to the interaction between unconscious person's app and helper persons. Generally, the helper person needs to access a physical device in order to see emergency identification on the injured person's smartphone. In other cases, the victim should wear special accessories (e.g., an alert id bracelet, smart watches, and NFC id card). With this kind of solutions, the intimate space of the unconscious person or victim can be affected by helper. Besides, the smartphone is a mobile device commonly used by people that allow the user to manage social distancing. Our aim was to identify unconscious people, facilitating natural interactions between human and mobile devices, without touching the unconscious person.

4.4 FAMA: Proxemic-based First Aid Mobile Application

This section describes our prototype based on the proxemics' theory to manage social proxemics distance among people. We illustrate the benefits of integrating this theory for first aid mobile application. Proxemic interaction provides foundational principles allowing users to have a seamless interaction with digital devices [9, 41]. These studies have demonstrated that DILMO proxemic dimensions help to create appropriate HCI for applications, while reducing physical contact with digital devices. The development of FAMA, our first aid application is a prototype that takes into consideration a subset of proxemic dimensions (distance, identity). The implementation of proxemic interaction allowed us to develop a novelty mobile application.

- Distance allows us to determine interaction zones. We evaluate the distances in order to fix general criteria of communication between both devices (unconscious person's device and helper's device).
- Identity allows us to identify a unconscious person. When a person changes his phone settings to a helper mode, he/she could identify people around him/her.

For our FAMA prototype, we define three interactive zones: Rescue Zone, Alert Zone, and Neutral Zone to offer better opportunities to unconscious or injury person to receive appropriate rescue aid. Each interaction zone was established in relation to distance between unconscious and helper person. Figure 4.1 shows the interaction zones for FAMA, these zones are inspired on previous researches [88].

- **Rescue Zone (0-1m)**. Represents the intimate and personal space. In this zone, the helper can interact with mobile apps and reports events.
- **Alert Zone (1m-4m)**. It represents the social space. People in this area, related to the unconscious person, can see the alert on their smartphones.
- **Neutral Zone(4m)**. It represents the public space where nobody receives information.

FAMA explores the use of the proxemic concept, to manage effectively social proxemic zones that provides rapid identification of unconscious people, generates alerts, and shares medical identification in a secure way. In this context, FAMA was developed with BLE Beacon technology. The identification of the unconscious injured person can be accessed from other smartphones by BLE.

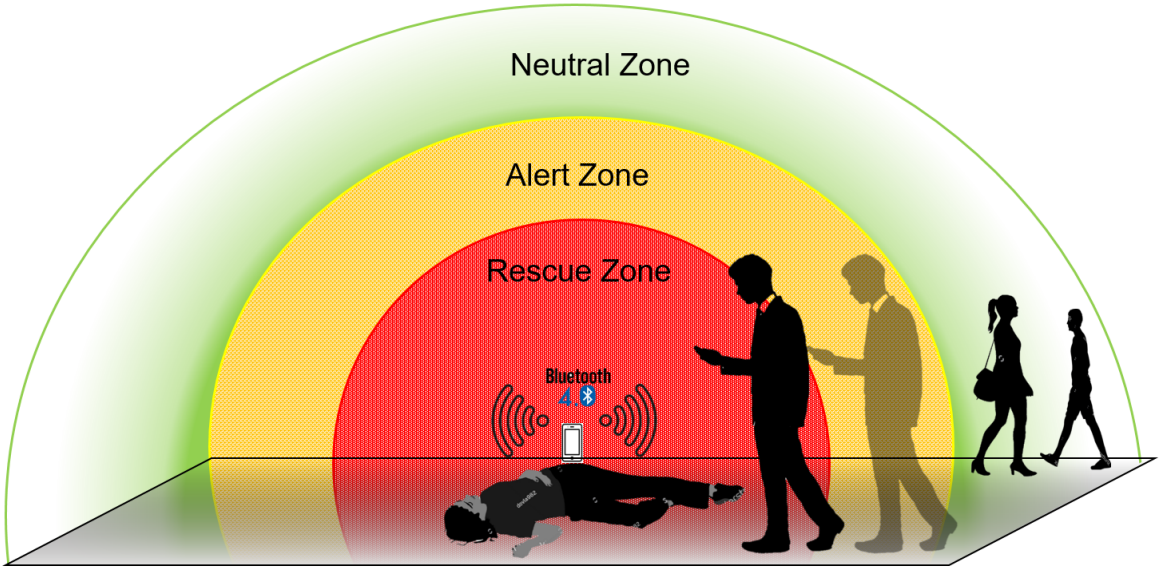


Fig. 4.1 FAMA's zones and interactions between users in the public space

4.5 Scenario the exploration study

To illustrate our scenario, we consider a person in a public space, who feels weak and is about to faint. His/her smartphone will act as a virtual BLE beacon and will start to share emergency identification by establishing a Bluetooth connection. In the current version of FAMA, the user has to explicitly start the S.O.S mode by pressing a button (see Figure 4.4 This scenario allows us to explore the theory of proxemics and manage social proxemic distance by implementing a first aid mobile application that provides rapid identification

of unconscious people, generates alerts, and shares people's identification securely. This emergency identification can be read by the smartphone of a helper person, exclusively when this is moving toward the Rescue Zone. People in the Alert Zone will then be able to see a notification; however, people in the Neutral Zone will not receive any notification. This scenario describes unexpected episodes in a public space, assuming that people carry their smartphone practically everywhere and at all times.

To evaluate the effectiveness of FAMA, we asked five PhD students to be in a university classroom (Neutral zone). We provided four smartphones running our prototype to theses students who play the role of helpers (H_1,H_2, H_3, H_4) and one for the unconscious or injured person whose device, in addition of our prototype, is running Medical ID (MID), an available first-aid mobile application. We choose MID [75], because it has the same purpose of our prototype related to obtain emergency identification of unconscious or injured person. Also, Medical ID has received good comments from users on the google play platform, and the application provides a web site that describes the application to users.

We measured time from which the helper person tries to identify an unconscious person until the moment a helper H_n obtains emergency identification. This time was measured, both for FAMA and MID. For MID the helper must search the smart cellphone inside the pants pocket or handbag of unconscious or injured person. In this case, the unconscious or injured person's intimate space could be affected by the helper when the helper was trying to find an unconscious person's smartphone. Also, it is very difficult to access an injured person's smartphone without moving it. While for FAMA the helpers person used the proxemic interaction application running on their smartphones in order to obtain the emergency identification of unconscious person without invading the intimate zone. In view of the foregoing, we have formulated two hypothesis as follows.

Hypotheses 1: Is it possible that a helper person can interact with unconscious person's smartphone, in order to obtain his/her emergency identification, without physical contact and reduce the recognition time with respect to current applications ?.

Hypotheses 2: Can An unconscious person or victim share emergency identification without explicit interaction with a device ?.

4.5.1 Results

This section describes the result of our proposed FAMA in the scenario of exploration, compared with existing first aid mobile application (MID solution). The experiment allows us to verify each question that was previously formulated to evaluate the benefits of using proxemic interaction with mobile devices.

Hypotheses 1: This first question is verified utilizing the FAMA application installed on mobile devices. Our application provides an unconscious person's identification faster than the MID mobile because the helper person does not need physical access to the injured person's mobile phone. The helper can obtain an unconscious identity base on social distancing. For MID, the recognition time can be long, depending on where the unconscious person carry his/her phone such as jean pocket,jacket pocket, and handbag. This could represent a huge waste of time to identify the unconscious persons by the helper. Figure 4.2 shows the recognition time in seconds of each helper H_n person by using our proposal prototype FAMA and MID solution. In our prototype, the recognition time is faster because the helper person does not need to physically access the smartphone of the unconscious person and unblock the unconscious person's smartphone. This experiment demonstrates FAMA's functionality and the successful use of proxemic interaction to identify an unconscious person using our first aid mobile application.

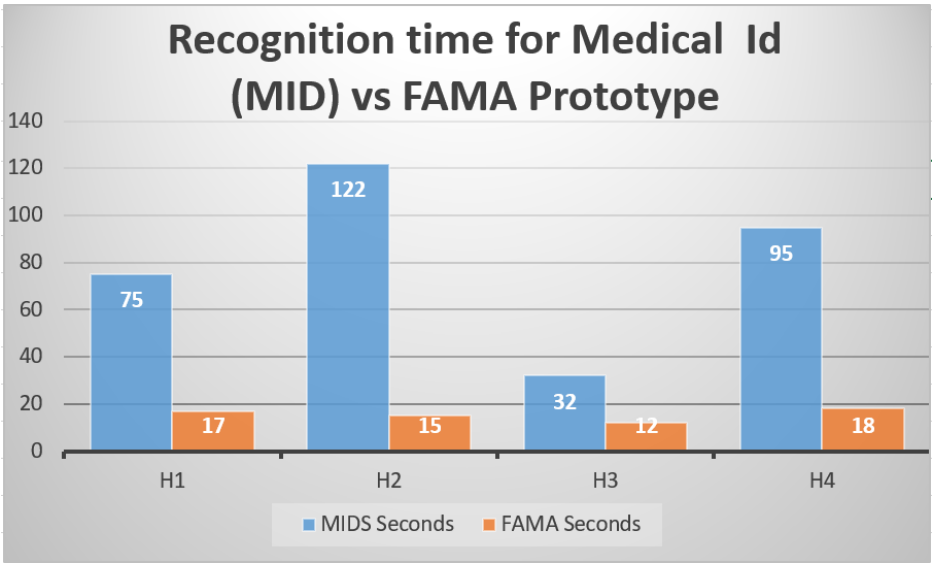


Fig. 4.2 Recognition time for Medical Id (MID) vs FAMA Prototype.

Hypotheses 2: The helper person accesses his own smartphone for recognizing unconscious person. Figure 4.3 shows results of a user survey related to the accessibility of the applications FAMA and MID, by participants who chose a scale from 1 to 10 (10 being the best). When we talk about accessibility, it holds features as the rapidness of obtaining medical identification and respecting the unconscious person's privacy. Our prototype scored better results than MID, according to the survey applied to each helper person.

These primary results confirm the potential benefit of using proxemic interaction to implement HCI, in the context of emergency and rescue services, in terms of better performance and accessibility, compared with existing first aid mobile applications.

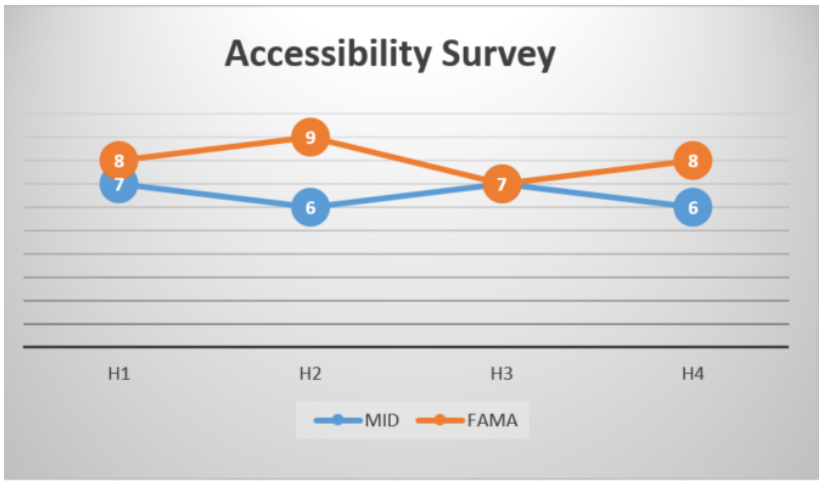


Fig. 4.3 Opinion score (on a scale from 1 to 10, 10 being the best) relate to accessibility.

4.5.2 FAMA Implementation

FAMA was developed for Android BLE mobile devices. It consists of two modules: Register and Helper modules. Figure 4.4 shows a print screen for the Register module. This is dedicated to create an alert when the unconscious or injured person demands help. The initial conditions must be filled out by users in order to share his/her emergency contact number. The Helper module is able to see emergency contact number identification of unconscious or injured person, depending on the helper person's location in the interaction zones. Figure 4.5 shows a print screen for the Helper module.

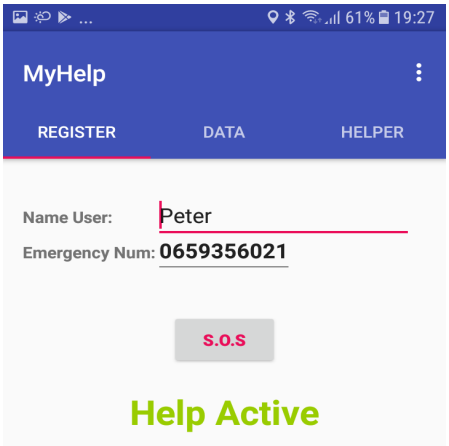


Fig. 4.4 Graphical user interface for creating an alert.

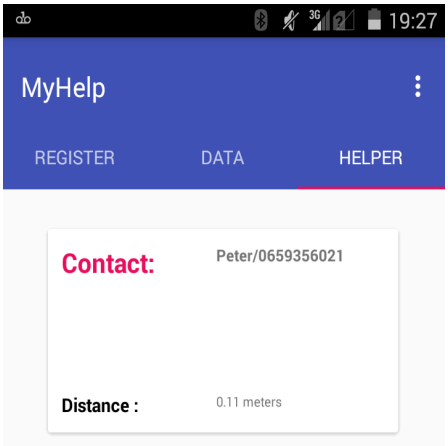


Fig. 4.5 Graphical user interface for helper module.

Next, we describe technical aspects of our prototype's development based on proxemic interaction and developed with BLE Beacon technology.

Bluetooth Low Energy (BLE)

BLE is used in order to obtain both the physical distance and identity between two mobile devices. In contrast to Classic Bluetooth, BLE is designed to provide significantly lower power consumption. This Technology allows Android apps to communicate with BLE devices with stricter power requirements, such as proximity sensors, heart rate monitors, and fitness devices. Android 4.3 (API level 18) introduces built-in platform support for BLE in the central role and provides native libraries that allowed us to implement and transmit proxemic information in small amounts of data between nearby devices using BLE Beacon technology. In the next, we describes how to create virtual beacon in mobile devices.

BLE Beacon virtualization

BLE is a digital radio protocol that allow us to create Beacon advertising base on proximity between BLE devices. To build virtual beacons for smartphones, it was required to use beacon protocols. In our prototype, we need to transmit a string data that allow user to identify unconscious person. This data is used to prove that a virtual beacon can diffuse different kinds of information by interaction zones. We use the Java library AlteBeacon [3]. Figure 4.6 shows FAMA's code which is based on AlteBeacon protocol for both create alert in proxemic interaction zones and obtain emergency identification.

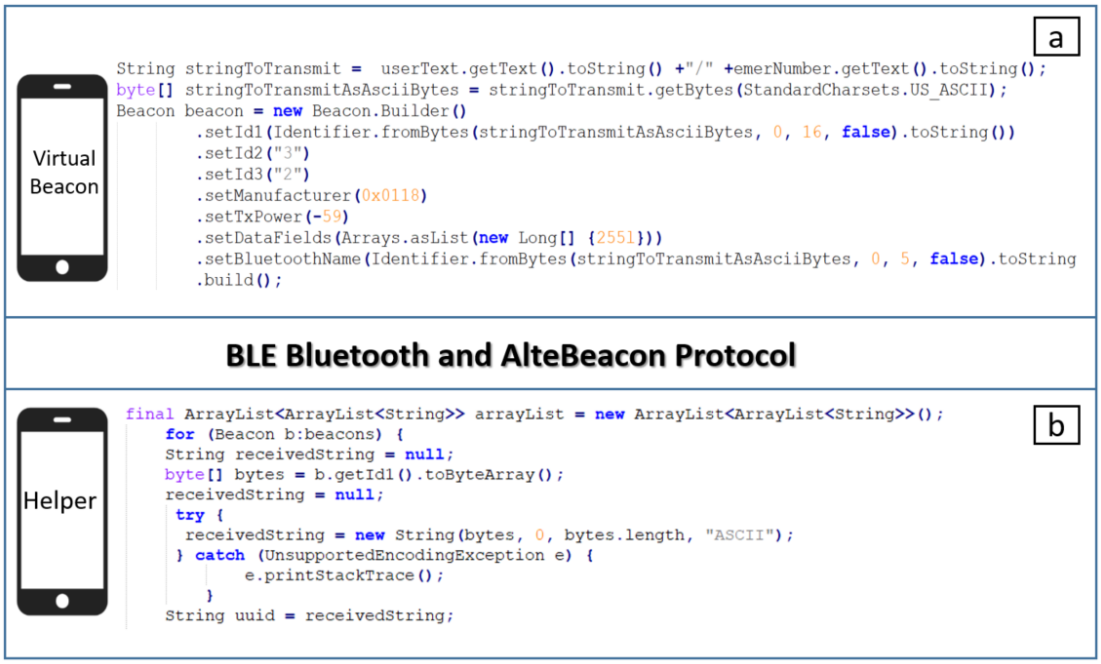


Fig. 4.6 FAMA's Architecture code: (a) Code for creating a virtual Beacon with Mobile devices; (b) Code for detecting unconscious person's smartphone.

Beacon protocol offers the possibility to transmit encoded string with AlteBeacon methods availed that we allow helper person to identify unconscious person utilizing universally Unique IDentifier, UUID from AlteBeacon protocol. The UUID can be transferred between mobile devices using Bluetooth BLE connect without paring. Figure 4.7 below shows the global architecture that describes how proxemic dimensions (distance and identity) were considered in the application.

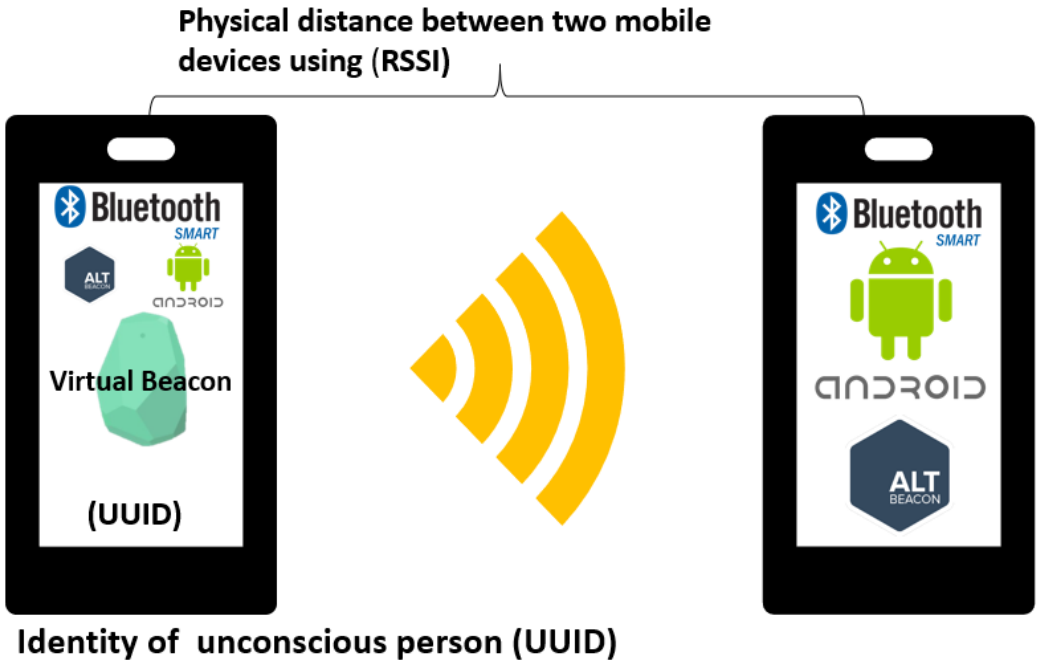


Fig. 4.7 FAMA's Global Architecture.

- The **distance dimension** is characterized through the signal strength of the Bluetooth that allowed us to estimate the physical distance between two smartphone.
- The **identity dimension** is implemented with UUID that allow the application to identify unconscious person.

FAMA architecture is base on Android and BLE libraries that allows developers to detect proximity between two mobile devices. Nevertheless, FAMA was developed without tools that support us to implement proxemic interactions in mobile devices.

4.6 Conclusion

We have developed FAMA, based on proxemic interaction that taking advantage of HCI. Through the first aid, the prototype illustrates that FAMA's interaction behaves better than the current first-aid applications. FAMA offers benefits for faster identification of unconscious or injured people, only using smartphones and avoiding physical contact between people. The scenario allows us to presents the results from the preliminary evaluation of our prototype by the users. Hypotheses showed the advantage of adopting proxemic interaction to implement HCI, in the context of emergency and rescue services, in terms of better performance and accessibility, related to existing first aid mobile applications. Implementing proxemic interaction with mobile devices can bring benefits to user experiences.

The current version of FAMA is focused on the improvement of interaction with first aid mobile applications. However, FAMA was implemented without tools that help us to design and implement proxemic mobile applications. The current proxemic tools do not offer feasible solutions for developing and implementing mobile applications. This proxemic mobile application development has motivated us to create solutions for supporting the design and implementation of proxemic mobile applications.

In the next chapter, we describe an approach for modelling proxemic environment based on a graphical DSL focusing on entities in the proxemic environments.

Chapter 5

A Graphical Domain-Specific Language for Modelling Mobile Proxemic Applications

As described in Chapter 3 there exists a notable lack of general approaches able to support design process to represent general proxemic behaviours. Thus, we propose a formal definitions and approach for modelling proxemic environments based on a graphical Domain-Specific Language (DSL). It allows non-specialist designers to express proxemic interactions for modeling proxemic environments and supports the development process. We have implemented a prototype to design proxemic behaviours in different proxemic environments.

5.1 Motivation to Develop a Graphical Domain-Specific Language

Intelligent environments are present in people's everyday life with the integration of digital devices. Proxemic interactions allow users to improve HCI experiences using these digital devices. As described in the Chapter 3, current studies in this area are focused on creating specific proxemic systems and few support is provided with DSL in order to better support the software development process. We aim to help design proxemic environments for non-specialist designers in order to express proxemic interactions among digital devices. We have created a graphical DSL to design and describe proxemic behaviours based on the notation system proposed by Edward T. Hall in [45]. He showed how people understand, interpret, and use space, especially related to the distance among people [35]. We chose a graphical DSL because it allows us to represent proxemic environments and translate into it

into an XML file that can be interpreted by the computer. Our work allows us to present a graphical notation that can help to design proxemic environment from contextual design. The formal definition provides the syntax for each symbol proposed that allows us to describe the proxemic environment.

The next Section 5.2 describes the formal definitions, the graphical, symbolic notations, and the XLM scheme description, including our proposed graphical DSL.

5.2 Formal Definitions for the graphical DSL

The formal definitions that govern our graphical DSL are mainly intended to define all components in a proxemic environment. A proxemic environment is described as a space containing a target entity that reacts according to DILMO dimensions of all other entities. The first terms that we need to formally state are related to the entities and the target entity that can interact in a proxemic environment. Def. 1 describes what we consider as entities and Def. 2 formally describes a target entity, that we call cyber physical system. With these formal definitions the proxemic environment (P_E) is defined. It means that proxemic zones (P_Z), the CPS, the entities E, the set of DILMO dimensions to be considered, and the initial conditions are established. From the formal definition, it is possible to define inference rules that can generate specific behaviors and actions of entities in the proxemic environment.

Definition 1 Entity (E). *An entity, denoted as E, represents an interaction object (e.g., a person, an object, a device), that can be identified or not according to requirement of scenario in the physical space.*

Definition 2 Cyber Physical System (CPS). *A cyber physical system, denoted as CPS, represents the target entity, from which a proxemic environment (P_E) is defined. All proxemic zones (P_Z) and DILMO dimensions are measured for all other entities (E) with respect to the CPS, and the interaction among them will be defined accordingly.*

We keep the same significance of DILMO dimensions from the original proxemic interaction proposal [9, 41]. Def. 3, Def. 4, Def. 5, Def. 6, and Def. 7 show our formal definitions for **Identity**, **Distance**, **Location**, **Movement**, and **Orientation**, respectively.

Definition 3 Identity (I). *The Identity, denoted as I, represents an entity (E) that has a unique identification in the proxemic environment.*

By analogy to Def. 1, an identity (I) is associated with a specific person or object.

Definition 4 Distance (D). *The distance is the physical measure of proximity between an entity (E) and the CPS; it is denoted as $E.D$.*

Definition 5 Location (L). *The location represents a cartesian notation indicating the location of an entity E with respect to the CPS in each proxemic zone; it is denoted as a point in the plane: $E.L = (x_i, y_i)$.*

Definition 6 Movement ($\vec{M}$). *The movement represents a vector that has magnitude (or length) and a direction of an entity E with respect to the CPS, denoted as $E.\vec{M} = (l, d)$; where l is the length and d is the direction, such as d ∈ {-1, 0, +1}:*

- *if $d = 1$ represents that E is moving towards the CPS;*
- *if $d = 0$ represents that E is static;*
- *if $d = -1$ represents that E is moving in the opposite direction with respect to the CPS.*

Definition 7 Orientation (O). *The orientation represents the face angle of an entity E with respect to the face of CPS, and it is denoted as $E.O$.*

Entities delimit proxemic zones according to the distance D, with respect to the *CPS*. Def. 8 presents the formal meaning of a proxemic zone.

Definition 8 Proxemic Zone (P_Z). *A proxemic zone represents a circular zone delimited by a maximum distance (radio) with respect to the CPS. There are four proxemic zones defined according to such maximum distance: $P\_Z_{intimate}$, $P\_Z_{personal}$, P_Z_{social}, and P_Z_{public}. The proxemic zone in which an entity is located is denoted as: $E.P\_Z$ or $I.P\_Z$, accordingly. Thus,*

- $E.P\_Z = P\_Z_{intimate}\ if\ 0 < E.D \leq MAX\_ID$;
- $E.P\_Z = P\_Z_{personal}\ if\ MAX\_ID < E.D \leq MAX\_PD$;
- $E.P\_Z = P\_Z_{social}\ if\ MAX\_PD < E.D \leq MAX\_SD$;
- $E.P\_Z = P\_Z_{public}\ if\ MAX\_SD < E.D \leq MAX\_PubD$.

where MAX_ID, MAX_PD, MAX_SD, and MAX_PudD represent the maximum distances that define the intimate, private, social, and public zones, respectively. They are parameters that have to be provided by users or developers.

A proxemic environment, as Def. 9 describes, is then conformed by the set of entities that interact with the target entity (*CPS*), according to DILMO dimensions and proxemic zones (P_Z).

Definition 9 Proxemic Environment (*P_E*). *A proxemic environment, denoted as P_E, represents a set of sensors and devices attached to entities (E) that can in turn interact according to D, I, L, M, and O. It is denoted as a four-tuple containing a target entity (CPS), a set of interaction objects, general or identifiable, and the distances that define the proxemic zones, such as*

$P\_E = < CPS,\ entities,\ distances >$

$$P\_E = < CPS,\ entities = \{E_1, ..., E_n | E_i \text{ is an Entity (Def. 1)}\},\ identities = \{I_1, I_2, ..., I_m | I_j \text{ is an Identity (Def. 3)}\},\ distances = \{MAX\_ID,\ MAX\_PD,\ MAX\_SD,\ MAX\_PubD\}(Def.\ 8) >$$

A *P_E* represents the system based on proxemic behaviors. The change of DILMO measures over the time of *E* in a *P_E*, defines their behaviors (see Def. 10).

Definition 10 Behavior (*B*). *A behavior represents the change of D, L, M, and O measures or the P_Z of an entity E , starting from its initial behaviour (B_0). It is denoted as a tuple $E.B_i =< E.D_i, E.L_i, E.M_i, E.O_i, E.P\_Z_i >$ and is produced by a transition from the previous behaviour B_{i-1}, such that:*

- $E.B_0 =< E.D_0, E.L_0, E.M_0, E.O_0, E.P\_Z_0 >$ *(initial behaviour);*
- $E.B_i = \nabla E.B_{i-1} =< E.D_i, E.L_i, E.M_i, E.O_i, E.P\_Z_i >$ *(for $i \geq 1$);*

  where ∇ is a user-defined function to detect the change of measures of DLMO dimensions.

Behaviors of entities might be defined by developers through user-defined ∇ functions. For example, a ∇ function to detect movement of an entity, can be implemented based on readings from an accelerometer sensor of a smartphone. Actually, when an entity's movement is detected, the rest of DLO measures can be affected. These behaviors should be specified by developers. Hence, according to behaviors of entities *E* in a *P_E*, they can react by performing actions to comply their functionalities. It is expected that the *CPS* in a *P_E* is the interaction object that should execute most actions in a *P_E*; however, it is also possible to define actions for other entities *E* in the *P_E*. Actions define the specific functionalities of proxemic applications and must be defined by developers which are not included in the DSL.

To define a *P_E* the following **initial conditions** are specified:

1. Distances that define the four *P_Z* are provided by designers/developers according to user requirements.

2. In a *P_E* exists one and only one target entity;
   thus, given a $P\_E =< CPS, entities, distances >$, *CPS* is not null.

3. The *CPS* represents the target interaction object, with the following properties:

   - The original location of *CPS* is denoted as $CPS.L = (0,0)$ based on Cartesian coordinate system, since proxemic zones and DILMO dimensions of all other interaction objects are determined with respect to the *CPS*. This DSL allows the designer to describes a two-dimensional space.
   - It has a face from which its visual field is defined according to the following angles: $\measuredangle_{MinAofV}$ and $\measuredangle_{MaxAofV}$. If $\measuredangle_{MinAofV}$=$\measuredangle_{MaxAofV}$=NULL, The *CPS* can not sense the entities' orientation. The Field of view (FoV) can be between the 0° and 360°

4. For all entities (E) in the system, DILMO dimensions and proxemic zones can be assigned as their initial behavior. Formally:

   $\forall E_i \in P\_E.entities$

   $E_i.B_0 =< D_0, L_0, M_0, O_0 >$;

   and L_0, M_0, O_0 can be $null$.

   It means that distance (D) is the only mandatory dimension for entities. Orientation is a dimension that only matters if the *CPS* has a view angle that shows the space where other entities can be detected (examples of such *CPS* are a screen, a smartphone using its camera. When the movement is equal to null the entity is static in the environment.

These definitions conform the formal basis of our DSL and are used to specify **conditions in a** *P_E* (behaviors – *B*) to which entities should react by executing *Actions*. Hence, from the initial conditions, proxemic *Actions* can be defined with respect to the *P_Z* and DILMO dimensions of entities in the *P_E*.

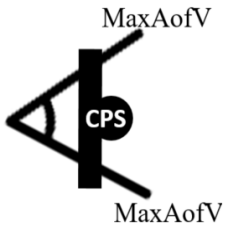
Next, we describe each graphic notation symbol that allows non-specialist designers to express proxemic interactions for modelling proxemic environments and supports the development process.

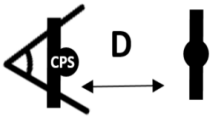
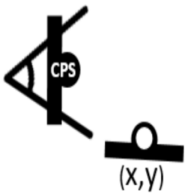
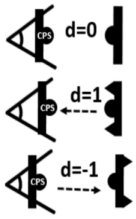
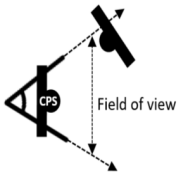
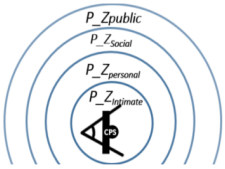
5.3 Graphical Notation of the graphical DSL

The graphical notation provided by our graphical DSL, represents in a visual way, all components in a proxemic environment; thus, it corresponds to the formal definitions presented in Section 5.2. Our graphical model is inspired on the notation system proposed by Edward T. Hall in [45], which provides an axis notation code that allows the representation of human relations. We proposed graphical symbols to represent a *P_E* conformed by the *P_Z*, entities,

and their DILMO dimensions. Each symbol can be described as entity in the *P_E*, which has several behaviors. Our graphical model is a standard graphical language that allows non-specialist designers to describe the proxemic interaction among physical user and devices using DILMO. The graphical language has been proposed to help developers to understand and implement the *P_E* established by designer. Table 5.1 illustrates each symbols from the corresponding formal notation described in Section 5.2.

Table 5.1 Graphical notation for proxemic environment.

	Physical System *CPS*: represents the target entity, from which a proxemic environment is defined in **Definition 2**. The *CPS* has a field of view angle that shows the space where other entities can be detected by the camera (examples of such *CPS* are a screen, a smartphone using its camera). If the angle between MaxFofV and MinFofV is equal to zero, the *CPS* cannot obtain entities in front to it or it can detect entities all around it (an example of such *CPS* is a smartphone detecting beacons near to it). The *CPS* is an entity which has relationship cardinality 1:*N* between the *CPS* and *E* or *I* that are involved in a *P_E*. 1:*N* refers to a one-to-many relationship one *CPS* the other side related to *E*.
	The entity *E* is denoted as white and black symbol, that represents a role in the *P_E* without a unique identification, for example a user, a person, a visitor or client. It is defined in Definition 1. The front face of *E* can be recognized and the orientation angle can be obtained from the *CPS*. The Entity *E* face is represented as white circle in one side of vertical line.
	The identity *I* is denoted as black, that represents a unique identification (e.g., "John", id number). It is defined in **Definition 3**. The front face of *I* can be recognized and the orientation angle can be obtained from the *CPS*. The identity *I* face is represented as semi-circle in one side of vertical line.
	The entity *E* is denoted as white and black symbol without face which is represented as a white circle in the middle of a vertical line. This symbol can be used to describe the proximity between entities without considering the identity and orientation.

Continuation of Table 5.1	
	The identity I is denoted as black without face which is represented as circle in the middle of a vertical line. This symbol can be used to represent proximity between identified entities.
	The physical distance between E and CPS is represented as $E.D$ according to **Definition 4**. This behaviour can be illustrated using a combination of symbols, such as CPS and E, in which is not necessary the face orientation. The P_E can has multiple entities with different distances from the CPS.
	The location of E with respect to the CPS is represented as $E.L = (x_i, y_i)$ according to **Definition 5**. The location in a P_E can be illustrated describing below of the interaction object symbol E the cartesian notation $E.L = (x_i, y_i)$ to each entity in the P_E.
	The movement of an E identified with respect to the CPS, is denoted as $E.\vec{M} = (l, d)$ according to **Definition 6**. The arrow orientation on the symbols describes the direction of entities in a P_E. The symbology illustrates the two-movement of E with respect to the CPS and static case of E.
	The orientation of an E identified with respect to the CPS, is denoted as $E.O$ according to **Definition 7**. The projection of vectors on the symbol illustrates the CPS field of view to detect an E in a P_E. The symbology allows the designer to represent more than one E in the field of view.
	CPS Proxemic zones P_Z are denoted as $P\_Z_{intimate}$, $P\_Z_{personal}$, P_Z_{social}, and P_Z_{public} according to **Definition 8**. The circle around the CPS describes each (P_Z) in a P_E. Only one CPS must be presented in a P_E for designing a P_E. The designer must provide the circle's sizes according to the requirement of the scenario.

5.3.1 Tangible interface

We made a tangible interface based on the symbols proposed in order to experiment with our graphical DSL. This interface allowed us to represent different proxemic behaviours with

a standard interface using the symbology above mentioned in Section 5.3 . The tangible interface was a first experience to prove the feasibility of graphical DSL. We built the interface using office supplies such as paper click, adhesive tape, and white paper. Our aim was to use a low-tech that allow us to describe proxemic behavior before writing code. We made each entity that was described in Table 5.1. The use of paper prototypes that allow users to express their understanding of a software application before implementation [80]. Next, we show some examples of proxemic environments that were simulated with our tangible interface (paper prototype). Figure 5.1 illustrates four proxemics scenarios that were modeled by our research team using the tangible interface:

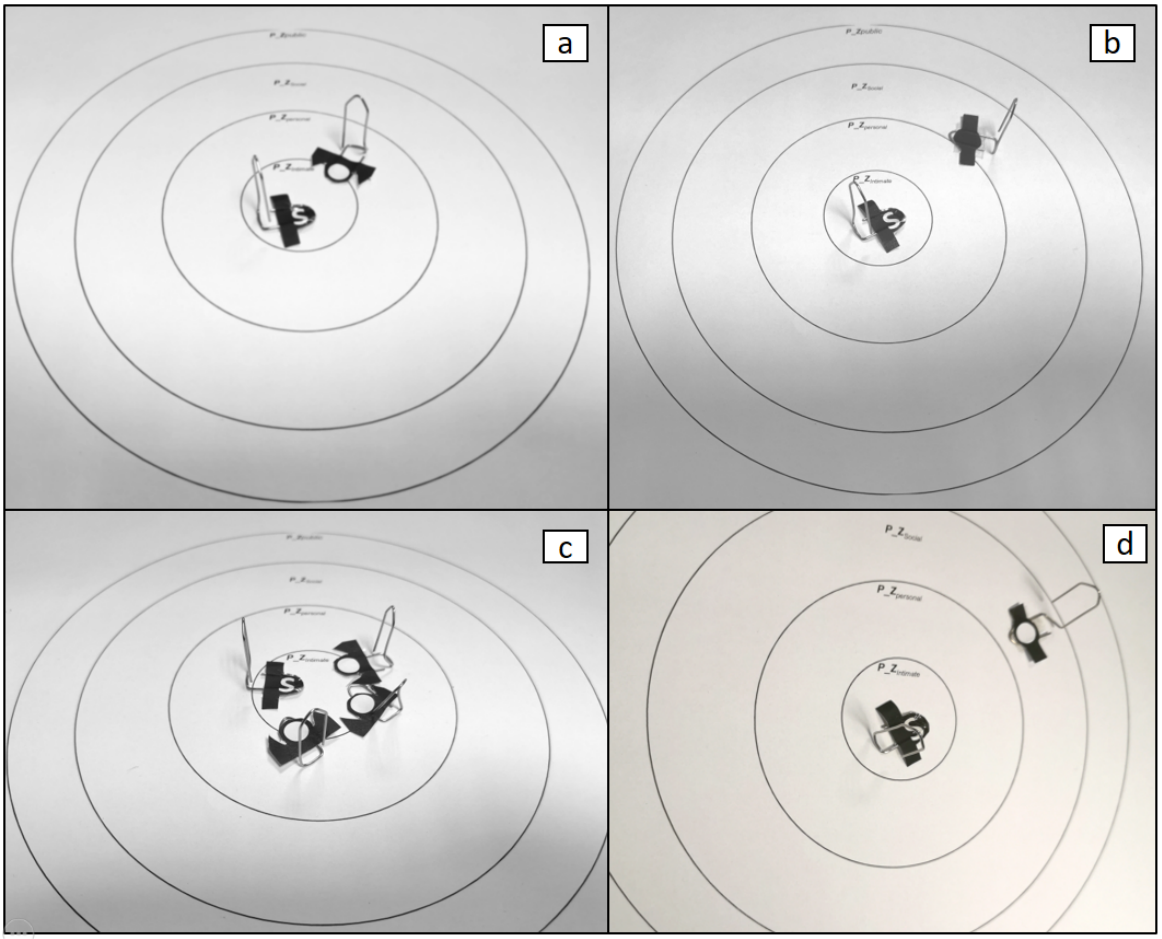


Fig. 5.1 Proxemics scenarios modelled with tangible interface

Scenario (a). In this case, we wanted to represent somebody who goes toward the *CPS* in which their physical distance are obtained based on face detection. The entity (*E*) is moving toward the *CPS*'s $P\_Z_{intimate}$ and the entity is oriented to *CPS*. The symbol describes an entity in which the orientation can be detected by the *CPS*. This

example shows a subset of proxemic dimensions such as distance, movement, and orientation – i.e., a **DMO** proxemic environment. This case can represent an interactive display that works as *CPS* and any user who acts as an entity. Figure 5.1 (a) shows the symbol that was chosen for representing the entity's front. When the user (E) is walking straight towards *CPS*, the *CPS* can trigger an event according to proxemic **DMO** dimensions' values.

Scenario (b). This case allowed us to represent a specific person or object that is stopped within the proxemic zone of the *CPS*. An identity (I) without face is static in the *CPS*'s P_Z_{social}. The symbol illustrates a subset of proxemic dimensions, such as **DI**. This case represents a smart environment in which a person carries a mobile device such as a mobile phone, smartwatch, smart card and, electronic bracelet, which can be identified. The mobile devices can be detected by the *CPS* using radio frequency signals to estimate the physical distance between a device and the *CPS*. Figure 5.1 (b) shows the symbol that was chosen for representing the entity without face.

Scenario (c). Scenario (a). textcolorblackThe third case shows three entities the are moving toward to *CPS*'s $P\_Z_{intimate}$ and the entities are oriented to the *CPS*. The *CPS* detects entities according to the *CPS* field of view. The symbols illustrate entities that implement a subset of proxemic dimensions, such as distance, movement and orientation – i.e., a **DMO** proxemic environment. The *CPS* can detected the entities in front it. With this case, the designer described graphically multiple entities that can interact with the *CPS*. The *CPS* can generate a specific event according to each entity detected in the simulated environment.

Scenario (d). The last case describes an entity (E) without face that is static in the *CPS*'s P_Z_{social}. The example allows designer to illustrate the interaction between the *CPS* and an entity based in physical distance between the *CPS* and E. This case represents a proximity marketing scenario based on beacon BLE technology that sends signals to any smart device nearby. The proximity targeting supports restaurants with interactions with their customers from a distance, including order-ahead and informing customers about the variety of dishes on the menu.

The symbols provides arrows that allow designers to describe the direction of movement of an entity with front or without front. When an entity without a face moving toward or moving away from the *CPS*'s $P\_Z_{personal}$. This first experience of modelling proxemic environments with tangible interaction allowed us to refine the symbology to simulate proxemic behaviour based on DILMO dimensions. For example, the movement sense of

entities was made by adding arrows to the entities. The fill color was used to identification of identity. The position of a semi-circle on the entity for describing the entity's orientation. This experience also permitted us to realize that we need to find a data structure or a markup language for gathering proxemic information in a file from the initial design.

5.4 Modelling Proxemic Environments: A Systematic Process

This section presents a general modelling process based on our graphical DSL. The DSL allows designers to represent proxemic environments and proxemic behaviours from initial conditions based on DILMO dimensions. The DSL is intended to provide standardization and a common language to model proxemic behaviours of each entity in a proxemic environment. The developers can generate an XML file from the blueprint previously created in the design phase.

We describe a systematic process to design proxemic environments based on our DSL. Our graphical DSL provides a symbology based on formal notations and graphical objects representing all components in a proxemic environment, i.e., entities, DILMO dimensions, proxemic behaviors, etc. During the design phase, the proxemic environment and the respective proxemic behaviors are graphically modelled. We described all symbols in Table 5.1. Our graphical DSL constitutes a tool that supports design phase of proxemic applications. Besides that, it establishes a systematic developing process consisting of the following stages: (i) Graphical modelling; (ii) XML generation based on XML Schema; and (iii) proxemic behaviour simulation. We designed XSD that allowed us to offer the developers data structure for supporting the XML generation. The XSD was created by the World Wide Web Consortium (W3C) to describe XML documents' content and structure in an XSD[1]. Below, we describe each stage that compose the modelling of proxemic environments. Figure 5.2 illustrates the systematic process for modelling proxemic environments.

5.4.1 Graphical modelling

At this stage, the designer creates a *P_E* using the symbols that were described in table 5.1. Our notation provides a set of symbols that enables designers to represent *P_E* in a graphical model. The designer must select the symbols to use for designing the *P_E*. To design a graphical proxemic environment the designers must be kept in mind that only one *CPS* must be created for a proxemic environment. The designer can add multiple

[1]https://www.w3.org/TR/xmlschema11-1/

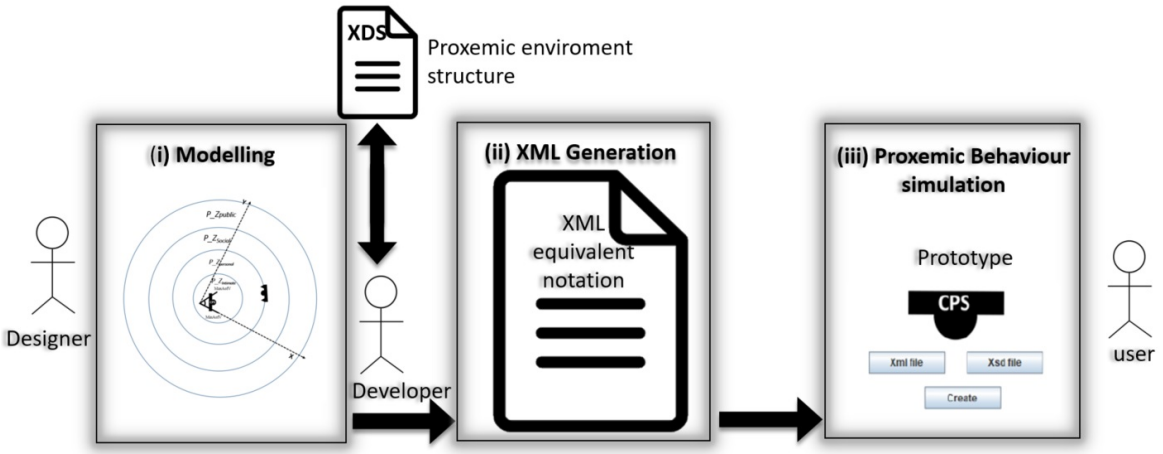


Fig. 5.2 Systematic process

entities to the proxemic environment; however, the maximum limit of entities that a *CPS* can detect is based on hardware capability. For example, if the *CPS* use a camera for sensing entities the maximum number of faces that can be detected is based on the camera's field of view. The *CPS* may use different sensors to detect entities according to the hardware available for using the systems. The designer can use any graphic design software to create a blueprint for the *P_E* and represent the entity's behaviour using the graphical notation aforementioned. The graphical notation language is focused on the design of proxemic applications by using a common language for non-specialist developers. The user has a variety of symbols that represent the entities as shown in Table 5.1. In Section 5.5, we will present a scenario, which describes one example showing the sequence of steps for simulation proxemic behaviours in *P_E*. This stage described the process that allow designers and end-users without programming knowledge to design proxemic applications for smart environments. In the next Section 5.4.2, we describe how the developer can convert the designer's blueprint to an XML file from the designer's initial design.

5.4.2 XML generation

The XML file is manually generated by the developer, who converts the blueprint according to its design. From the formal specification and the graphical design of a proxemic environment, it is possible to generate an XML file to keep the *P_E*: initial conditions. The proxemic information related to the initial conditions of the *P_E*, as well as the proxemic behaviour and actions are stored in an XML file based on the XML schema.

With the aim to guide the developer for converting the graphical modelling into an XML file, we designed XSD to describe multiples entities that interact with only one *CPS* in the *P_E*. This XSD enables to present the graphical modelling as a list of elements and attributes (data structure) which the computer systems can interpret. We describe each element of *P_E* with graphical view XSD that allowed us to present an abstract representation of the *P_E*, showing the XSD structure without syntactic confusion. Figure 5.3 shows a *P_E* that has been structured with graphical view XSD. The XSD graphical view structure provided us a logical view of *P_E* using XML Scheme. The XSD defines the shape or structure of an XML file that contains each component of *P_E*. The XSD file contains a syntax that allows the description of each element of the *P_E*. We implemented XSD Editor [2] eclipse IDE plug-in to create XSD graph view and source code.

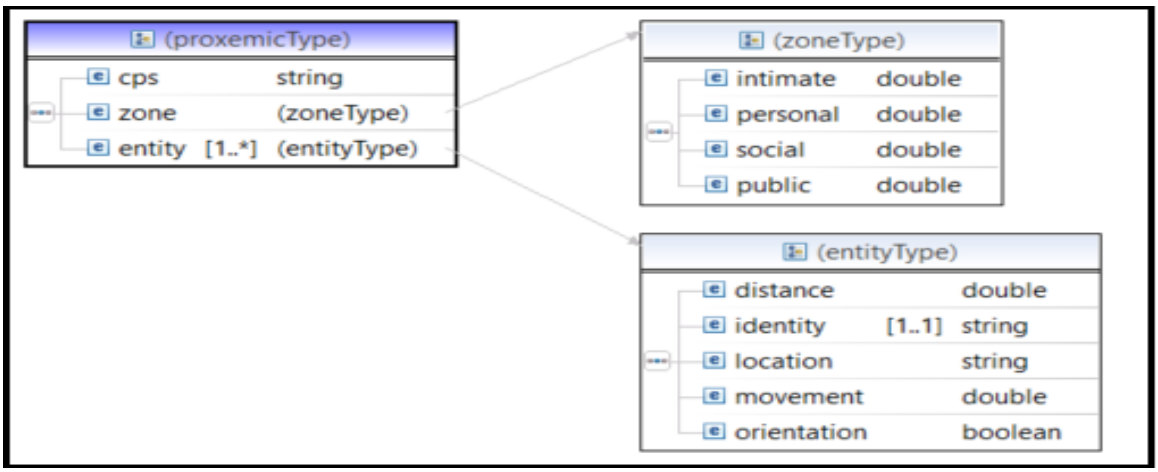


Fig. 5.3 Graphical XML Schema (XSD)

Below, we show XML Schema source code describing each component *P_E* (see Figure 5.4).

Figure 5.4 (a) shows XSD complexType element that allows developer to describe the attributes of a CPS such as *P_Z*: $P\_Z_{intimate}$, $P\_Z_{personal}$, P_Z_{social}, and P_Z_{public}. Figure 5.4 (b) shows XSD complexType elements that allow developer to describe the attribute type of one entity (*E*) or a list of entities such as DILMO proxemic dimensions.

Next, we show an example of equivalent XML code that describe a graphic modelling with one entity identified in the *P_E* (see Figure 5.5).

(**a**) This element allows developers to register the physical distance between an entity identified and *CPS*, defined by the designer.

(**b**) This element allows developers to register an entity identified (*I*) that is recognized by the *CPS*, (for example using face recognition).

[2]https://wiki.eclipse.org/Introduction_to_the_XSD_Editor

Fig. 5.4 XSD complexType element source code; (a) CPS XSD source code; (b) List of XSD source code.

(**c**) This element describes a single point in 2D two-dimensional systems relative to the *CPS*. The *CPS* has the origin coordinates (0,0).

(**d**) This element allows the developer to describe the movement direction of an entity.

(**e**) The element allows the developers to describe if an entity is oriented towards the *CPS*. This example illustrates an identity that is oriented towards the *CPS* (true).

All values are obtained from initial conditions established by the designer. However, these vales can be updated according to the technologies used to sensing the proxemic information. Figure 5.6 shows the *P_E* scheme source code proposed that guide developer to manually generate the XML file. This structure is equivalent to graphical model in a way that computer can process the proxemic information. In Section 5.5, we show the equivalent XML file that is generated from graphical modelling an formal notation.

This process can be improved in the future in order to create a file automatically using a software application. One of our aims in this phase of research was to find a data structure that would enable developers to register proxemic information in the an XML file.

With the purpose of minimize the complexity of steps that developers need to read XML file we implemented JAXB class generator. The JAXB [74] technology simplifies access to XML documents that describes the *P_E*. This tool establishes mappings between Java classes and the XML schema. We implemented the JAXB Builder eclipse plugin. This plugin allows the developer to process proxemic information from an XML file, including a JAXB project wizard for generating the classes. Figure 5.7 illustrates a UML class diagram that describes the structure and relationship among entities and proxemic zones, represented in

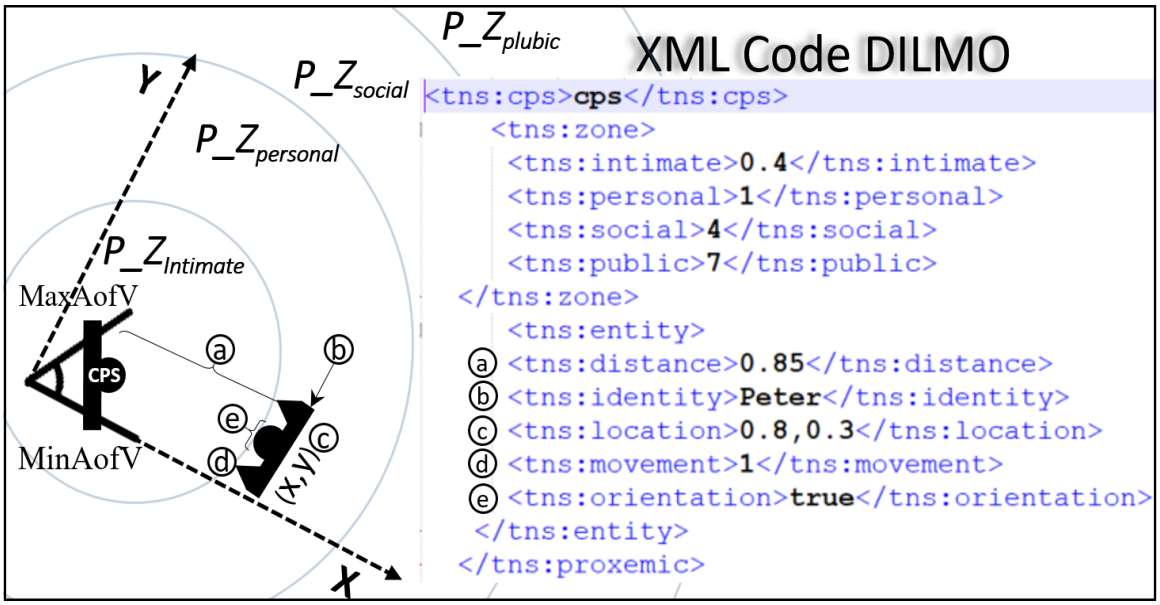


Fig. 5.5 XML Code from graphical view

the *P_E*. The XML scheme can also be used by the developer to automatically generate Java classes to gather and record proxemic information during the development phase.

5.4.3 Proxemic behaviour simulation

Once the *P_E* is defined and modeled, it is possible to simulate initial proxemic behavior according to the considered DILMO dimensions to simulate the proxemic actions of the *CPS* and entities. In the implementation phase, developers decide how DILMO dimensions are captured according to the available sensor technology (e.g., BLE, depth camera, kinect sensor, smart mobile capabilities). Also, the specific actions of the *CPS* are implemented according to user requirements (e.g., play a song, display a video, show publicity, transmit information, face recognition).

The interfaces and classes generated in the previous stage, can be integrated, included, and used in the developing of a specific proxemic application. We developed a prototype of our DSL able to support the specification of proxemic environments: entities, proxemic zones, DILMO combinations, and conditions to react; from which the corresponding XML file. The XML file represents the *P_E* that is manually generated from the initial conditions by the developer. This file contains proxemic information of each entity which is uploaded using the prototype's graphical user interface (GUI). The prototype's GUI elements are explained on Figure 5.8.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.example.org/DilmoXMLSchema004"
 xmlns:tns="http://www.example.org/DilmoXMLSchema004"
 elementFormDefault="qualified">
<xs:element  name="proxemic">
    <xs:complexType>
          <xs:sequence>
          <xs:element name="cps" type="xs:string"></xs:element>
          <xs:element name="zone" >
            <xs:complexType>
              <xs:sequence>
                <xs:element name="intimate" type="xs:double"/>
                <xs:element name="personal" type="xs:double"/>
                <xs:element name="social" type="xs:double"/>
                <xs:element name="public" type="xs:double"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        <xs:element name="entity" maxOccurs="unbounded">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="distance" type="xs:double"/>
                <xs:element name="identity" type="xs:string" minOccurs="1"/>
                <xs:element name="location" type="xs:string"/>
                <xs:element name="movement" type="xs:double"/>
                <xs:element name="orientation" type="xs:boolean"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Fig. 5.6 *P_E* XML Schema (XDS)

The GUI allows user to update the entities' proxemic behaviour such as: (i) update the distances; (ii) adjust the DILMO dimensions in the frame that represents each entity condition (in the grey zone) ; (iii) specify that an action is performed by the *CPS* (by clicking on the Action button). Also other actions can be generated from different behaviors; thus, it is possible to simulate multiple proxemic interactions in the same *P_E*. The initial conditions that can be updated using an input range slider to simulate the different distance between an entity and the CPS. The prototype allows user to view the corresponding interaction code based on the entity's behavior and a proxemics zone in which an entity is encased (by clicking on the Action button). The developer can identify any proxemic behavior of entities that were established by the designer in a specific proxemic zone. In this prototype, orientation of

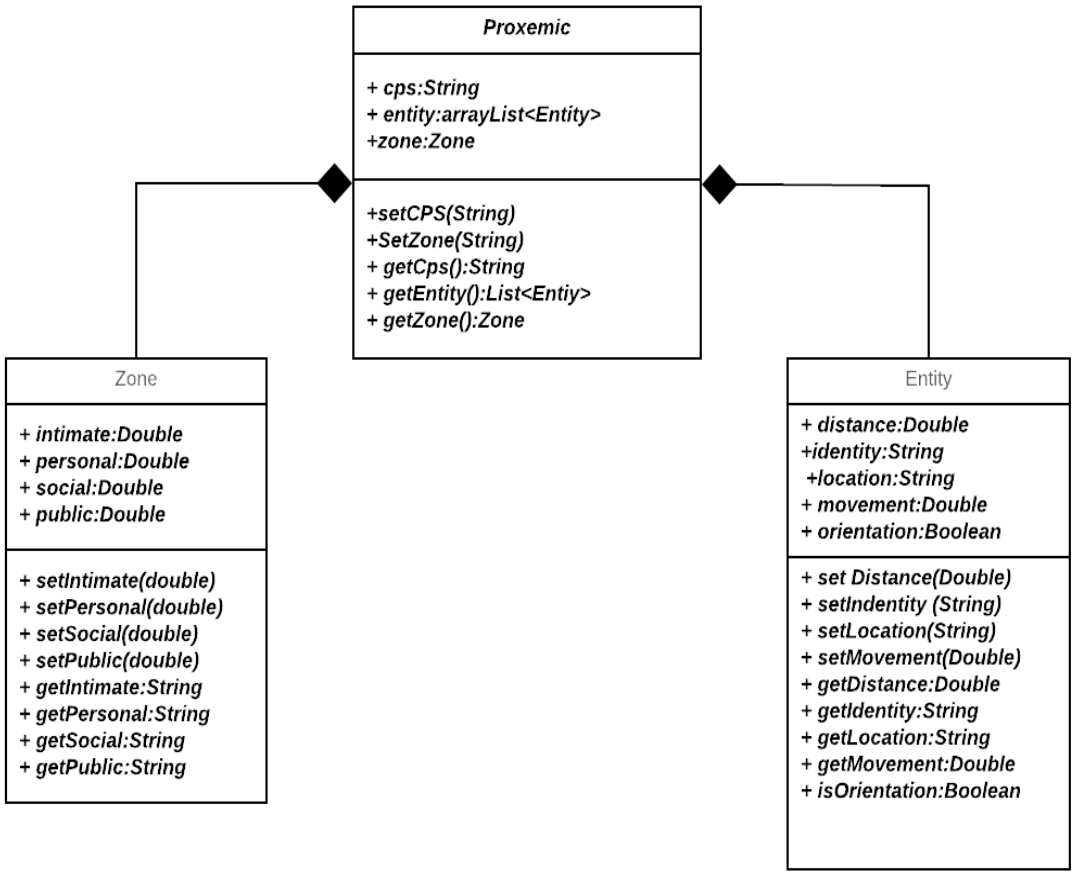


Fig. 5.7 UML class diagram derived from the XSD

entities are represented as boolean parameter indicating if the entity is in the field of vision of the *CPS* or if the entity is outside of field of view the *CPS*.

Furthermore, the graphical notation presented in table 5.1 could be implemented as a graphical interface for a drag and drop facility to use the tool. Developers will be able to assign their user-defined functions to actions; and technology to gather DILMO dimensions could be specified.

To demonstrate the suitability and functionality of our proposed graphical DSL, we have implemented a scenario to show the design of proxemic behaviors in a proxemic environment, with their respective generated XML file. In the following, we describe a systematic process that allow designer modelling a proxemic environment.

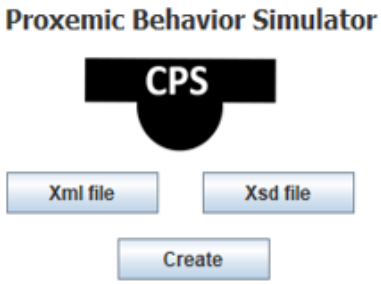
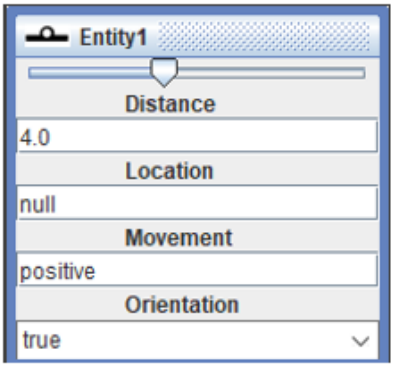
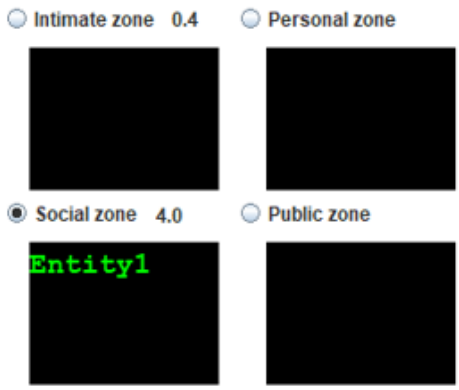
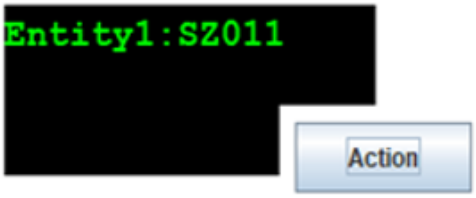
GUI	DESCRIPTION
	The set of buttons allow to simulate a P_Z from initial design. The XML file button allows users to upload XML file that has proxemic information of P_Z . The XSD file allows user to validate XML file structure using XSD scheme. The button Create generates the frames according to entities number in the P_Z.
	The frame symbolizes an entity in P_Z which contains four input text fields and an input range slider. The input text fields show the DILMO measures from initial conditions that can be updated by the user using an input range slider to simulate the different distance between an entity and CPS. The entity's orientation to CPS can be updated with the select button.
	The entities are shown according to the distance to CPS which are arranged based on the CPS's proxemic zones (represented by the black squares), a proxemic zone can have one or more entities.
	The Action button allows the user to see the corresponding interaction code based on both an entity's behavior and a proxemics zone in which entity is encased. The user must select at least one proxemic zone.

Fig. 5.8 Description of GUI Prototype

5.5 Graphical modelling of Proxemic scenario: an illustrative example

To illustrate a simulation of proxemic behaviours using our DSL we describe an *P_E* scenario that simulates activities of daily living, in which people interact with smart environments where I_1 and I_2, move towards the *CPS* into the personal zone ($P\_Z_{personal}$) and face the *CPS*; and the entity E_1, who stands in front the *CPS* in the social zone (P_Z_{social}), is detected; thus different actions of the *CPS* can be accordingly defined. With the purpose of modelling of proxemic scenario, we follow the systematic process: (i) Graphical modelling, (ii) XML generation and, (iii) proxemic behaviour simulation.

- **Graphical Modelling**: we designed graphical representation of this scenario and the initial conditions. Figure 5.9 shows the design that emerged from symbology proposed. Representing entities (*E*) or identities (*I*) in a *P_E*, depends on the nature of actions: if they are generic (i.e., same action for a specific behaviour of any entity) or if they should be performed as a response to of a specific behaviour of a specific identified entity (i.e., an identity). For example, in a *P_E* where a digital screen (*CPS*) shows advertisement messages (*Action*), when people is approaching it (*Behavior*), people can be represented as entities (*E*). Meanwhile, in a *P_E* where an automatic door (*CPS*) allows entering the garage (*Action*), when a specific car plate arrives (*Behavior*), the car should be modelled as an identity (*I*).

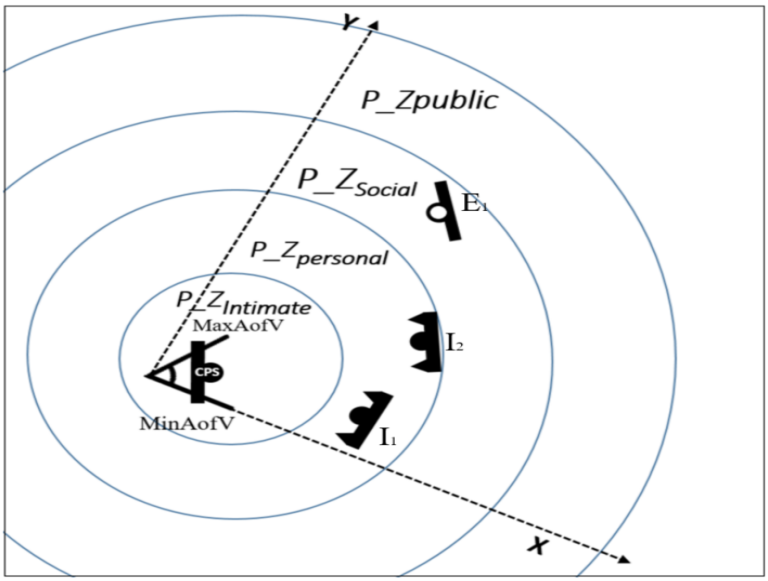


Fig. 5.9 Graphical model for scenario

Formal notion to describe the initial conditions of *P_E* that was modelled in Figure 5.9:

- The four tuple of *P_E*: $P\_E =< CPS, entities = \{E_1\},\ identities = \{I_1, I_2\},\ distances = \{0.4, 1, 4, 7\} >$;
- $CPS.L = (0,0)$;
  $\measuredangle_{MinAofV} = 0^\circ$ and $\measuredangle_{MaxAofV} = 60^\circ$;
- Distance of entities: $I_1.D = 0.85$, $I_2.D = 0.80$, $E_1.D = 3$;
  Location of $I_1.L = (0.8, 0.3)$;
  Movement of I_1 and I_2 (towards *CPS*): $I_1.\vec{M} = (l_1, 1)$ and $I_2.\vec{M} = (l_2, 1)$;
  Movement of E_1 (static):$E_1.\vec{M} = (l_3, 0)$;
  Orientation of entities (faces of entities are in the area of vision of the *CPS*): $I_1.O = 20^\circ$, $I_2.O = 35^\circ$, $E_1.O = 75^\circ$, thus $\measuredangle_{MinAofV} \leq I_1.O, E_1.O \leq \measuredangle_{MaxAofV}$;
- Proxemic zones of entities:
  $I_1.P\_Z$, $I_2.P\_Z$=$P\_Z_{personal}$, $E_1.P\_Z$=P_Z_{social};
- Thus, initial behaviour of entities are:
  $I_1.B_0 =< 0.85, (0.8, 0.3), (l_1, 1), 20^\circ, P\_Z_{personal} >$
  $I_2.B_0 =< 0.8, NULL, (l_2, 1), 35^\circ, P\_Z_{personal} >$
  $E_1.B_0 =< 3, NULL, (l_3, 0), 75^\circ, P\_Z_{social} >$

From these initial conditions, the *CPS* can react with $Action_1$ and $Action_2$, that can be denoted in several ways, for example: $if\ \exists\ E$ in P_Z_{social}, then $CPS.Action_1$.

$\forall\ I_j \in P\_E.identities \wedge I_j$ in $P\_Z_{personal}$, then $CPS.Action_2$.

- **XML generation:** we converted the design that was shown in Figure 5.9 to an XML file. The XML scheme allows developers implementing the right structure for the scenario described. Figure 5.10 describes the XML equivalent to the design that was modeled in design phase. The entities' behavior are formally modelled during graphical modelling. This file has been composed of the *CPS* and the set of entities according to the designer's graphical modeling. According to the graphical modelling, the *CPS* proxemic zones and initial conditions of entities have been recorded in the XML file. The prototype reads the XML based on the initial conditions that were fixed in the design phase. In the development phase, the XML file will be updated according to the system's events. This XML file presents an approach for modelling proxemic environment based on a graphical DSL focusing on entities in the *P_E*.

```
<?xml version="1.0"  encoding="UTF-8"?>
<tns:proxemic  xmlns:tns="http://www.example.org/DilmoXMLSchema004"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://www.jmdoudoux.com/test/jaxb biblio.xsd  ">
<tns:cps>cps</tns:cps>
    <tns:zone>
     <tns:intimate>0.4</tns:intimate>
     <tns:personal>1</tns:personal>
     <tns:social>4</tns:social>
     <tns:public>7</tns:public>
  </tns:zone>
     <tns:entity>
     <tns:distance>0.85</tns:distance>
     <tns:identity>Peter</tns:identity>
     <tns:location>0.8,0.3</tns:location>
     <tns:movement>1</tns:movement>
     <tns:orientation>true</tns:orientation>
   </tns:entity>
    <tns:entity>
     <tns:distance>0.8</tns:distance>
     <tns:identity>Ana</tns:identity>
     <tns:location>null</tns:location>
     <tns:movement>1</tns:movement>
     <tns:orientation>true</tns:orientation>
   </tns:entity>
    <tns:entity>
     <tns:distance>3</tns:distance>
     <tns:identity>Entity</tns:identity>
     <tns:location>null</tns:location>
     <tns:movement>0</tns:movement>
     <tns:orientation>false</tns:orientation>
   </tns:entity>
  </tns:proxemic>
```

Fig. 5.10 XML file for scenario.

- **Proxemic behaviour simulation:** we implemented the scenario previously described in Section 5.5. We show how the prototype can support the developer's implementation process from the design phase. Figure 5.11 shows the representation of the initial conditions of this **Scenario** with our prototype. The scenario shows how the prototype can support the developer's implementation process from design. When the XML files are uploaded, and the user makes a click on Create button the prototype will show the respective *P_E* and entities in the grey zone of the prototype. Once the *P_E* has been created through prototype the user can see the initial conditions which can be updated using the interface graphic. For this scenario, the user has selected a personal zone using a radio button. When the user click on the "Action button," the black panel shows the interaction code of each entity in the *P_E*.

We have implemented a prototype that helps end-user to describe proxemic behaviours in different proxemic environments based on our DSL proposed. Nevertheless, new challenges arise when modelling proxemic environments that we describe in the next Section 5.6 .

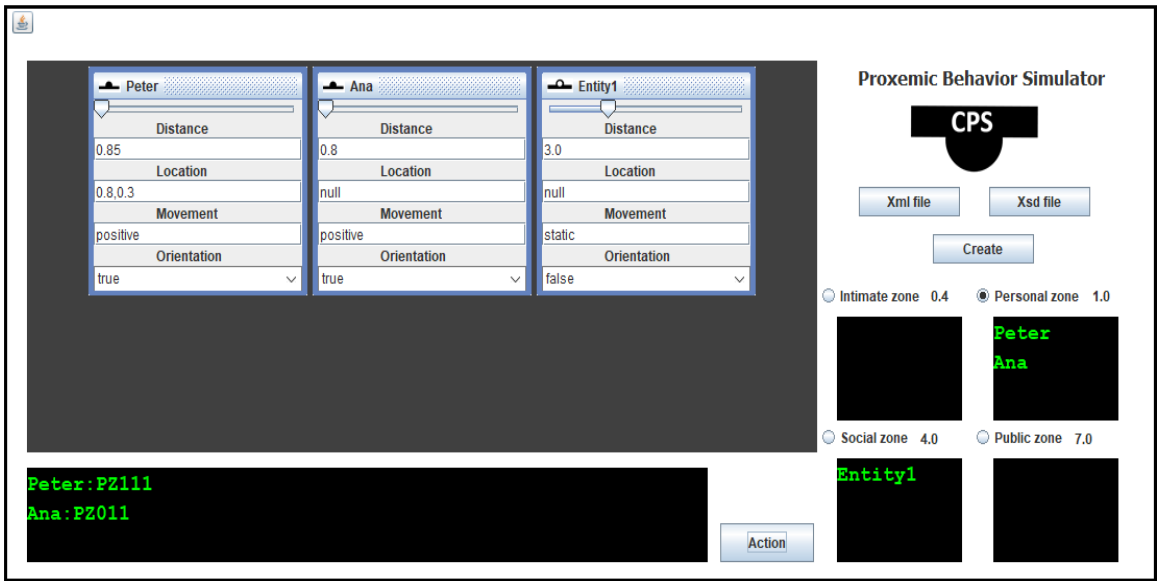


Fig. 5.11 Scenario modelled with our DSL

5.6 Limitations of the Proposed Graphical DSL

We proposed an approach for modelling proxemic environments based on a DSL. Our proposed graphical DSL provides an symbology based on formal notations and graphic objects representing components in a proxemic environment. Nevertheless, the current symbology does not include a fine-grained notation for representing gesture interaction. Adding gesture interaction to proxemic interaction can improve proxemic interactions in a specific context. A future version of our DSL could consist of an element that describes the kind of behaviour. We showed that our proposition allows a transition from the design phase to the implementation phase through our prototype. Notwithstanding, the graphical interface of the prototype presented in Section 5.4.3 could be implemented in a visual programming language, allowing designers to obtain a better perspective from proxemic environments.

5.7 Conclusion

In this chapter, we presented an approach for modelling a proxemic environment based on a graphical DSL. Our proposition is focused on entities in the proxemic environments, helping developers identify the trigger conditions from the design proposed. The DSL allows non-specialist designers to describe proxemic interactions for modelling proxemic environments and supports the development process. In order to prove its suitability of the proposition, we have implemented a prototype to simulate proxemic behaviours for proxemic environments.

We built XML scheme that allows the developer to convert a graphical modelled to a data structure. The XSD scheme file generated from the DSL allows developers to create classes in order to support development from initial design. We have implemented a prototype with the purpose to show the design of proxemic behaviors in proxemic environments, with their respective XML file. We plan to develop an integral framework for designing and implementing proxemic mobile applications, supported by this DSL. In the next chapter, we illustrate a framework that supports the implementation of functions to process the proxemic information in mobile devices.

Chapter 6

Proxemic Environments: A Framework for Developing Mobile Applications based on Proxemic Interactions

This chapter describes a framework that supports the development of applications based on proxemic interactions with mobile devices. Our proposal is a development approach to support the construction of mobile proxemic applications for smart proxemic environments *P_E* from DSL design, based on mobile technology and smart wearable technology. This framework proposes a concrete solution for developing mobile proxemic applications comprised of entities whose interactions are defined according to DILMO combinations. In the next Section 6.1, we describe the technology based on Android Operating Systems used for sensing proxemic information.

6.1 Sensing Proxemic Information with Mobile Devices

Progress in mobile technology has led to smartphone evolution with advanced connectivity features that have quickly made it a constant presence in our daily lives [27]. Most mobile devices are currently equipped with a wide range of hardware such as camera, motion sensor, GPS, wireless, and Bluetooth Low Energy (BLE) technology. Android Operations Systems provides native libraries (APKs) that allow the developer to obtain environmental conditions from the mobile devices associated with DILMO proxemic dimensions. Next, we describe technologies that are used for sensing proxemic information from mobile devices.

6.1.1 Mobile Computer Vision

Mobile computer vision is a powerful technology based on advanced image processing. The Android operating system provides libraries that allow developers to detect human face and some objects, such as bar-codes and QR-codes. Face detection is the process of automatically locating human faces in a image. A face that is detected is reported at a position with an associated size and orientation. When the face is detected, it can be searched for landmarks such as the eyes and nose.

Face Orientation

Android vision libraries provides methods that allow developers to detect human faces. Once a list of faces is detected the developer can gather information about each face and specific landmarks on these face. Android library provides the measurement of Euler Y and Euler Z (but not Euler X) for detected faces [29]. All coordinate values are absolute where image position (0, 0) represents the upper-left corner of the image. Figure 6.1 shows face's orientation to obtain Euler X, Euler Y, and Euler Z angles.

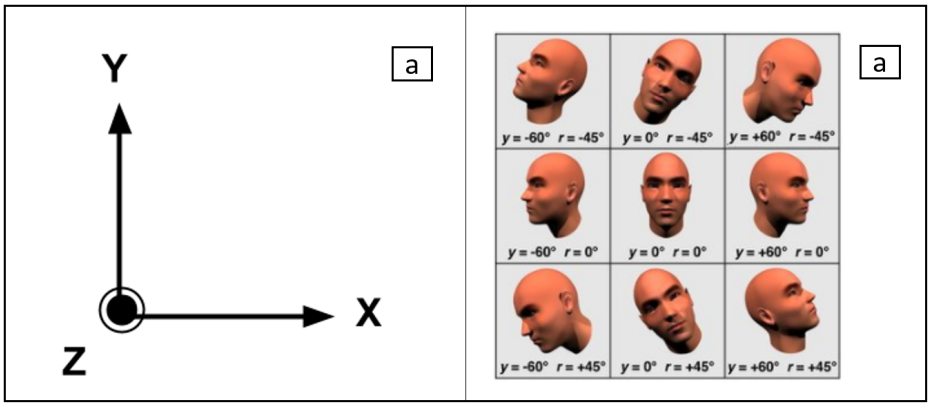


Fig. 6.1 Pose angle estimation: (a) The coordinate system with the image in the XY plane and the Z axis coming out of the figure. (b) Pose angle examples where yEuler Y, rEuler Z.

6.1.2 Motion and position Sensor

Motion sensors allow developers to measure physical movements of mobile devices, such as orientation, rotation, and acceleration. Android provides methods that let developers monitor the motion of a mobile device. At present, most Android-powered devices have an accelerometer, a gyroscope, and include magnetometer.

- **Accelerometer** determines the acceleration applied to the device, including the force of gravity in three-dimensional space. This sensor returns forces applied on the phone for x, y and z directions for the three coordinate axes in meters per second squared, (m/s^2) in the X, Y, and Z. Each axis is given separately. The force of gravity gives a value of 9.81 divided on all three axis, depending on how the phone is oriented. For example, if the phone lies on a table with the display pointing upwards, the force in z-axis is 9.81 while y and z is 0. Figure 6.2 shows the X, Y, and Z orientation axes relative to a typical Android mobile phone.

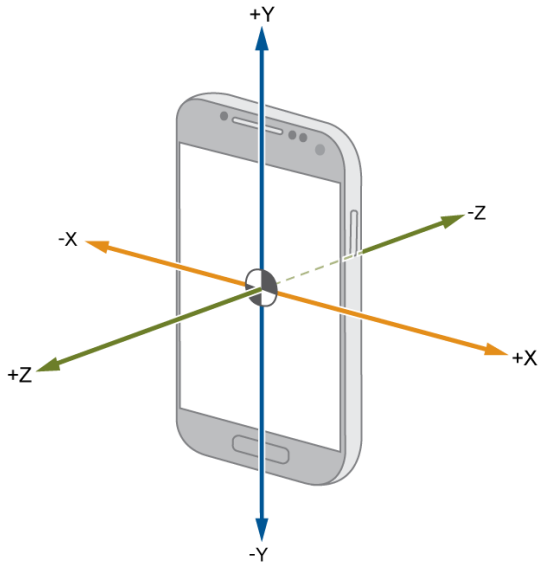


Fig. 6.2 Android accelerometer coordinate system

- **Gyroscope** measures the movement of the device, returning the rate of rotation in degrees per second (rad/s) around each device axis around (change of angles over the timestep). This sensor is usually used for games that explicit response to device movements. The coordinate system is equal to the one used for the acceleration sensor.

- **Magnetometer** utilizes the modern solid state technology that detects the Earth's magnetic field along three perpendicular axes X, Y, and Z. The sensor produces voltage which is proportional to the strength and polarity of the magnetic field along the axis each sensor is directed. The Magnetometer sensor reads the strength of the magnetic field around an Android device.

6.1.3 Bluetooth Low Energy

Bluetooth is a standard wireless technology for sending and receiving data. Bluetooth is used for building Beacon protocol. Beacon can be implemented in mobile devices to uses as a radio transmitter. The receiver measures the signal's strength when the transmitter is close; the signal is strong, and when the transmitter is far, the signal is weak.

6.2 Framework Architecture

A framework is an abstraction in which software provides generic functionalities and guidance that allows programmers to develop systems based on the basic building blocks [18]. The architecture of our framework is designed to allow the developer to focus on how to obtain the data collected from the smart environment (i.e., smartphone, wearable device sensors). These sensors are used to trigger events based on user conditions. Our proposition takes advantage of the current capabilities of mobile technology trends related to gathering and processing different types of data that can be used to create proxemic applications. Our approach helps the developer with the development process and the management of proxemic information to establish the appropriate combination of DILMO dimensions and, implement the mobile application in the *P_E*. To support this process, the framework based on android platform is composed by mainly three components aligned with each step (see Figure 6.3): (i) Proxemic Zones module; (ii) DILMO module; and (iii) an API that supports the instantiation of the two previous mentioned components. In the following, we describe each module in detail.

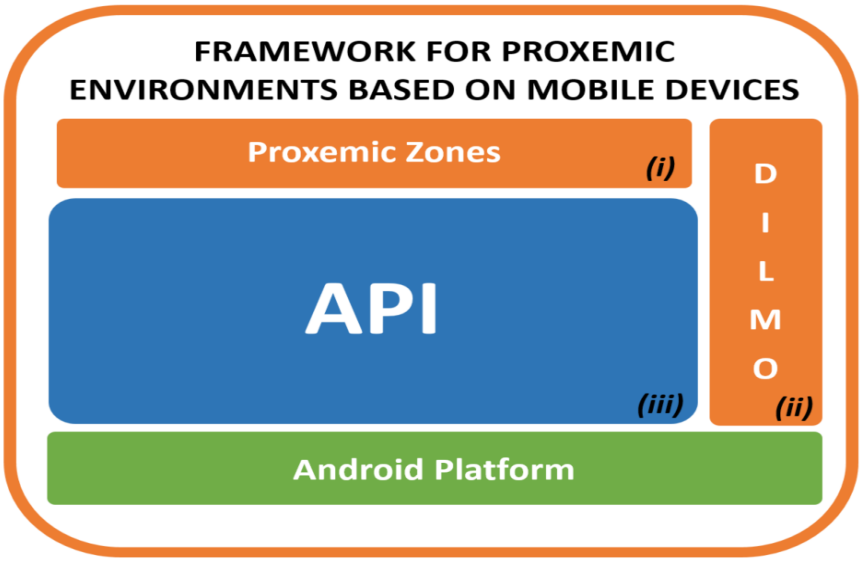


Fig. 6.3 Framework architecture.

The **Proxemic Zones module** allows the definition of proxemic zones sizes (*P_Z*), according to user needs. The interaction between two entities changes in accordance of the *P_Z* in which they are located. The values of distances that delimit the proxemic zones are configured through the API. Figure 6.4 is a guide that allows the developer to know which methods must be implemented on the API or which objects must be created from the API according to DILMO dimensions for processing proxemic information. For example, a DIL proxemic environment (row 9 in Figure 6.4) means that Distance (*D*) and Location (*L*) are considered for Identities (*I*). Combination of proxemic dimensions are also valid, although the entities have not unique identification; e.g., *a person*, *a device beacon*, instead of *the smartphone's owner*, *my device beacon*. Hence, proxemic environments denoted in Figure 6.4.

#	D	I	L	M	O	MIX	Proxemic Environment
1	■					D	Physical length (D) between entities.
2			■			L	Position (L) of an interaction object (entity).
3				■		M	Motion (M) capture of an interaction object (entity).
4					■	O	Face orientation (O) between entities.
5	■	■				DI	Interaction based on proximity (D) between Identities.
6		■	■			IL	Interaction based on the physical location (L) of Identities.
7		■		■		IM	Interaction based on movement tracking (M) of Identities.
8		■			■	IO	Interaction based on face to face orientation (O) of Identities.
9	■	■	■			DIL	Interaction based on proximity (D) and positions (L) of Identities..
10	■	■		■		DIM	Interaction based on proximity (D) according to movement tracking (M) of Identities.
11	■	■			■	DIO	Interaction based on proximity (D) and face to face orientation (O) of Identities.
12		■	■	■		ILM	Interaction based on physical location (L) and movement tracking (M) of Identities.
13		■	■		■	ILO	Interaction based on face to face orientation (O) and position (L) of Identities.
14		■		■	■	IMO	Interaction based on movement tracking (M) and face to face orientation (O) of Identities.
15	■	■	■	■		DILM	Interaction based on proximity (D) and physical location (L) with movement tracking (M) of Identities.
16	■	■	■		■	DILO	Interaction based on proximity (D), location (L) and face to face orientation (O) of Identities.
17	■	■		■	■	DIMO	Interaction based on proximity (D) and face to face orientation (O) according to movement (M) of Identities.
18		■	■	■	■	ILMO	Interaction based on location (L) and face to face orientation (O) according to movement (M) of Identities.
19	■		■			DL	Physical length (D) between entities.
20	■			■		DM	Interaction based on proximity (D) according to movement tracking (M) of entities.
21	■				■	DO	Interaction based on proximity (D) and face to face orientation (O) of entities
22			■	■		LM	Interaction based on physical location (L) and movement tracking (M) of entities.
23			■		■	LO	Interaction based on face to face orientation (O) and position (L) of entities.
24				■	■	MO	Interaction based on movement tracking (M) and face to face orientation (O) of entities.
25	■		■	■		DLM	Physical length (D) between entities.
26	■		■		■	DLO	Interaction based on proximity (D), location (L) and face to face orientation (O) of entities.
27	■			■	■	DMO	Interaction based on proximity (D) and face to face orientation (O) according to movement (M) of entities.
28			■	■	■	LMO	Interaction based on location (L) and face to face orientation (O) according to movement (M) of entitites.
29	■		■	■	■	DLMO	Interaction based on proximity (D) and location (L) and face to face orientation (O) according to movement (M) of entities.
30	■	■	■	■	■	DILMO	Interaction based on all proxemic interaction dimensions.

Fig. 6.4 DILMO proxemic dimensions and nomenclature to describe each combination that is available through the API methods for processing proxemic information.

The **API** facilitates developers processing proxemic information and values. The API is available for free download (see Figure). The API provides 7 classes called (Dilmo, ProxZone, Distance, Entity, Movement, Location, and Orientation) to define the *P_Z*, as well as to manage the different combinations of DILMO dimensions. For example, for a DIL proxemic environment, methods to identify entities (*I*) and to process *D* and *L* are available in DILMO class. Thus, the API behaves as a bridge between the Proxemic Zones and DILMO modules.

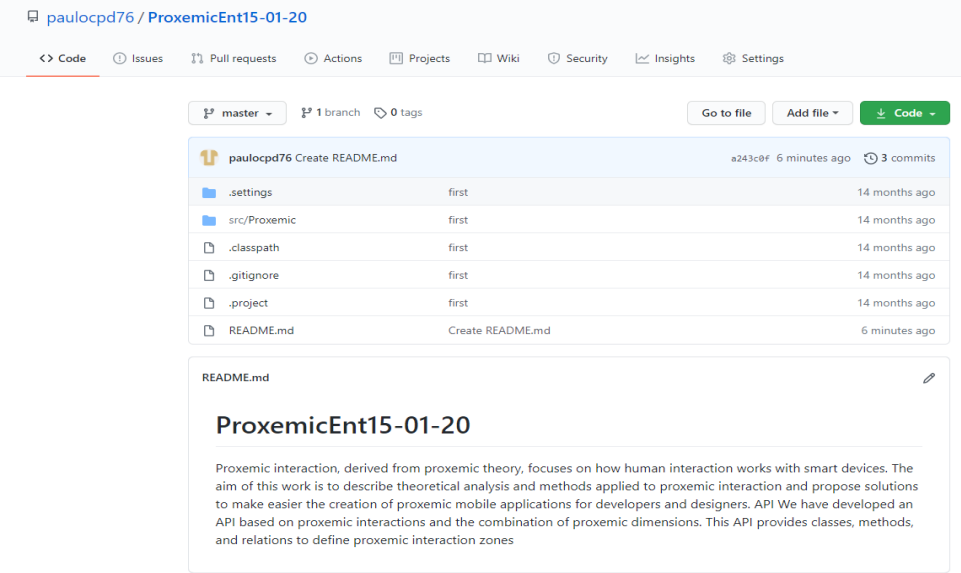


Fig. 6.5 Screenshot: The API [1] website and javadoc documentation.

In the current version of our framework, the API considers the extraction of DILMO values from smartphones or mobile devices based on the Android native libraries (APKs). The API provides methods that the developers can implement for processing proxemic information using motion sensors and mobile computer vision cameras. The majority of current smartphones have a wide range of sensors in their hardware configuration [30], which allow the application to run proxemic apps. For example, through the BLE beacons mechanisms [3], it is possible to know the distance (*D*) between two mobile devices. The distance is estimated using the signal strength of the mobile device's Bluetooth signal. Another way to estimate the distance between two entities is to use computer vision (face detection). Android provide methods for detecting faces as image frame. This image frame size is used to estimate the distance between the user and the mobile phone camera through the API methods.

[1] https://github.com/paulocpd76/ProxemicEnt15-01-20.git/

6.2.1 API Implementation

In this section, we describe the API structure and some of the most important available methods. The API lets developers build *P_Z* and process proxemic information from Android sensors, that are required for implementing a *P_E*. It was developed in Java, hence the `jar` files are provided to be added to the Android Studio platform. Figure 6.6 describes the structure of the API, represented by a UML Class diagram. The main classes are described as follows.

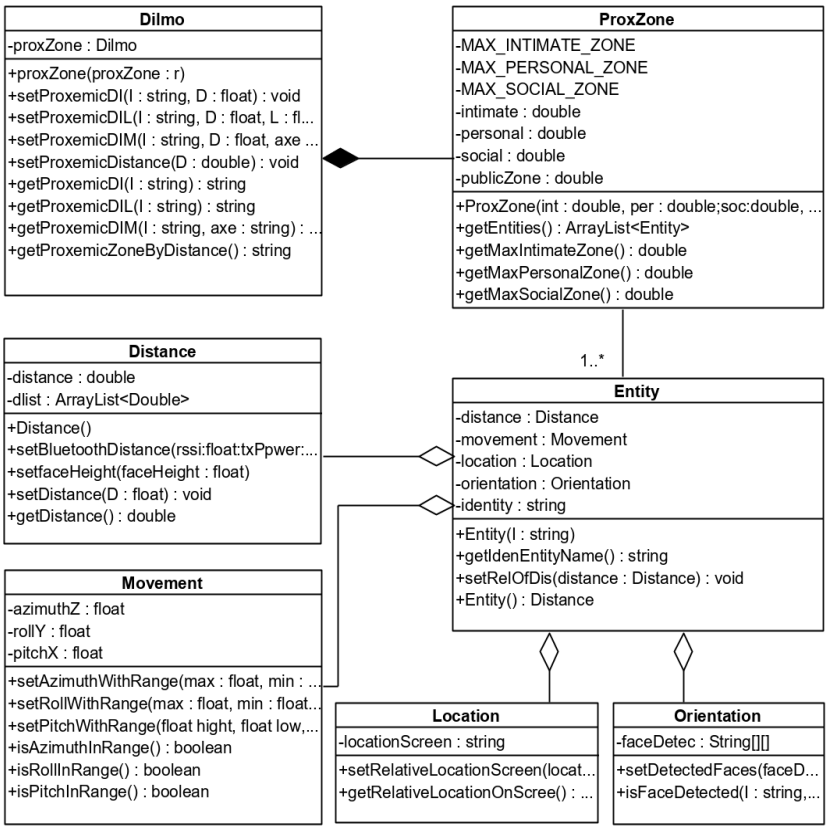


Fig. 6.6 UML Class Diagram of the API (7 classes).

1. The `ProxZone class` allows to define proxemic zones (*P_Z*) according to user requirements (i.e., user/developer decides the measures that delimit each *P_Z*), when this class is instantiated (i.e., by its constructor method). The constructor method of this class receives as parameters the respective maximum measures of distance *D*, which define each *P_Z*. A *P_Z* can be associated to one or more entities.

   Figure 6.7 shows an example of the `ProxZone` constructor method, in which the maximum distance, in meters, for each *P_Z* are specified (see Def. 8): $P\_Z_{intimate}$ is

delimited from 0 to 0.25 meter, $P\_Z_{personal}$ is defined from 0.26 meter to 0.45 meter, P_Z_{social} is from 0.46 meter to 1 meter, and P_Z_{public} is depicted from 1.1 meters to 2 meters.

Inappropriate arguments are validated in order to have valid measurements (e.g., not overlapped zones, right order of measures).

```
//Définition des zones proxémiques utilisée
proxzone = new ProxZone(0.25D, 0.45D, 1.0D, 2.0D)
```

Fig. 6.7 Example of ProxZone Class Constructor invocation.

2. The `DILMO class` is useful for developing P_E. It offers the possibility of identifying the P_Z of all entities E (or identities I) which will interact in the P_E. This class allows to define relations among proxemic dimensions, according to our proposed combinations of DILMO (see Figure 6.4). Its main methods are:

   (a) The `setProxemicDI(String I, double D)` method allows assigning a P_Z to an identity (I), based on the distance (D).

   (b) The `getProxemicDI(String I)` method allows obtaining the P_Z of an identity (I).

   (c) The `setProxemicDIL(String I,double D, float L)` allows assigning a P_Z to an identity (I) based on the distance (D) and processing the location (L).

   (d) The `getProxemicDIL(String I)` method allows obtaining the P_Z and relative location of the identity (I).

   (e) The `setProxemicDistance(double D)` method allows assigning the P_Z to an entity, according to the distance (D).

   (f) The `getProxemicZoneByDistance()` method returns the P_Z of an entity, based on the distance.

3. The `Distance class` allows the developer to estimate the distance among identities (I). Distance (D) can be calculated by using any available method. In the current version of our API, we have integrated some methods to calculate D, based on the Android platform, such as:

   (a) The `setBluetoothDistance(double rssi, double txtPower)` that allows estimating distance based on BLE.

(b) The `setFaceHeight(float faceHeight)`, which allows estimating the distance from camera using visual computing; distance is proportional to the height of the detected face.

(c) The `getDistance()` method, that allows obtaining D in meters.

4. The `Entity class` represents the interaction objects in a P_E, whose behavior is determined or will be determined according to its DILMO proxemic dimensions. This class allows the discovering of the entities on a P_E.

5. The `Location class` provides methods to manage location (L) of interaction objects (E and I). As in the `Distance class`, location (L) of entities can be calculated by any available method. Currently, we have integrated in our API some methods to calculate L, such as:

   (a) The `setRelativeLocationCPS(float L)` method, which sets the relative position of an entity E to CPS, based on the coordinates of E on the CPS.

   (b) The `getRelativeLocationOnCPS()` method, which returns the relative position of an entity E.

6. The `Movement class` has methods that allow motion processing from the coordinate system of smart-mobile sensors (e.g., Azimuth, Pitch, Roll), such as:

   (a) The `setAzimuthWithRange(float MAX, float MIN, float value)` method which allows processing the azimuth angle (rad/s) of an entity E, that has been established by the developer.

   (b) The `isAzimuthInRange()` method returns true if the azimuth angle of an interaction object (E) is within a range of reference.

7. The `Orientation class` provides methods to validate the face orientation (O) of interaction objects on a P_E. Some of them are:

   (a) The `setDetectedFaces(String[][] detectedFa ces,ProxZone p)` method, that receives a collection of faces to be defined as interaction objects (E or I) in the P_E.

   (b) The `isFaceDetected(String I,String P_Z)` method, which returns true if a specific face (I) is detected in the P_Z.

6.3 Proof-Of-Concept of our Framework

Our goal is to demonstrate that our framework allows developers to build proxemic mobile applications effectively. For this purpose, we have tested the implementation of two mobile applications, called IntelliPlayer and Tonic, based on proxemic interactions and developed by undergraduate students. These apps were implemented using Android Studio platform version 3.3; however a higher Android studio version can be used. The two apps[1] are available for free downloading.

Both apps have been developed by undergraduate students, as part of their final project in computer science. The developing team was formed by four students whose average age was 21 years-old, who have developer experience using Java object-oriented programming. They are enrolled in the computer science programs level 2 *(IUT des Pays de l'Adour Bachelors Universitaires de Technologie DUT Informatique, BAC +2).* The experiment was the first challenge for them implementing Android applications and MobileHCI based on proxemic interactions.

Two training sessions of two hours each were organised for the students. The training process allows students to understand the systematic process for building proxemic mobile applications with our framework. They learned: (i) how to define each P_Z; (ii) how to select each combination of proxemic dimension for recreating a P_E; and (iii) how to use methods and classes of the API. The development time of both applications was 64 hours done by two developers over a period of 4 weeks. IntelliPlayer took 44 hours of work, while Tonic was finished in 20 hours, in the same four weeks. Development time was obtained from student interviews and observation of code developing in classrooms. The students developed Tonic mobile application in the first place to understand how the API works and to determine their respective functions for processing proxemic information from mobile sensors. IntelliPlayer was developed in the second plate because this mobile application includes more functionalities than the previous one. Below, we describe both applications that were develop by students as an proof-of-concept.

Tonic is an educational mobile app for learning musical notes, developed for illustrative purposes. In the first step of the approach, the students have defined four proxemic zones: P_Z_{intime} (0 mts to 0.5mts), $P\_Z_{personal}$ (0.51 mts to 1 mts), P_Z_{social} (1.1 mts to 2 mts), and P_Z_{public} (2.1 mts to 4 mts) that were defined in the DSL (see Figure 6.8). In the second step, a DIMO combination was stated for the proxemic environment, P_E.

Tonic allows a user to play a note and modify it from her/his smartphome on another smart mobile. The user's smartphone is identified (i.e., it is an I) by the mobile device which

[1]The APPS are available in https://github.com/llagar910e/

plays the sound (i.e., it is an *CPS*). Thus, *CPS* plays and modifies a sound, by using proxemic interactions based on the *P_Z*, and on *D*, *I*, *M*, and *O* dimensions (i.e., a DIMO proxemic environment). In Tonic, the distance (*D*) between the two devices is obtained by using BLE technology. *I* broadcasts its identifier to nearby portable electronic devices, thus it is caught by *CPS*. The volume of the sound is adjusted according to the *P_Z* in which *CPS* is, with respect to *I* (see Figure 6.8).

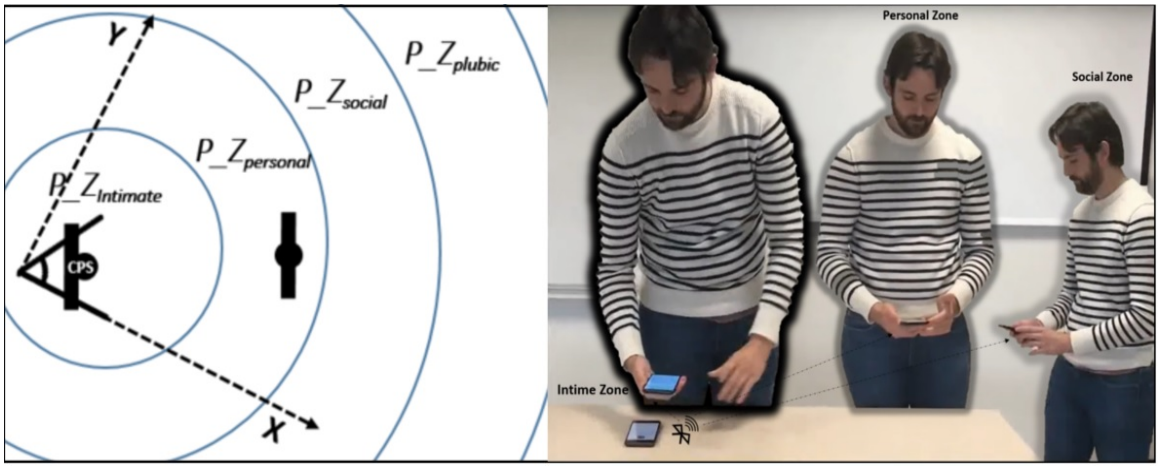


Fig. 6.8 Tonic Proxemic Zones based on Bluetooth Low Energy.

The musical notes are changed according to the movement (*M*) and orientation (*O*) of *CPS*, with respect to *I*. According to *M* a tone is increased/decreased, while according to *O* a semi-tone is increased/decreased. *M* and *O* are calculated based on the capabilities of the smartphone, such as accelerometer, gyroscope, compass, and magnetometer. These sensors provide proxemic information that is mainly used in the API. Movement *M* was determined by using methods `setAzimuthWithRange(float MAX, float MIN, float value)` and `isAzimuthInRange()` described in items 6.(a) and 6.(b) in Section 6.2.1; while *O* was managed by methods based on the smartphone's technical capacities. To manage *P_Z* and *D*, the methods used from the API, in the third step of the approach, were those described in items 2.(b), 2.(c), and 3.(a) in Section 6.2.1: `ProxemicDI(String I)`, `setProxemicDIL(String I ,double D, float L)`, and `setBluetoothDistance(double rssi, double txtPower)`.

IntelliPlayer is a mobile application that plays a video in a smartphone and reacts according to four proxemic zones and DILO proxemic dimensions. In the first step of the approach, the four *P_Z* were created: P_Z_{intime} (0 mts to 0.25mts), $P\_Z_{personal}$ (0.26 mts to 0.45 mts), P_Z_{social} (0.46mts to 1 mts), and P_Z_{public} (1.1 mts to 2 mts). Then, in the second step, the MobileHCI was designed according to *D*, *I*, *L*, and *O*, thus a DILO *P_E* was defined. Figure 6.9 shows the proxemic zones that have been defined by the developer through the API and DSL.

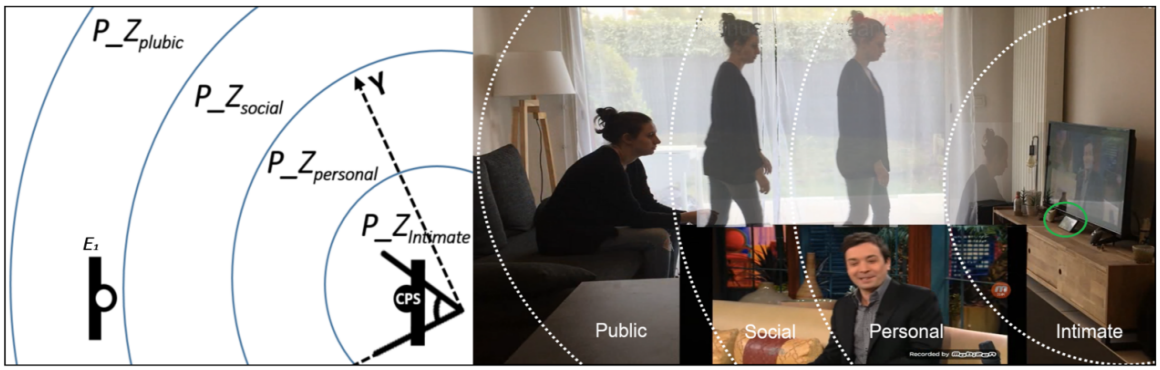


Fig. 6.9 IntelliPlayer proxemic zones.

With this application, The students illustrate a proxemic environment using a mobile player app that reacts to the distance (D) and location (L) of a person (E_1) and his face orientation (O), with respect to the smartphone (CPS) displaying a video. The computer vision technique has been used for this purpose, based on the properties of an Android camera and through the API methods `setFaceHeight(float faceHeight)` and `getDistance()` described respectively in items 3.(b) and 3.(c) in Section 6.2.1. Figure 6.10 shows a block code of this case. With the distance (D) between the user (E_1) and the smartphone (CPS), IntelliPlayer determines the proxemic zone (P_Z) of E_1 (a user), with respect to the smartphone (CPS). To do so, it invokes the method `getProxemicZoneByDistance()` described in item 2.(f) in Section 6.2.1.

```
Distance d= new Distance();
d.setfaceHeight(faces.valueAt(i).getHeight());
String id =String.valueOf(faces.valueAt(i).getId());
dilmo.setProxemicDI(id, d.getDistance());
dilmo.getProxemicDI(id);
changerVolume(dilmo.getProxemicDI(id));
//Log.i("FaceId", "Zone : "+ id + "/"+ dilmo.getProxemicDI(id));
```

Fig. 6.10 Block code of IntelliPlayer.

IntelliPlayer automatically adjusts the volume of the video according to the P_Z in which E_1 (the user) is with respect to the smartphone (CPS): when E_1 is in P_Z_{intime}, it decreases to 25% volume of speaker; for $P\_Z_{personal}$, it increases to 50% volume; for P_Z_{social}, it increases to 75% volume; and for P_Z_{public}, it increases to 100% volume). When a second person (E_2) is in front of the smartphone camera (CPS), the application verifies if both users are looking at the screen at the same time, as shown in Figure 6.11(a) (method `setDetectedFaces (String[][]detectedFaces, ProxZone p)`, item 7.(a) in Section 6.2.1). In the case one user (E_1 or E_2) turns his face, the video will be paused automatically by the mobile application (see Figure 6.11(b)).

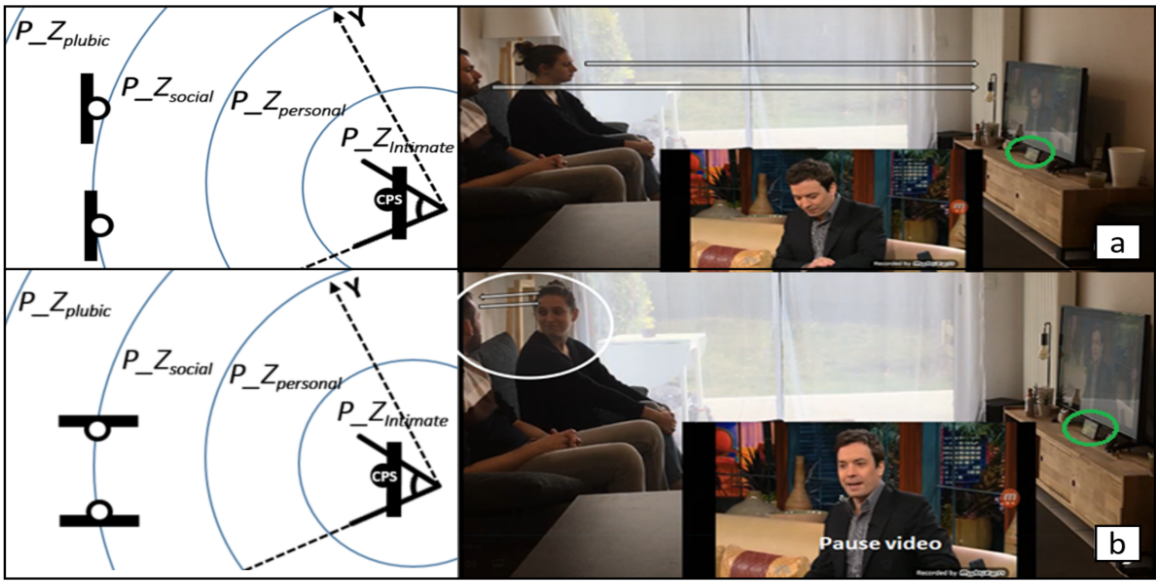


Fig. 6.11 Play pause video using users faces orientation.

Another useful function of IntelliPlayer is to provide a video description that users can read on the screen according to user location (L) (see Figure 6.12). When a user (E_1 or E_2) is in the $P\_Z_{personal}$ and his orientation (O) is in front of the screen, the application can obtain the face location (L) (see `setRelativeLocationScreen(float L)` and `getRelativeLocationOnScreen()` in item 5.(a) and 5.(b) in Section 6.2.1) to split the screen, with the video (running) and information about the video on the right or on the left, according to the detected L. The correct use and instance of methods and classes of the API conforms the third step of the proposed approach.

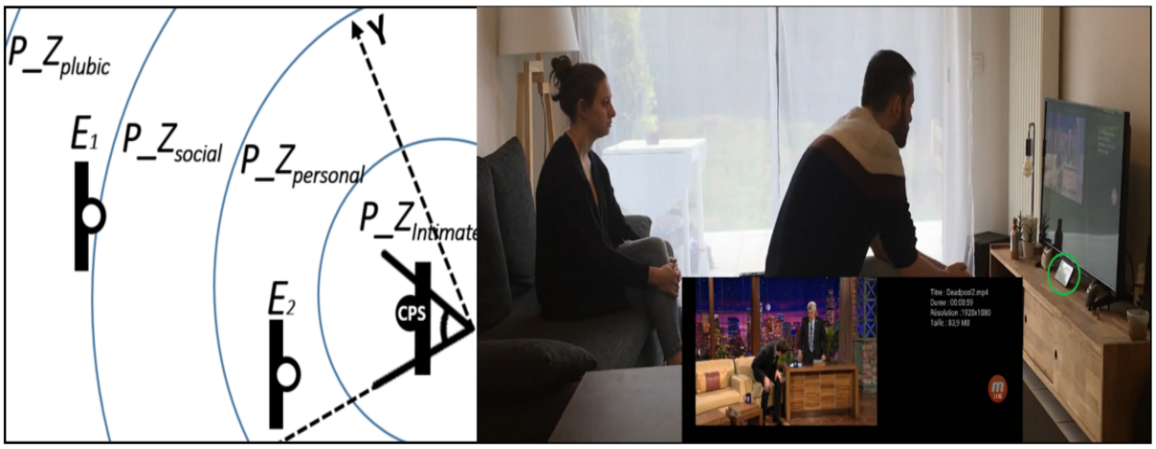


Fig. 6.12 The split view provides video description.

6.4 API Validation

We applied two of more used strategies to validate API and tool-kit according to Lendo [60]. This work analyzed evaluation techniques applied to 68 toolkit papers in the HCI field that allowed them to propose the most relevant criteria from the aforementioned strategies (demonstration, usability study, technical benchmarks, and heuristic).

6.4.1 Demonstration

Our goal was to demonstrate that our API permits students to build efficiently proxemic mobile applications. Through our API, the students developed novelty mobile apps based on proxemics using only mobile device capabilities that offer alternative forms of interaction with mobile devices. For this matter, API process proxemic information from different sensing methods such as mobile computer vision and BLE Technology

6.4.2 Usability Study

It describes the facility of use of the proposed API for the developing of proxemic mobile apps. To this extent, the aforementioned API has an architecture that allows developers to set the initial conditions of a proxemic environment with two lines of code such as `ProxZone p = new ProxZone(0.5,1,4,50) and Dilmo dilmo= new Dilmo(p);` while adding new identities require only one line of code which facilitates the relationships between object and target for example: `dilmo.setProxemicDI("Peter", 0.5)`. The API is supported by documentation that provides information about classes and methods and a table of proxemics nomenclature that provides a guide for selecting methods according to a combination of proxemics dimensions DILMO. In order to evaluate the usability of the API, we applied a survey composed of nine questions to the undergraduate students who developed the applications described. The score of each question was taken from the average of responses. This average is the sum of each student's response divided by the total number of students in the survey. We obtain the answer from the aforementioned survey. Results of the survey are shown in Figure 6.13.

These results indicate that 100% of surveyed students have strongly agreed with **Q1**: *"It was easy to implement the API with Android Studio"* and **Q9**: *"The API is useful for the development of proxemic applications with Android?"*. While 95 % of students endorsed **Q5**: *"The API allowed you to process information of a combination of DILMO dimensions"*, **Q7**: *"The API's method for estimating distance based on face detection was accurate"*, and **Q8**: *"The API allows you to obtain the proxemic zone when it is using Bluetooth proximity*

sensing". For **Q4**: *"The API allows you to improve your productivity for developing proxemic applications by hiding complexity"* and **Q6** *"The API allows you to create the proxemic zones quickly"*, 78% of students expressed acceptation. Finally, **Q2**: *"The documentation provides enough information to interact with the API"* and **Q3**: *"The API provides enough examples for creating proxemic applications"*, were accepted only by 53% and 61% of students, respectively.

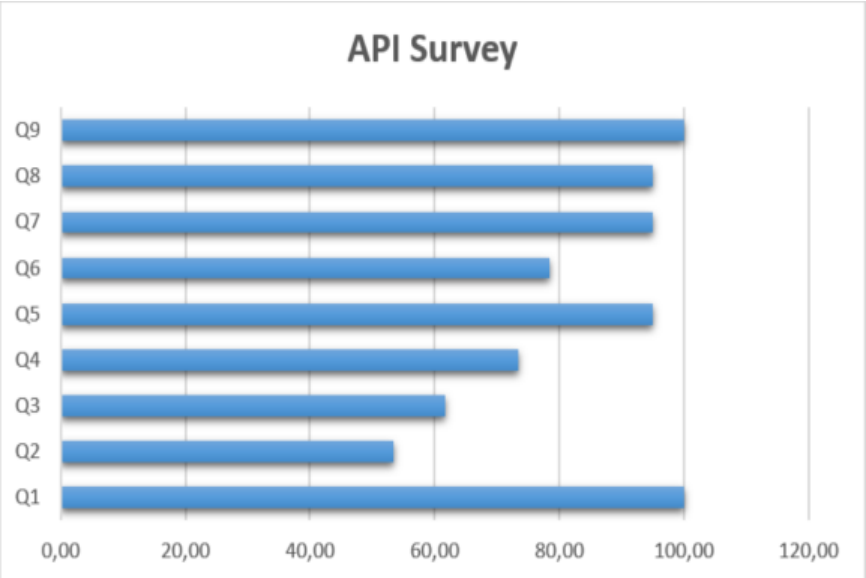


Fig. 6.13 Results of students feedback.

The proof-of-concept and survey allow corroborating that the API supports development of proxemic applications and creation of proxemic environments, based on the exclusive use of mobile devices. These applications offer portable implementations that facilitate using the proxemic interaction in comparison to previous works that have used fixed platforms for similar purposes such as ProximiThings [16] and Toolkit [64]. Moreover, the survey results allow us to know aspects to be developed, such as the quality of documentation and the examples presented in the API.

6.5 Conclusion

Mobile technologies are frequently used in daily life for different activities, and their use keeps increasing. This chapter introduced a framework that includes an API for developing proxemic applications for smart environments comprised of entities whose interactions are supported by proxemic dimensions DILMO. We demonstrated the framework's effectiveness through the proof-of-concept, which details the implementation of two proxemic mobile

applications developed by undergraduate students in computer science. With this framework, we provide a tool that can help developers to build novelty proxemic mobile applications using social distancing.

The following chapter describes two mobile applications implemented using our framework and DSL in the current pandemic scenario to avoid virus spread. We present two prototypes of two mobile applications as a proof-of-concept, using several combinations of proxemic DILMO dimensions in accordance with real-life needs.

Chapter 7

Proxemic Mobile App Design and Development to Avoid the Spreading of Infections

In this chapter, we illustrate how proxemic interaction in mobile devices that can avoid the spreading of nosocomial infection. We use our DSL to design the proxemic environment and the framework to develop two mobile applications as proof-of-concept and validation of our approach. The pandemic produced by the COVID-19 has affected people's daily lives. One of the main problem is the quickly and easy spreading of the virus, that causes a huge quantity of infected people assisting to healthcare centers. Thus, nosocomial infections (also called as hospital-acquired infections) of COVID-19 affect Health Care Workers (HCWs) in the workplace according to recent studies [62, 89]. This infection is defined as contamination during the process of care in a hospital and also it is considered as occupational infections among HCWs. HCWs use technologies in hospital, laboratories, intensive care units, and operating rooms; thus during any interaction with computers and mobile devices, they come into close contact with strongly contaminated devices.

7.1 Avoid the Spreading of Infections: motivation research

Previous studies demonstrated drastic reductions of nosocomial transmission of COVID-19 thanks, in large part, to physical distancing among people in hospitals [57, 82, 15]. However, contamination of computer user interfaces and mobile devices, due to their touchability, are frequent in the clinical workrooms contributing to spread of COVID-19 [91]. Several studies suggest touchless interaction to avoid the contact of contaminated materials [37, 67, 55, 78].

Touchless interaction has been used in operating rooms (ORs) during surgical procedures, notably to provide control of medical image analysis, such as the work proposed in [67], which uses the notion of proxemic and touchless interaction systems in surgical contexts. However, previous studies reveal that nosocomial transmissions is also a real problem for HCWs in other areas of the hospital, even beyond the OR [23, 83].

Mobile technologies in hospital are increasing in a way without precedent in different areas of the hospital. HCWs can interact in different contexts through electronic devices, such as personal mobile phones, tablets, and wearable technologies, to accomplish their daily tasks, based on specific Human-Computer Interaction (HCI). Some studies have emphasized that mobile devices are reservoirs for pathogens with potential to cause nosocomial infections [22, 47, 26].

In order to reduce the HCWs contamination through mobile devices, we propose the use interpersonal distances to develop mobile applications based on proxemic HCI, using combinations of proxemic dimensions – i.e., Distance, Identity, Location, Movement, and Orientation (DILMO dimensions). We are interested in handling interpersonal distances for interacting with mobile devices, according to different combination of DILMO dimensions. We are therefore seeking to develop new useful user interfaces, that allow HCWs to reduce the touchability of mobile devices and limit the spread of nosocomial infection of COVID-19. We also propose a development architecture based on mobile technology to support the easy construction of proxemic HCI for mobile apps. To show the usability and suitability of our proposal, we present two prototypes of apps for mobile devices as proof-of-concept, using several combination of DILMO dimensions to model proxemic HCI that allow flexibility in interpersonal and devices-people interactions.

7.2 HCI to reduce the risks of nosocomial infection for HCWs

In this section, we present a review of studies that describe the difficulties and risks of HCWs using computer equipment in the workplaces and how HCI can be used to reduce such as risks. We also survey applications that show how proxemic DILMO dimensions have been used to implement interactions among digital devices and devices-people.

Today, HCWs are facing a difficult situation in the workplace because many of them have acquired COVID-19 disease during medical service [62]. The HCWs are exposed to a nosocomial infection that exists in a specific location, such as a hospital and more specifically from health care equipment. In fact, the use of technology is very common in health care settings however it is a difficulty for fully secure access to computer controls (sterilization for example) which increases opportunities to spread pathogens [23]. Previous studies have

shown microbial contamination of computer peripheral like keyboard and mouse used to enter information in a care center during the workday [34]. Moreover, recent studies show that the use of mobile phones inside hospitals might serve as repositories of microorganisms that could be rapidly transmitted from the mobile phones to the HCWs' hands and therefore can help the transmission of bacterial from one patient to another person [7, 52].

Besides, Harvard University researchers published results of the simulation of a mathematical COVID-19 model [56, 57], which predicts that recurrent winter outbreaks will probably happen after the first, most severe pandemic wave; prolonged or intermittent physical distancing may be necessary until 2024. Therefore, it is vitally important to reduce the spread of COVID-19. Previous studies demonstrated drastic reductions in infections and deaths thanks, in large part, to the physical distancing among individuals [57, 82, 15]. Physical distancing in the health centers is important for HCWs. Several studies suggest that implementing physical distancing in the hospital helps to prevent nosocomial transmission of COVID-19 and ensures the health of HCWs to meet the hard challenge. For example, the study presented in [5], suggests the reallocation of mobile workstations, laptops, or desktops to private rooms that avoid all sources of contamination and clean surfaces such as keyboards and desktops frequently because COVID-19 can survive on surfaces for several hours.

Some studies have proposed the use of computer equipment in hospital environments based on touchless interaction to help decrease the risk of nosocomial contamination and prevent nosocomial transmission of COVID-19 [23, 51]. In particular, the work proposed by O'Hara in [67] uses touchless interaction and voice commands to control and manage imaging technologies within a surgical setting where touchless inputs might permit new kinds of interaction during surgery.

Touchless computer interfaces are widely used in ORs for decreasing opportunities for the spread of pathogens on computer controls and facilitating HCI [67, 49, 23]. However, the Health Informatics Journal [23] has recommended to explore the use of touchless systems in other areas of the hospital environment. Besides, touchless systems [37, 67, 55, 78] have utilized Microsoft Kinect sensor, which produces motion tracking with far high accuracy. Notwithstanding, the studio presented in [19] demonstrates that mobile technologies could offer radiology residents the highest usability when employing touchless interaction.

All those previous mentioned works demonstrate the current interest for researchers and medical professionals to decrease physical interactions with computers to avoid spreading of infections. In this context, proxemic interactions can play an important role to implement HCI that reduce touchability of devices. Therefore, it is necessary to provide software architectures on mobile devices that incorporate DILMO dimensions for reducing the risk of cross-contamination with mobile devices.

7.3 DILMO in Medical environments

In this section, we demonstrate how the combination of DILMO proxemic dimensions can be used to create appropriate proxemic HCI for applications in medical environments, to improve seamless interactions between digital devices and users, while reducing the risks of noscomial transmissions. Proxemic dimensions are captured from sensors, which means that a variety of hardware technologies can be substituted or combined for sensing proxemic information [64]. Thus, measures of DILMO dimensions should be gathered from the available technology (i.e., sensors) in mobile devices.

Table 7.1 shows examples of combination of DILMO dimensions in medical environments and the hardware of mobile devices required for sensing proxemic information from the environment. Currently, most mobile devices are equipped with a camera, GPS facilities, wireless connections as Bluetooth, and Bluetooth Low Energy Beacon (BLE) technology. At least one sensor must be implemented for sensing proxemic information. It's possible to use a combination of mobile sensors to obtain multiples proxemic dimensions that privilege the touch-less interaction in medical environments.

Table 7.1 DILMO Medical enviroments

DILMO	Description	Camera	BLE	GPS
D	Distance between HCWs and devices	✓	✓	.
I	HCWs' identity or device identifier	✓	✓	.
L	Medical equipment position	.	.	✓
M	HCWs Gesture	✓	.	.
O	HCWs' gaze or face orientation	✓	.	.
DI	Physical distancing between entities	.	✓	.
IO	HCW's orientation and identity	✓	.	.
DIO	IO in a proxemic zone	✓	✓	.
DIMO	IO in a proxemic zone and body tracking	✓	.	.

We characterize some DILMO combinations and how proxemic information can be obtained from mobile devices. Furthermore, we describe semantic information of such as combinations that can be used to create mobile applications for medical environments.

In the following, we describe some examples of DILMO combinations in medical service applications that allow limiting touch interactions. It is not described as an exhaustive list, some other DILMO combinations can be defined, according to the available technology and requirements of applications. Actually, the inverse condition can also be exploited: DILMO combinations can characterize the hardware necessary to implement medical applications on mobile devices, in which the user interaction with contaminated devices is avoided. Thus, it

is possible to identify DILMO combinations according the available hardware or to decide which mobile hardware is needed to implement specific DILMO combinations.

- **DI** combination allows identifying specific HCWs (Identity) and their proxemic zones (according to Distance among them). This combination enables HCWs to keep physical distance with others using mobile devices. Moreover, the use of **DI** allows the development of applications that interact using the proximity between HCWs or between HCWs and medical device.
- **IO** combination allows facial recognition (Orientation) of authorized users (Identity) for access control using mobile computer vision, for areas such as laboratory, pharmacy, emergency rooms, and isolation rooms, for example, to access sensible patient information or private services.
- **DIO** combination offers HCWs the possibility to interact with information systems avoiding physical contact with computer peripheral. With BLE technology it is possible to estimate Distance between mobile devices and to get their Identities; while mobile computer vision allows sensing HCW's orientation and identification. This combination can avoid the usage of peripherals, which are difficult to sterilise and a potential source of contamination, such as mouses, keyboards, and smartphones.
- **DIMO** combination allows building mobile apps for gesture interaction (Distance, Movement, and Orientation) that help the HCWs (Identity) to manipulate information (such as images) displayed on the screen, using mobile computer vision and body tracking. DIMO applications provide more direct control of medical procedures based on gesture interaction captured with mobile computer vision.

These DILMO combinations are the base to develop some applications intended to reduce the spread of nosocomial infection of COVID-19 in hospital environments. In the next Section 7.4, we propose an architecture aimed to support the development of such as applications, based on the basic technology of mobile devices.

7.4 Architecture to support the development of DILMO-based Mobile applications

Our proposed development architecture, shown in Figure 7.1, is supported from our API and BLE libraries that allows developers to manage different social distances according to requirements of the specif application. Through the mobile devices sensors, the proxemic

information is gathered (i.e., DILMO measures), according to the desired combination (see Table 7.1). We describe each component of our proposed architecture as follows.

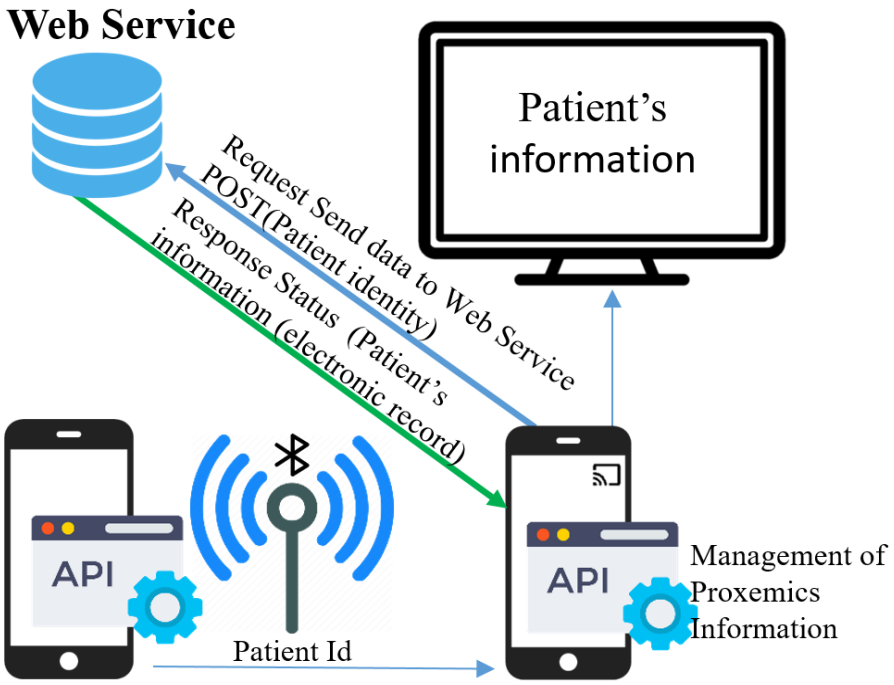


Fig. 7.1 Architecture to support development DILMO Mobile applications

7.4.1 Proximity: Virtual Bluetooth Low Energy Beacon (BLE) on smartphone

To create a virtual beacons BLE in a mobile device, it is required to use Beacon protocols. We employ the Java android library AlteBeacon [3], which allows to estimate the physical distance between two devices and catch their corresponding UUID (identity). In our architecture, mobile devices transmit string data to another mobile using BLE, which is processed in real-time through the API.

7.4.2 API for proxemic social interaction management

We implemented an API that lets developers define the proxemic zones required for implementing mobile application based on social distancing. Depending on the application to be developed, it will be necessary to define several proxemic zones (as the four proxemic zones proposed by T. Hall [46]). Through the API, developers are able to define and customize the size of the proxemic zones. It also offers methods that allow developers to obtain the proxemic zone of entities, based on the distance between mobile devices. The

API offers methods to process DILMO information locally, without a connection to a server. However, connection to the server can be established according to DILMO conditions. The `ProxZone class` allows to define proxemic zones according to user requirements (i.e., the user/developer is able to decide the measures that delimit each proxemic zone when this class is instantiated by its constructor method). The `DILMO class` offers the possibility of identifying the proxemic zones of all entities which will interact in the environment; and processing the DILMO information from mobile device sensors.

7.4.3 Web Service for storing patient information

The proposed architecture uses a web service, which has been created for experimental purposes. The patient information is provided according to client needs (health centres). We implement a post method that sends a patient key to a server from HCW's smartphone. The patient information is then returned by the server to the mobile phone, which in turns transfer that information by cast receivers to a screen in the room. The access to the web server and the screen should be configured in HCWs' mobile devices and it will be allowed according to the implemented DILMO combination. The interpersonal distances are locally managed by the mobile devices using Bluetooth and the API.

7.5 Prototype: Proxemic Mobile Applications based social distancing POC

To demonstrate the applicability and suitability, in the medical area, of our proposal, we built two mobile applications with proxemic HCI, as proof-of-concept, which are based on some DILMO combinations. These applications illustrate how the need of reducing the spread of bacteria associated with nosocomial infections, can be approached, by avoiding computer's peripheral and mobile devices touching and maintaining physical distance among HCWs. The scenarios described with both applications give a better understanding of proxemic interactions in medical environments.

The first application, called *InZone-19 DIO*, allows careful social interactions in hospital environments, reducing physical contact with hardware and limits the spread of nosocomial infection of COVID-19. The second one, called *InZone-19 DI*, helps to keep physical distancing between two HCWs. Table 7.2 summarizes API's methods, AlteBeacon (AlteB) libraries and mobile vision methods from Android native libraries (APKs) that were implemented to build both *InZone-19 DIO* and *InZone-19 DI* applications. Our API considers the extraction of DIO values from smartphones or mobile devices based on the Android operating system

by using the Bluetooth technology and the camera that the majority of smart devices have in their hardware configuration.

Table 7.2 Methods Implemented

API/APKs/AlteB	Class	Constructor and Method	*InZone-19 DI*	*InZone-19 DIO*
API	ProxZone	ProxZone(0.5, 1.0, 4.0, 50.0)	✓	✓
API	Dilmo	Dilmo(proxzone)	✓	✓
API	Dilmo	setProxemicDistance (distance)	✓	✓
API	Dilmo	getProxemicZoneByDistance()	✓	✓
API	Dilmo	setProxemicDI(uuid,D)	.	✓
API	Dilmo	getProxemicDI(uuid)	.	✓
APKs	Frame	Builder()	.	✓
APks	Face	detect(frame)	.	✓
AlteB	Beacon	getId1()	✓	✓
AlteB	Beacon	getDistance()	✓	✓

7.5.1 InZone-19 DI (scenario 1): Mobile app reacting to distance and identity

An important strategy to limit nosocomial transmission is to implement *physical distancing* or avoiding close contact with others [5]. This *physical distancing* should be kept during the whole work day and in every where in the hospital. For example, during meal time, HCWs are encouraged to maintain *physical distancing*. To help maintain this requirement, we implement *InZone-19 DI* app, a **DI** proxemic application. Design of proxemic environment for scenario 1 formal definition and graphical design (see Figure 7.2).

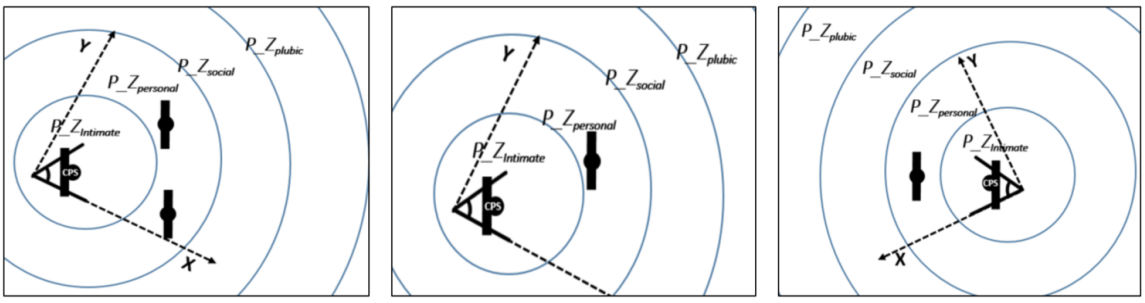


Fig. 7.2 Scenario 1: Design of proxemic environment graphical representation for each HWC who needs to keep physical distancing.

- The four tuple of P_E: $P\_E =< CPS, identities\{I_n\}, distances = \{0.5, 1, 4, 50\} >$;

- Distance of identities: $I.D = 1$;

  $\measuredangle_{MinAofV} \leq I.O, \leq \measuredangle_{MaxAofV}$;

- Proxemic zones of entities:

  $I.P\_Z{=}P\_Z_{personal}$

  From these conditions, the *CPS* can react with $Action_1$, that can be denoted in several ways, for example:

  $if\ \exists\ I$ in $P\_Z_{personal}$ then $CPS.Action_1$.

To illustrate this scenario, we explain the use case shown in Figure 7.3: there are three HCWs in the hospital hall, which carry their smartphones in their pocket, running the *InZone DI* app that retrieves UUIDs around it. When a HCW invades the personal zone of another HCW, their smartphones trigger an alarm, which allows users to know that they are infringing the *social distancing*. In this way, the app creates a virtual security bubble which keeps physical distancing between two or more individuals only using smartphones.

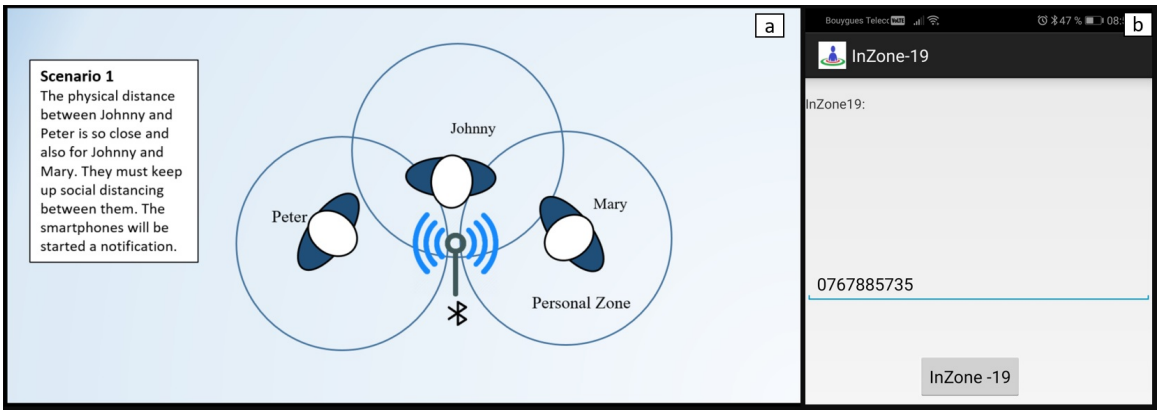


Fig. 7.3 Scenario1: (a) Physical distancing between HCWs exclusively using Bluetooth BLE mobile devices. Each HCW also acts as an *CPS* there are multiples P_E; (b) Graphical user interface (GUI) that allows the HCW to start virtual security bubble based on his/her id phone number.

7.5.2 InZone-19 DIO (scenario 2): Mobile app reacting to distance, identity, and orientation

In a health care scenario, where touchable interaction must be limited to the minimum necessary between HCWs and mobile devices, we use an appropriate subset of proxemic dimensions (**DIO**, as shown in Table 7.1). DIO combination allows developing a mobile

application which can help HCWs to obtain electronic patient record using a mobile phone without touching the mobile device.

Design of proxemic environment for scenario 1 formal definition and graphical design (see Figure 7.4).

- The four tuple of P_E: $P\_E =< CPS, entities = identities = \{I\}, distances = \{0.5, 1, 4, 50\} >$;
- $CPS.L = (0,0)$;

  $\measuredangle_{MinAofV} = 0°$ and $\measuredangle_{MaxAofV} = 60°$;
- Distance of entities: $I.D = 1$;

  Orientation of entities (faces of entities are in the area of vision of the CPS): $I.O = 30°$, thus

  $\measuredangle_{MinAofV} \leq I.O, \leq \measuredangle_{MaxAofV}$;
- Proxemic zones of entities:

  $I.P\_Z$=$P\_Z_{personal}$

  From these conditions, the CPS can react with $Action_1$, that can be denoted in several ways, for example:

  $if\ \exists\ I$ in $P\_Z_{personal} \wedge \measuredangle_{MinAofV} \leq I.O \leq \measuredangle_{MaxAofV}$, then $CPS.Action_1$.

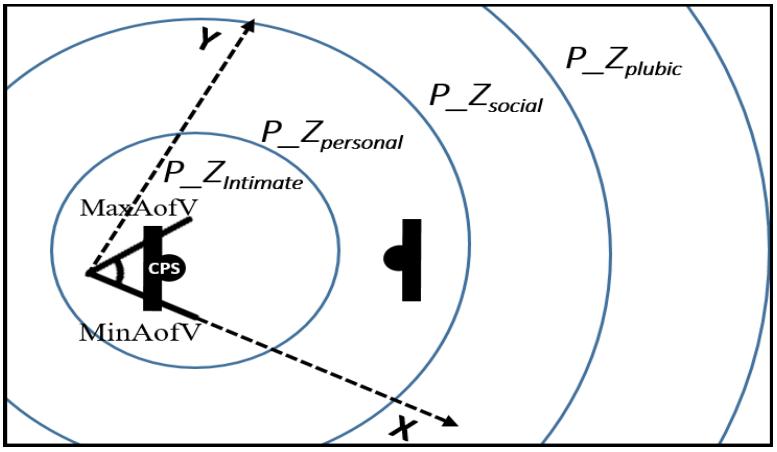


Fig. 7.4 Scenario 2: Design of proxemic environment graphical representation.

To illustrate our scenario, we consider a positive COVID patient, who lies in a hospital bed. A mobile phone, with *InZone-19 DIO* installed, remains on the left side of the hospital bed, let's called it `patient device`. The `patient device` is acting as a virtual BLE beacon and shares identification by establishing a Bluetooth connection with the mobile

devices of HWCs, let's called them `HWC devices`, which also is executing *InZone-19 DIO* app.This scenario, shown is Figure 7.5, illustrates how proxemic interaction decreases the risk of nosocomial transmission of COVID-19, avoiding the physical manipulation of mobile devices and computer peripheral equipment in hospital rooms.

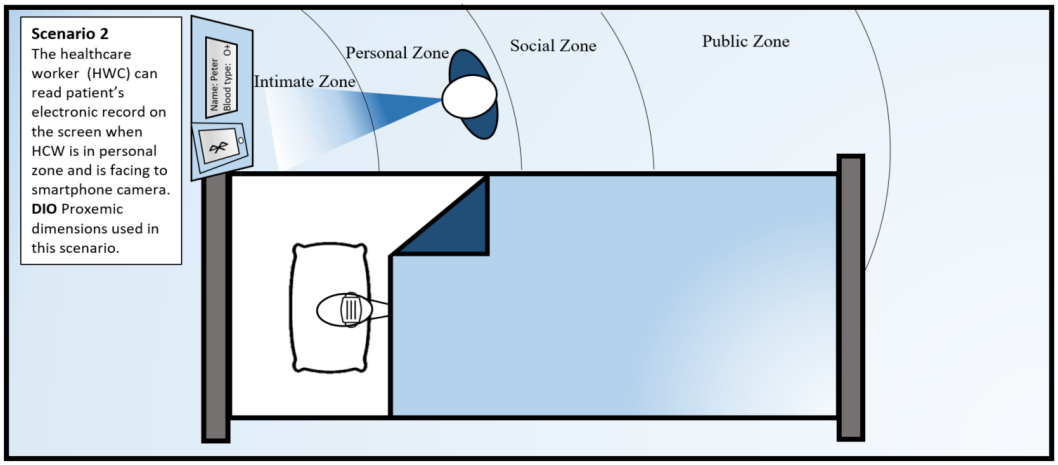


Fig. 7.5 Scenario 2: HWC reads a patient's electronic record from the screen using proxemic interaction in which the HCW avoid physical contact with his mobile phone.

When an HWC is in the intimate zone or personal zone of the `patient device` (Distance dimension), and her/his face is oriented towards its camera (Orientation dimension), the identification of the patient (Identity dimension) is sent to the `HWC device`, which is in the pocket of the HWC. Figure 7.6 shows an HWC situation in which the HWC is carrying a wrapping smartphone in plastic in the pocket to avoid physical contact with the smartphone during a medical check-up.

Fig. 7.6 HWC carries her smartphone in her pocket.

An application in the `HWC device` sends a request to the server, along with the identification obtained from the `patient device` by Bluetooth, asking to the web server for the whole information of the patient. Then, the HCW can read the patient's electronic records on the screen in the wall of the room, exclusively using proxemic interaction. Figure 7.7 shows an example of hospital environment using the mobile application inZone-19 where the HWC is carrying her smartphone in a safe place out of the nosocomial infection.

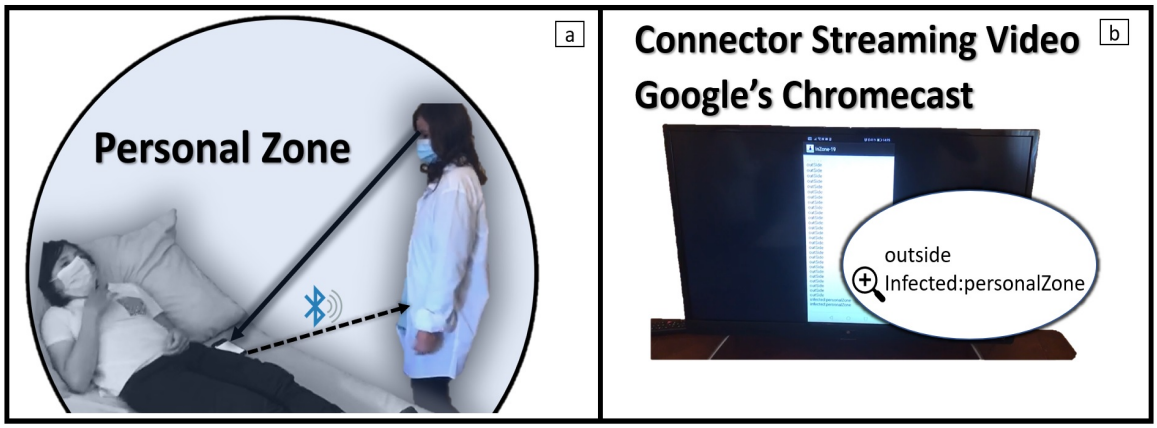


Fig. 7.7 Hospital environment: (a) HCW is patient's personal zone and is oriented to the phone's camera; (b) The HCW's smartphone transmits the information from server to chromecast TV.

7.6 Conclusion

In this chapter, we explore how proxemic interactions can exploit the capabilities of mobile devices to the needs of the healthcare sector improving HWCs interaction on mobile devices. Based on the proxemic dimensions (i.e., Distance, Identity Orientation – DIO), it is possible to implement proxemic HCI that reduces the touchability of mobile devices. To support the development of mobile applications based on proxemic HCI, we also proposed a development architecture, lying in mobile devices technology and an API such as that shown in Figure 7.1. We presented two scenarios based on combinations of DIO dimensions, which help to limit physical contact and touch interaction with mobile devices using interpersonal distances. Both applications were developed with the proposed architecture, which allows developers making suitable several combinations of subset of DILMO to each scenario. We described the development of two mobile applications base on our DSL and framework as a proof-of-concept using proxemic interaction in the current pandemic scenario to avoid virus spread in line with people's real-life needs.

Chapter 8

Conclusion and Future Works

In this research, we proposed an approach with the aim to develop proxemic mobile interaction which can be implemented in mobile devices. Moreover, this dissertation investigated the social theory of proxemics and how it is used to improve the HCI in an ubiquitous environment. We analyzed and investigated the evolution of HCI based on proxemic interaction.Additionally, we described previous proposals that implemented all DILMO proxemic dimensions or a subset of them. Besides, we summarized sensing technologies that were implemented with the purpose to obtain proxemic information by previous proposal. We examined the current tools aiming to design and implement proxemic systems in order to identify proxemic mobile development weaknesses. The main contributions of this study can be summarized as follows.

8.1 Contributions

- **First Aid Mobile Application (FAMA)**, we have developed a proxemic mobile application that identifies an unconscious or an injured person using smart mobile devices. We focused on studies and proposals using proxemic interaction that support HCI. Furthermore, we also evaluated several first aid mobile applications that currently exist in the marketplace. Our first proxemic mobile application lets us know and determine relations between entities and people, such as digital devices, non-digital devices, and persons using DILMO proxemic dimensions. This study showed the advantage of implementing BLE Beacon for obtaining the proximity between two entities. Moreover, this first work demonstrated that proxemic interaction in mobile devices allows users more seamless and natural interactions between human and mobile devices. This experience enabled us to highlight that current proxemic tools did

not offer achievable solutions to design and implement proxemic mobile applications. Published in *Conference on l'Interaction Homme-Machine (L'IHM 2018).*

- **Proxemic Environments Modelling Based on a Graphical Domain-Specific Language**. In this case, the purpose was to present a common language that supports the design of proxemic applications in the design phase towards the implementation process's uniformity, which inspired on the notation system proposed by Edward T. Hall. Additionally, we proposed an intuitive symbology based on graphical objects that allowed designers representing all components in a proxemic environment such as entities, DILMO dimensions, and proxemic behaviours. We described proxemic environments using a formal notation. Besides, we have created a prototype in order to support the specification of proxemic environments from design phase. This language is the foundation of the design of proxemic representation environment, which can continue to evolve for supporting tools that help end-users without programming knowledge to implement their design. Published in *International Conference on Computer Systems and Applications (AICCSA 2020).*

- **Framework for Developing Proxemic Mobile Applications**. We proposed a framework that helps developers to build proxemic mobile applications based on DILMO proxemic dimensions governing the HCI field. The framework was based on a sequential process in order to enable developers to build proxemic mobile applications supported by an API, in which the end-users only need to use mobile devices or wearable devices in order to run the applications. The prof-of-concept and survey allowed us to validate the API. The aforementioned API allowed developers to create applications, by reducing the difficulty level of implementation. Moreover, the API's methods can be updated, and it is possible to add new methods with the purpose to improve the processing of proxemic information from mobile device sensors. Published in *International Conference on Advances in Mobile Computing & Multimedia (MoMM 2020).*

- **Proxemic Interactions in Mobile Devices to Avoid the Spreading of Infections**. Our work previously brought us the technical and theoretical background necessary for developing proxemic mobile applications that respond to society's real-life needs. This work aimed to facilitate the mobile application development in the healthcare sector, improving HCWs' interaction on mobile devices. We developed two prototypes of the mobile applications, which involved a combination of proxemic dimensions in order to stop spreading nosocomial infection of COVID-19 and others. Solutions were proposed with the aim to avoid physical contact among computer peripheral, mobile devices, and

HCWs. Our solutions allow HCWs to maintain social distancing between HCWs using mobile devices. Additionally, we reduced touchability between the user and mobile devices and maintained semantic interaction. Our proposal was based on low-cost architecture web and current mobile devices capabilities that allow healthcare sector to implement proxemic applications in different hospital environments. Published in *International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2020).*

8.2 Future Work

In this thesis, we have proposed an approach that allows the development and implementation of proxemic applications in mobile devices using mobile sensors. We describe possible directions for future works that would advance our study and improve the tools proposed and experimental experiences.

8.2.1 DSL notation and prototype user interface

We developed a prototype that supports the specification of proxemic environments from our graphical DSL. Notwithstanding, the graphical user interface can be improved by adding a drag and drop facility. It could also be implemented with augmented reality or tangible interface in order to improve the user experience when designing proxemic environments. The current Domain-Specific Language (DSL) is based on a symbology which does not include notation for representing gesture control. Our research has addressed the modeling of proxemic environments aiming to represent entities' proxemic behavior. With this knowledge, it is possible to implement a new DSL version including both the validation and approach to reporting results from the experiment.

8.2.2 API methods

The current version of the API considers the extraction of DILMO values from smartphones or mobile devices based on Android native libraries (APKs). The new smartphone mobile computer vision technologies have included stereo depth cameras that improve the detection of human motion tracking. In this respect, we can add more methods that allow developers to quickly implement the motion tracking of objects or people using libraries like TensorFlow and OpenCV for android studio. Combining proxemic interaction and depth mobile computer vision can offer a wide range of applications such as gesture interaction, gesture control, face

recognition, and object with more accurate detection that allows end-users to improve their mobile HCI.

8.2.3 New scenarios for evaluating DSL and Framework

Proxemics described how individuals perceive their personal space relative to the distance between themselves and others. This theory allows us to create mobile applications reducing physical contact with hardware keeping up interactions with devices. Nonetheless, we can continue to find more meaningful scenarios in hospital environments to avoid additional complexity used healthcare equipment applying proxemic interactions. Through this research, we show that proxemic interactions can be implemented with mobile devices in different contexts, which may be adapted according to the use of people's physical space such as interactive public display, multimedia controller, information Kiosks, and medical area, etc. Extend the validation of the framework' API with more experimentation and with more developers' users.

References

[1] 2BEID (2018). Medical bracelets. https://2beid.com/en/.

[2] Abawajy, J. H. (2009). Human-computer interaction in ubiquitous computing environments. *International journal of pervasive computing and communications*.

[3] AltBeacon (2018). Altbeacon - the open proximity beacon. https://altbeacon.org/.

[4] Appventive (2018). Ice: In case of emergency - appventive. http://www.appventive.com/ice.

[5] Arora, V. M., Chivu, M., Schram, A., and Meltzer, D. (2020). Implementing physical distancing in the hospital: A key strategy to prevent nosocomial transmission of covid-19. *Journal of hospital medicine.*, 15(5):290–291.

[6] Bâce, M., Staal, S., Sörös, G., and Corbellini, G. (2017). Collocated multi-user gestural interactions with unmodified wearable devices. *Augmented Human Research*, 2(1):6.

[7] Badr, R. I., Badr, H. I., and Ali, N. M. (2012). Mobile phones and nosocomial infections. *Int j infect Control*, 8(2):1–5.

[8] Bakker, S., Hausen, D., and Selker, T. (2016). *Peripheral Interaction: Challenges and Opportunities for HCI in the Periphery of Attention*. Springer.

[9] Ballendat, T., Marquardt, N., and Saul, G. (2010). Proxemic interaction: designing for a proximity and orientation-aware environment. In *Proc. of Internat. Conf. on Interactive Tabletops and Surfaces*, pages 121–130.

[10] Bhagya, S., Samarakoon, P., Sirithunge, H. C., Viraj, M., Muthugala, J., Buddhika, A., and Jayasekara, P. (2018). Proxemics and approach evaluation by service robot based on user behavior in domestic environment. In *Proc. of Internat. Conf. on Intelligent Robots and Systems*, pages 8192–8199.

[11] Boulos, M. N. K., Wheeler, S., Tavares, C., and Jones, R. (2011). How smartphones are changing the face of mobile and participatory healthcare: an overview, with example from ecaalyx. *Biomedical engineering online*, 10(1):24.

[12] Brock, M., Quigley, A., and Kristensson, P. O. (2018). Change blindness in proximity-aware mobile interfaces. In *Proc. of CHI Conf. on Human Factors in Computing Systems*, pages 1–7.

[13] Brudy, F., Holz, C., Rädle, R., Wu, C.-J., Houben, S., Klokmose, C. N., and Marquardt, N. (2019). Cross-device taxonomy: Survey, opportunities and challenges of interactions spanning across multiple devices. In *Proc. of CHI Conf. on Human Factors in Computing Systems*, pages 1–28.

[14] Brudy, F., Suwanwatcharachat, S., Zhang, W., Houben, S., and Marquardt, N. (2018). Eagleview: A video analysis tool for visualising and querying spatial interactions of people and devices. In *Proc. of Internat. Conf. on Interactive Surfaces and Spaces*, pages 61–72.

[15] Brzezinski, A., Deiana, G., Kecht, V., and Van Dijcke, D. (2020). The covid-19 pandemic: government vs. community action across the united states. *Covid Economics: Vetted and Real-Time Papers*, 7:115–156.

[16] Cardenas, C. and Garcia-Macias, J. A. (2017). Proximithings: Implementing proxemic interactions in the internet of things. *Procedia Computer Science*, 113:49–56.

[17] Carroll, J. M. and Long, J. (1991). *Designing interaction: Psychology at the human-computer interface*. CUP Archive.

[18] Chan, F. K. and Thong, J. Y. (2009). Acceptance of agile methodologies: A critical review and conceptual framework. *Decision support systems*, 46(4):803–814.

[19] Chao, C., Tan, J., Castillo, E. M., Zawaideh, M., Roberts, A., and Kinney, T. (2014). Comparative efficacy of new interfaces for intra-procedural imaging review: the microsoft kinect, hillcrest labs loop pointer, and the apple ipad. *Journal of digital imaging*, 27(4):463–469.

[20] Cheraghi, S. A., Namboodiri, V., and Walker, L. (2017). Guidebeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 121–130. IEEE.

[21] Cho, Y., Bianchi-Berthouze, N., Marquardt, N., and Julier, S. J. (2018). Deep thermal imaging: Proximate material type recognition in the wild through deep learning of spatial surface temperature patterns. In *Proc. of CHI Conf. on Human Factors in Computing Systems*, pages 1–13.

[22] Cobb, A. and Lazar, B. (2020). Mobile device usage contributes to nosocomial infections. *Radiologic Technology*, 91(3):303–307.

[23] Cronin, S. and Doherty, G. (2019). Touchless computer interfaces in hospitals: A review. *Health informatics journal*, 25(4):1325–1342.

[24] Cross, R. (2012). Red cross: Mobile apps for emergencies gain in popularity | mobihealthnews. http://www.mobihealthnews.com/18402/red-cross-mobile-apps-for-emergencies-gain-in-popularity.

[25] Data, U. M. (2018). Medical id bracelets. https://www.universalmedicaldata.com/.

[26] Debnath, T., Bhowmik, S., Islam, T., and Chowdhury, M. M. H. (2018). Presence of multidrug-resistant bacteria on mobile phones of healthcare workers accelerates the spread of nosocomial infection and regarded as a threat to public health in bangladesh. *Journal of microscopy and ultrastructure*, 6(3):165.

[27] Del Rosario, M. B., Redmond, S. J., and Lovell, N. H. (2015). Tracking the evolution of smartphone sensing for monitoring human movement. *Sensors*, 15(8):18901–18933.

[28] Desjardins, A., Neustaedter, C., Greenberg, S., and Wakkary, R. (2014). Collaboration surrounding beacon use during companion avalanche rescue. In *Proc. of ACM Conf. on Computer supported cooperative work & social computing*, pages 877–887.

[29] Developers, A. (2018). Android developers. https://developer.android.com/design/.

[30] Developers, G. (2109). Sensors overview. https://developer.android.com/guide/topics/sensors/sensors_overview#java.

[31] Dingler, T., Funk, M., and Alt, F. (2015). Interaction proxemics: Combining physical spaces for seamless gesture interaction. In *Proc. of Internat. Symposium on Pervasive Displays*, pages 107–114.

[32] Dostal, J., Hinrichs, U., Kristensson, P. O., and Quigley, A. (2014). Spidereyes: designing attention-and proximity-aware collaborative interfaces for wall-sized displays. In *Proc. of Internat. Conf. on Intelligent User Interfaces*, pages 143–152.

[33] Dostal, J., Kristensson, P. O., and Quigley, A. (2013). Multi-view proxemics: distance and position sensitive interaction. In *Proc. of ACM Internat. Symposium on Pervasive Displays*, pages 1–6.

[34] Engelhart, S., Fischnaller, E., Simon, A., Gebel, J., Büttgen, S., and Exner, M. (2008). Microbial contamination of computer user interfaces (keyboard, mouse) in a tertiary care centre under conditions of practice. *Hyg Med*, 33(12):504–7.

[35] Evans, G. W., Lepore, S. J., and Allen, K. M. (2000). Cross-cultural differences in tolerance for crowding: Fact or fiction? *Journal of Personality and Social Psychology*, 79(2):204.

[36] Fowler, M. (2010). *Domain-specific languages*. Pearson Education.

[37] Gallo, L., Placitelli, A. P., and Ciampi, M. (2011). Controller-free exploration of medical image data: Experiencing the kinect. In *Proc. of 24th international symposium on computer-based medical systems (CBMS)*, pages 1–6. IEEE.

[38] Gao, L., Waechter, K. A., and Bai, X. (2015). Understanding consumers' continuance intention towards mobile purchase: A theoretical framework and empirical study–a case of china. *Computers in Human Behavior*, 53:249–262.

[39] Garcia-Macias, J. A., Ramos, A. G., Hasimoto-Beltran, R., and Hernandez, S. E. P. (2019). Uasisi: a modular and adaptable wearable system to assist the visually impaired. *Procedia Computer Science*, 151:425–430.

[40] Giner, P., Cetina, C., Fons, J., and Pelechano, V. (2009). A framework for the reconfiguration of ubicomp systems. In *3rd Symposium of Ubiquitous Computing and Ambient Intelligence 2008*, pages 1–10. Springer.

[41] Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R., and Wang, M. (2011). Proxemic interactions: the new ubicomp? *Interactions*, 18(1):42–50.

[42] Grønbæk, J. E., Knudsen, M. S., O'Hara, K., Krogh, P. G., Vermeulen, J., and Petersen, M. G. (2020). Proxemics beyond proximity: Designing for flexible social interaction through cross-device interaction. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

[43] Grønbæk, J. E., Linding, C., Kromann, A., Jensen, T. F. H., and Petersen, M. G. (2019). Proxemics play: Exploring the interplay between mobile devices and interiors. In *Proc. of the Companion Publication on Designing Interactive Systems Conference*, pages 177–181.

[44] Grønbæk, J. E. and O'Hara, K. (2016). Built-in device orientation sensors for ad-hoc pairing and spatial awareness. In *Proc. of Cross-Surface Workshop*.

[45] Hall, E. T. (1963). A system for the notation of proxemic behavior. *American Anthropologist*, 65(5):1003–1026.

[46] Hall, E. T. (1966). *The Hidden Dimension: An anthropologist examines man's use of space in private and public*. New York: Anchor Books; Doubleday & Company, Inc.

[47] Ibrahim, R., Badr, H. I. B., and Ali, N. M. (2012). Mobile phones and nosocomial infections. *Int J Infect Control*, 8:1–5.

[48] Jaimes, A. and Sebe, N. (2005). Multimodal human computer interaction: A survey. In *International Workshop on Human-Computer Interaction*, pages 1–15. Springer.

[49] Jalaliniya, S., Smith, J., Sousa, M., Büthe, L., and Pederson, T. (2013). Touch-less interaction with medical images using hand & foot gestures. In *Proc. of ACM conference on Pervasive and ubiquitous computing adjunct publication*, pages 1265–1274.

[50] Jeanne, V., Jegaden, F.-X., Kleihorst, R., Danilin, A., and Schueler, B. (2006). Real-time face detection on a dual-sensor smart camera using smooth-edges technique. In *International Workshop on Distributed Smart Cameras (DSC 06)*, volume 31.

[51] Juhnke, B. J. (2013). Evaluating the microsoft kinect compared to the mouse as an effective interaction device for medical imaging manipulations, iowa state university digital. *Dissertation thesis*.

[52] Karabay, O., Koçoglu, E., Tahtaci, M., et al. (2007). The role of mobile phones in the spread of bacteria associated with nosocomial infections. *J Infect Dev Ctries*, 1(1):72–73.

[53] Karray, F., Alemzadeh, M., Saleh, J. A., and Arab, M. N. (2008). Human-computer interaction: Overview on state of the art. *INTERNATIONAL JOURNAL ON SMART SENSING AND INTELLIGENT SYSTEMS*, 1.

[54] Kim, H.-J., Kim, J.-W., and Nam, T.-J. (2016). ministudio: Designers' tool for prototyping ubicomp space with interactive miniature. In *Proc. of CHI Conf. on Human Factors in Computing Systems*, pages 213–224.

[55] Kim, Y., Leonard, S., Shademan, A., Krieger, A., and Kim, P. (2014). Kinect technology for hand tracking control of surgical robots: technical and surgical skill comparison to current robotic masters. *Surgical endoscopy*, 28(6):1993–2000.

[56] Kissler, S. M., Tedijanto, C., Goldstein, E., Grad, Y. H., and Lipsitch, M. (2020a). Projecting the transmission dynamics of sars-cov-2 through the postpandemic period. *Science*, 368(6493):860–868.

[57] Kissler, S. M., Tedijanto, C., Lipsitch, M., and Grad, Y. (2020b). Social distancing strategies for curbing the covid-19 epidemic.

[58] Lalanda, P., Nigay, L., and Martinet, M. (2014). Multimodal interactions in dynamic, service-oriented pervasive environments. In *2014 IEEE International Conference on Services Computing*, pages 251–258. IEEE.

[59] Ledo, D., Greenberg, S., Marquardt, N., and Boring, S. (2015). Proxemic-aware controls: Designing remote controls for ubiquitous computing ecologies. In *Proc. of Internat. Conf. on Human-Computer Interaction with Mobile Devices and Services*, pages 187–198.

[60] Ledo, D., Houben, S., Vermeulen, J., Marquardt, N., Oehlberg, L., and Greenberg, S. (2018). Evaluation strategies for hci toolkit research. In *Proc. of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–17.

[61] Leyvand, T., Meekhof, C., Wei, Y.-C., Sun, J., and Guo, B. (2011). Kinect identity: Technology and experience. *Computer*, 44(4):94–96.

[62] Ling, L., Wong, W., Wan, W., Choi, G., and Joynt, G. (2020). Infection control in non-clinical areas during the covid-19 pandemic. *Anaesthesia*, 75(7):962–963.

[63] Lyytinen, K. and Yoo, Y. (2002). Ubiquitous computing. *Communications of the ACM*, 45(12):63–96.

[64] Marquardt, N., Diaz-Marino, R., Boring, S., and Greenberg, S. (2011). The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proc. of Symposium on User Interface Software and Technology*, pages 315–326.

[65] Marquardt, N. and Greenberg, S. (2015). Proxemic interactions: From theory to practice. *Synthesis Lectures on Human-Centered Informatics*, 8(1):1–199.

[66] Marquardt, N., Hinckley, K., and Greenberg, S. (2012). Cross-device interaction via micro-mobility and f-formations. In *Proc. of Symposium on User interface software and technology*, pages 13–22.

[67] Mentis, H. M., O'Hara, K., Sellen, A., and Trivedi, R. (2012). Interaction proxemics and image use in neurosurgery. In *Proc. of the CHI Conf. on Human Factors in Computing Systems*, pages 927–936.

[68] Microsoft (2018). Kinect identity: Player recognition in xbox - microsoft research. https://www.microsoft.com/en-us/research/video/kinect-identity-player-recognition-in-xbox.

[69] Microsoft (2109). Cameraspacepoint structure | microsoft docs. https://docs.microsoft.com/en-us/previous-versions/windows/kinect/dn758354(v%3Dieb.10).

[70] Mojgan, G., Marvin, P., Wong, C., Wallace, J. R., and Scott, S. D. (2018). Increasing passersby engagement with public large interactive displays: A study of proxemics and conation. In *Proc. of Internat. Conf. on Human Factors on Computing System*, pages 1–14.

[71] Mostafa, A. E., Greenberg, S., Vital Brazil, E., Sharlin, E., and Sousa, M. C. (2013). Interacting with microseismic visualizations. In *Proc. of CHI Extended Abstracts on Human Factors in Computing Systems*, pages 1749–1754.

[72] Myers, B. A. (1998). A brief history of human-computer interaction technology. *interactions*, 5(2):44–54.

[73] Nardi, B. A. (1996). Activity theory and human-computer interaction. *Context and consciousness: Activity theory and human-computer interaction*, 436:7–16.

[74] Oracle (2019). Using the jaxb class generator and jaxb users guide.

[75] Pellegrino, L. (2018). Medical id - the android app that could save your life! https://medicalid.app/.

[76] Rector, K., Salmon, K., Thornton, D., Joshi, N., and Morris, M. R. (2017). Eyes-free art: Exploring proxemic audio interfaces for blind and low vision art engagement. *Proc. of Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):93.

[77] Roussel, N., Evans, H., and Hansen, H. (2004). Mirrorspace: using proximity as an interface to video-mediated communication. In *Proc. of Internat. Conf. on Pervasive Computing*, pages 345–350.

[78] Ruppert, G. C., Reis, L. O., A., P. H. J., de Moraes, T. F., and da Silva, J. V. L. (2012). Touchless gesture user interface for interactive image visualization in urological surgery. *World journal of urology*, 30(5):687–691.

[79] Schmidt, A. (2000). Implicit human computer interaction through context. *Personal technologies*, 4(2):191–199.

[80] Scialdone, M. J. and Connolly, A. J. (2020). How to teach information systems students to design better user interfaces through paper prototyping. *Journal of Information Systems Education*, 31(3):179.

[81] Sørensen, H., Kristensen, M. G., Kjeldskov, J., and Skov, M. B. (2013). Proxemic interaction in a multi-room music system. In *Proc. of Australian Computer-Human Interaction Conf.: Augmentation, Application, Innovation, Collaboration*, pages 153–162.

[82] Soucy, J. R., Sturrock, S. L., Berry, I., Daneman, N., MacFadden, D. R., and Brown, K. A. (2020). Estimating the effect of physical distancing on the covid-19 pandemic using an urban mobility index. *medRxiv*.

[83] Thomas-Rüddel, D., Winning, J., Dickmann, P., Ouart, D., Kortgen, A., Janssens, U., and Bauer, M. (2020). Coronavirus disease 2019 (covid-19): update for anesthesiologists and intensivists march 2020. *Der Anaesthesist*, pages 1–10.

[84] Van Deursen, A., Klint, P., and Visser, J. (2000). Domain-specific languages: An annotated bibliography. *ACM Sigplan Notices*, 35(6):26–36.

[85] Vermeulen, J., Luyten, K., Coninx, K., Marquardt, N., and Bird, J. (2015). Proxemic flow: Dynamic peripheral floor visualizations for revealing and mediating large surface interactions. In *Proc. of Human-Computer Interaction*, pages 107–114.

[86] vicon (2108). About vicon motion systems. https://www.vicon.com/vicon/about.

[87] Visser, E. (2007). Webdsl: A case study in domain-specific language engineering, generative and transformational techniques in software engineering ii: International summer school, gttse 2007, braga, portugal, july 2-7, 2007. revised papers.

[88] Vogel, D. and Balakrishnan, R. (2004). Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 137–146.

[89] Vukkadala, N., Qian, Z. J., Holsinger, F. C., Patel, Z. M., and Rosenthal, E. (2020). Covid-19 and the otolaryngologist: preliminary evidence-based review. *The Laryngoscope*.

[90] Wang, M., Boring, S., and Greenberg, S. (2012). Proxemic peddler: a public advertising display that captures and preserves the attention of a passerby. In *Proc. of Internat. Symposium on Pervasive Displays*, pages 1–7.

[91] Wang, Y., Wang, Y., and Chen, Y.and Qin, Q. (2020). Unique epidemiological and clinical features of the emerging 2019 novel coronavirus pneumonia (covid-19) implicate special control measures. *Journal of medical virology*, 92(6):568–576.

[92] Wang, Y., Yang, X., Zhao, Y., Liu, Y., and Cuthbert, L. (2013). Bluetooth positioning using rssi and triangulation methods. In *Proc. of Consumer Communications and Networking Conference*, pages 837–842.

[93] Weiser, M. (1991). The computer for the 21 st century. *Scientific american*, 265(3):94–105.

[94] Wolf, K., Abdelrahman, Y., Kubitza, T., and Schmidt, A. (2016). Proxemic zones of exhibits and their manipulation using floor projection. In *Proc. of ACM Internat. Symposium on Pervasive Displays*, pages 33–37.

[95] Zhang, Z. (2012). Microsoft kinect sensor and its effect. *IEEE multimedia*, 19(2):4–10.

[96] Zidek, A., Tailor, S., and Harle, R. (2018). Bellrock: Anonymous proximity beacons from personal devices. In *In Proc. of International Conference on Pervasive Computing and Communications*, PerCom'18, pages 1–10. IEEE.