TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

ALFONSO CAMBERO LORENZO

CONFIDENTIALITY IN A MOBILE LOCATION BASED SERVICE

Master of Science Thesis

Examiners: Mikko Tiusanen,
Marko Helenius
Examiners and topic approved by the Council of the Faculty of Computing and Electrical Engineering on 4th June 2014

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master’s Degree Programme in Telecommunications (Universidad Politécnica de Madrid)
CAMBERO LORENZO, ALFONSO: Confidentiality in a mobile location based service
Master of Science Thesis, 42 pages, 4 Appendix pages
July 2014
Major: Computer Science, electronics, and communications
Examiners: Professor Mikko Tiusanen, Research fellow Marko Helenius
Keywords: Location Based Service, confidentiality, data security, privacy, data flow, mobile computing.

This thesis tackles the current issue of privacy in the growing mobile location-based applications, which provide an added value to users using their location. These location-based applications need a specific study of privacy attending to tracking, in order guarantee user right of privacy.

This thesis is divided into two parts. The first discusses privacy and data security. In order to link terms data security and privacy, and overcome the differences between both fields, confidentiality is introduced. Data security ensures confidentiality, which guarantees user privacy. Since privacy is a social phenomenon, influenced by society, it will be incorporated through a survey of user opinion. Also, legislation about privacy needs to be taken into account.

In the constructive part, a location-based application is built, in order to demonstrate how considering interactions of privacy, confidentiality, and data security influence a real application. A survey offered to potential users is used to define the level of privacy expected. Finally, the system is evaluated against confidentiality level established by privacy level from the survey results. In addition, this application will provide to society a tool to request help from people near the user, creating a social network based on trust in other users.

PREFACE

Since I was a child, technology has been present in my life. This thesis culminates everything I have learnt during my twenty-three years, from the primary school where I started to love mathematics and physics, to the university where I'm finishing my post-secondary education as a telecommunication engineer.

It has not been easy reaching my goal but I have been lucky because of the unconditional support of my parents, Santiago and Fabiola, and my sisters, Inés and María. I would like to remember with nostalgia and affection my grandparents, Antonio and Tila, who always trusted me. I thank my aunts, Chus and Carmen, my uncle, Toñín, my cousins, Beatriz and Patricia, and especially to my cousin Borja, for their encouragement.

I also want to mention my university classmates with whom I have spent the last five years of my academic preparation, my friends from Coria and Madrid, and chiefly Elena who always backed me when I have needed it. Moreover, to all the friends that I have met in Tampere and who I have shared my Erasmus experience with.

Finally, I am very grateful to Mikko Tiusanen for his patience and dedication in this thesis, and for his hospitality during my stay in Finland. Furthermore, I thank Marko Helenius for his contribution in this work.

Tampere, July 25, 2014

Alfonso Cambero Lorenzo.

TABLE OF CONTENTS

TERMS AND DEFINITIONS

API Application Programming Interface, specifies how some software component should interact with others.

App Mobile application.

CDF Cumulative Distribution Function (Zwillinger & Kokoska 2010).

CGI Common Gateway Interface, standard method used to generate dynamic content on web pages and web applications. (W3C 2011).

Ciphertext Unreadable output data of an encryption algorithm.

GCM Google Cloud Messaging for Android, service that allows sending and receiving data between a server and an Android device.

GDT Google Device Token, unique identifying number provided by GCM services to the device and used to send message to it by GCM.

GPS Global Positioning System, a satellite-based navigation system (National Research Council 1995).

GSM Global System for Mobile Communications, standard to describe protocols for second generation (2G) digital cellular networks used by mobile phones.

HashMap Java HashMap, data structure used to implement an associative array, a structure that can map key strings to values.

HTTP Hypertext Transfer Protocol is an application-level protocol for distributed, collaborative, hypermedia information systems (Fielding et al 1999).

HTTPS Hypertext Transfer Protocol Secure, an application-level protocol for distributed, collaborative, hypermedia information systems over a secure network (Rescorla & RTFM 2000).

IP Internet Protocol, the communication protocol that provides an identification and location system for computers on networks and routes traffic across the Internet (Deering & Hinden 1998).

IPv6 Internet Protocol version 6, the latest version of IP.

LBS Location Based Service.

OS Operating System.

PDF Probability Density Function (Parzen 1962).

Plaintext Input data to an encryption algorithm.

Pull Client initiates communication with server, common in web programming.

Push Server initiates communication to a client, less common in web programming.

SDK Software Development Kit, tools that help create applications for a certain software package or framework.

UML Unified Modeling Language (OMG 2014).

URL Uniform Resource Locator, string used to identify a resource.

VoIP Voice-over-Internet Protocol, a way to carry phone calls over an IP data network (Cisco 2008).

1. INTRODUCTION

At the time of smartphones and social networks, location based applications start to be more significant, since users expect to receive custom services that know their position. Location based services (LBSs) are expected to form an important part of the future computing environments that will seamlessly and ubiquitously integrate into our life (National Research Council 2003).

With telecommunications and smartphones, with many social networks and tracking apps where people share their current location, privacy is coming to focus. Society is concerned about privacy issues, because keeping your information private is harder with the Internet and the sharing society, where everybody publish pictures, location, opinions, on the web. Recently, social networks, such as Facebook and Twitter, have been in the focus of public attention. People request privacy, but what is privacy when you are voluntarily sharing your own information? How to provide a high level of privacy in this case? The designer needs to understand what level of privacy users could expect in LBSs.

In this thesis, confidentiality will be used as a bridge between the social concept of privacy and the technological one of data security, in order to achieve privacy goals in an LBS. An example location-based application, *Help Button*, has been used to highlight how confidentiality links privacy and data security. The *Help Button* system provides a tool to connect users in order to have a help community for old people or people with health problems. It will allow a quick way to request help and receive it from some volunteer user.

This thesis is structured in two main parts. First one is focused on LBSs, privacy and how to achieve the privacy goal. Second one is the development and evaluation of the app. First part introduces the problem definition of this thesis. It continues by explaining LBSs, and analysing privacy and data security. Second part describes briefly what is the Android operating system in order to know its features included in the app. Next, the app implementation is explained, and the system is evaluated according to privacy and data security.

2. CONFIDENTIALITY

Privacy is everyone’s right to deny access to something personal, data in this thesis. On the other hand, data security technology provides tools to develop the system. Between the technological tools and privacy is confidentiality, which is in charge of guaranteeing that data is only available to those authorized. Also, legislation needs to be taken into account.

There is a relation between privacy and data security, but it is not direct. As we can see in Figure 2.1., both are related by confidentiality.

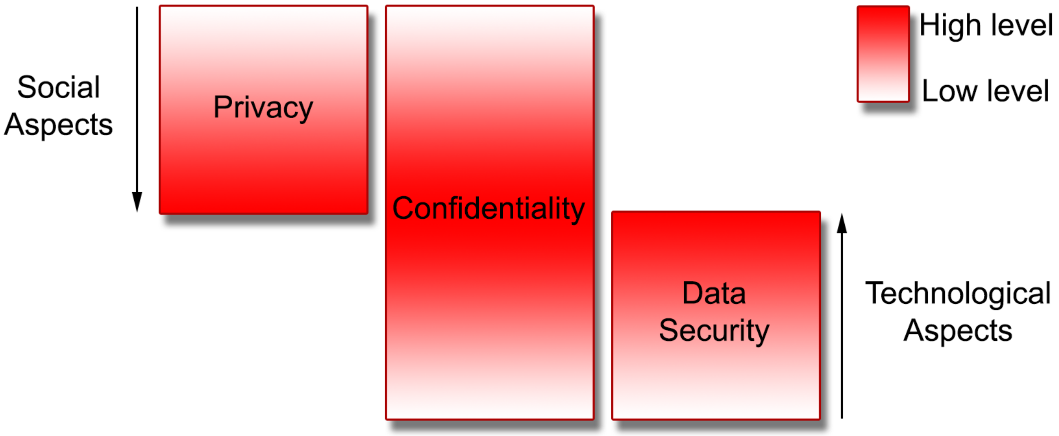


Figure 2.1 *Privacy, confidentiality, and data security relation.*

Confidentiality is understood as the treatment of information that an individual discloses in a relationship of trust and with the expectation that it will not be divulged to others without permission in ways that are inconsistent with the understanding of the original disclosure (University of California at Irvine 2014). Confidentiality is a non-innate information feature, that is, provided by the person or system that receives it to the information owner.

Balancing data integrity, availability and confidentiality is considered the way to ensure data privacy in a system (Olivier 2002). Data integrity ensures that information is not changed or corrupted, e.g., due to hardware failure, technical problem, or human error. Guaranteeing data integrity is difficult, restoring it is even more. Data should be available to those that are authorized for it when they need it. Many factors influence data availability: confidentiality, equipment, network capacity, and user activity.

In order to get integrity and availability, backing up data is essential. Creating copies of the data, system can confront some corrupted data since there is a backup to restore it, assuring integrity, since data is the same as the original, and availability, because data is accessed even if the original is not available. Backups bring new data issues. Backups must be protected as the original data is since both contain confidential data.

Integrity comprises also authenticity and accuracy. Authenticity checks data source and data content in order to ensure that source is authorized and data is the proper. Accuracy checks that the data has not been tampered with and that the source is reliable avoiding threats based on data covered up or sources involved in non-secure surroundings.

Data security, as a technological tool, can be used to achieve confidential handling of data. To obtain a high level of confidentiality, the system must be designed to attain this. The more correctly security mechanisms are used in the system protection, the more confidential is it.

On the other hand, we can relate privacy and confidentiality, since the higher system confidentiality is, the higher privacy provided by the system is. Society establishes expected level of privacy, setting how well protected information must be, in order to restrict access to unauthorized users. Confidentiality resides in the app design, since it depends on how data, and therefore information, is handled. Given that, society does not modify confidentiality level, technology does.

This thesis considers controlling access to personal data in a mobile location based service (LBS). User location, the main information that an LBS uses, is sensitive data, the handling of which directly influences user privacy. The aim of this work is to achieve privacy of the user location through confidentiality. We shall use a social network application, *Help Button,* to see how data security and privacy relate to the confidentiality of user location in such a system.

Each kind of information needs specific technological tools. In this work, data security tools for LBSs will be used, trying to achieve a level of higher confidentiality in such a system, in order to get the higher privacy level.

System evaluation will be done from a technical point of view, analysing data security side, checking the system's confidentiality. Next, privacy will be explored from a social point of view, achieving confidentiality level required for the privacy expected. System will be scored considered on confidentiality, omitting functionality or user opinion about its utility.

Access control, flow controls, inference controls, and cryptographic controls make up data security (Denning & Denning 1979), criteria for data system evaluation, here. Data security will be evaluated according to the system code implementation, avoiding physical security of the system, which will not be taken into account, since is not a goal in this thesis. Users will provide privacy evaluation through a survey. Data obtained from it will be analysed in order to get the privacy level expected by *Help Button* users.

3. LOCATION BASED SERVICES

Mobile location-based services are a mobile device feature, where using location provides an added value (Schiller & Voisard 2004). LBS refers to any system that takes into account the geographic location of an entity, the object triggering location information. It can be human or non-human (Junglas & Watson 2008).

In an LBS, there are always at least two entities involved, just like there are at least two people in a conversation. One of the entities' locations can be considered relative to the other. Moreover, an entity can be static or dynamic, static when it does not move, such as a shop; dynamic when it does, although it need not, such as a car. One of the entities is always the object of the LBS, the location of which is recorded. One of the entities is the receiver of the localization information.

There are two different tracking services: location-tracking and position-aware services. Location tracking provides the added value using the user location, and sending it to an external system, a third party. Position-aware supplies the location information to the user (the information requester), perhaps without any third party being involved in the system.

Moreover, LBSs can be classified into person-oriented and device-oriented services. The first one is focused on the position of a person to enhance a service and the user can control the service, e.g., a map of pictures. Device-oriented services are external to the user, e.g., child tracking. Service can locate people, animals, or objects that hold the device. Usually, the user does not control the service.

Another classification is according to the nature of the service: push and pull services. Push services provide information without the user having to actively request it, e.g., an indoor proximity system that provides advertisements to users when they enter into a store. Pull services, in contrast, mean that user has to request the information, e.g., bus tracking.

In order to provide a location service, devices implement several technologies to get location data. Since 1989, when the US government published the GPS technology, this has been used for location. However, in the late 1990s new technological advances in mobile networks allowed using these in order to get the user location.

GPS accuracy, as that of Galileo (ESA 2013) accuracy, is very high, around 3–4 meters. This feature provides a high level of added value to LBSs. On the other hand, mobile network accuracy can be around 100 meters in urban areas, but only up to 3 kilometer accuracy in rural areas. Accuracy level limits the areas where the LBSs can be used, and the quality of the service. A list of accuracy levels for different applications is shown in Table 3.1.

Table 3.1. *Overview of LBS applications and level of accuracy required (Bellocci et al. 2002).*

Application	Accuracy	Tracking service
News	Low	Location-tracking
Directions	High	Position-aware
Car Navigation	Medium to High	Position-aware
Emergency	High	Location-tracking
Child Tracking	Medium to High	Location-tracking
Personal Navigation	High	Position-aware
Location-Sensitive Billing	Medium to Low	Position-aware

4. PRIVACY

To guarantee user's rights of privacy, from a social point of view, systems should control data in a confidential way. European directives will be taken into account, since European countries have to follow them in their own legislations.

European commission states that European countries must protect the rights and freedoms of persons with respect to the processing of personal data by automated means, e.g., a computer database of customers, through laws and directives (1995). LBSs are affected by this directive since the system should know the user location. Related to this feature, directive specifies a data subject's right of access to data: every data subject should have the right to obtain a confirmation from the controller as to whether or not data relating to him or her are being processed, and to be notified of the data undergoing processing, and be allowed rectification, erasure, or blocking of data. Also, it specifies the right to object to the processing of data.

(European commission 2002) describes a specific directive about LBSs. It regulates the electronic communications sector. This directive concerns processing security, confidentiality, and data retention. It regulates LBSs about location data (Article 9):

- "Where location data other than traffic data, relating to users or subscribers of public communications networks or publicly available electronic communications services, can be processed, such data may only be processed when it is made anonymous or with the consent of the user for the duration necessary for the provision of a service."

  This extract from the directive states that a service must ask to user about the location usage before doing it, or the data obtained from an user must be handled so that nobody would be able to know who is the user providing the data.

- "The service provider must inform the users or subscribers, prior to obtaining their consent, of the type of location data other than traffic data which will be processed, of the purpose and duration of the processing, and whether the data will be transmitted to a third party".

  This establishes the needed of Terms of Service in order to inform users about everything related to location data.

- "Users or subscribers shall be given the possibility to withdraw their consent for the processing of location data at any time"

  This lays down that users have the right to decide when the service must stop to collect data from them.

- "(…) the users or subscribers must continue to have the possibility, using a simple means and free of charge, of temporarily refusing the processing of such data for each connection to the network or for each transmission of a communication".

  According to this statement, mobile LBSs must provide, say, an easily accessible button to disable location service.

4.1 LBS privacy

LBS privacy is based on anonymous tracking. If the LBS does not use a login system, thus, maintaining users' anonymity, the privacy of the system users is guaranteed. However, (Gruteser & Grunwals 2003; Kalnis et al. 2006) showed that simply dropping the issuer's personal identification data may not be sufficient to make the request anonymous. For example, a person uses everyday the same bus to go to work. To optimize the time, this user uses every morning a LBS that minimizes waiting time at the bus stop. In the evening, this user opens the LBS again from the office, in order to check the bus to come back home. Although system does not have any information about the user, with the help of external knowledge about the location of certain user, he or she could be tracked.

To avoid this kind of issues, a solution could be using spatial generalization algorithms. The idea of this technique is generalize the location (and time), e.g., defining areas instead longitude and latitude, information contained in a LBS request (Mascetti & Bettini 2007).

Another solution to protect LBS privacy could be using false position data (dummies) at the same time that the real position data is used. This technique inundates the system with fake locations that protect users' privacy making it impossible to detect the real one (Kido et al. 2005).

In addition, concerning social aspects, LBSs are maintained by people, which could raise new privacy issues. To avoid this kind of privacy threats from inside the service, protocols should be designed to minimize the human operation, increasing confidentiality; training about privacy and confidentiality should also be given to system operators.

4.2 Good privacy practices in LBSs

Most privacy concerns appear when personal location data are made available to third parties other than the mobile phone operator (Fisher & Dobson 2003), since operators can obtain a location from cellular connection signal. This carries risks, but also some advantages are achieved. Because privacy is highly influenced by social aspects, in general, accepting the sharing of location data depends on the third party. Data sharing is more acceptable when the third party has rights or responsibilities in relation to the person being tracked, e.g., tracking your own children, or when the person is using the LBS knowing which data is being shared.

LBSs could be described as a new form of slavery based upon location control, named Geoslavery, contravening Article 4 of the Universal Declaration of Human Rights (Fisher & Dobson 2003; United Nations 1948). To safeguard user rights and avoid this kind of privacy issues, some good practices should be used in LBSs in order to guarantee privacy social aspects of:

- Programs should collect the minimum amount of personally identifiable information necessary.
- Programs should have strong policies to protect the privacy and security of personally identifiable data.
- Data collection and use policies should reflect respect for the rights of individuals and community groups and minimize undue burden.
- Programs should have policies and procedures to ensure the quality of any data they collect or use.
- Programs have the obligation to use and disseminate summary data to relevant stakeholders in a timely manner.
- System should minimize the number of persons and entities granted access to identifiable data.
- Program officials should be responsible stewards.
- Location data should never be shared with a third party if the added value does not depend on it, e.g., running tracker does not depend on location sharing; children tracking app depends on data sharing with parents. Sharing data creates new privacy issues.
- Legislation must be followed. They safeguard citizen rights.

5. DATA SECURITY

Data security involves basically four kinds of safeguards, each related but distinct (Denning & Denning 1979). These safeguards are access, flow, inference, and cryptographic controls. Briefly, access controls regulate which user may enter the system and which not, flow controls restrict amount of data sets accessible to a user, inference controls protect databases, and data encryption prevents unauthorized disclosing of confidential information in transit or in storage.

Confidentiality and, therefore, privacy are achieved through data security. Data must be stored ensuring data confidentiality, integrity, and availability. (Harauz et al. 2009)

5.1 Access control

Access controls restrict access to the data by unauthorized users. Confidential systems must ensure that everyone accessing data has the privilege to do it. Access control relies on and coexists with other security services in a computer system (Figure 5.1; Sandhu & Samarati 1994). This process includes the stages authorization, authentication, and audit.

Authorization defines a subjects' rights to access a system or content. Modern operating systems (OSs) implement authorization policies as formal set of permissions. Three main types of access are: read (R), write (W), and execute (X). In Figure 5.1, authorization stage is done in the authorization database, checking user rights, and notifying them to the reference monitor.

Downes states that authentication stage includes, additionally, identification. Identification is the act of claiming identity, a set of one or more signs defining a distinct entity. Authentication is the act of verifying that identity, where verification consists in establishing, to the satisfaction of the verifier, that the sign signifies the entity (2005). Both processes verify that an identity is bound to the entity that makes an assertion or claim of identity. In the system of Figure 5.1, identification is done when the user introduces his or her system credentials, and these are sent to the reference monitor in order to authenticate them.

Audits are an important part of the access control in order to guarantee security. Audit is an *a posteriori* analysis of all the requests and activities of users in the system (Sandhu & Samarati 1994). Audits are important to detect security violations and recreating security incidents in order to improve the system security. Figure 5.1. shows that audit covers all the system, controlling all the processes in this.

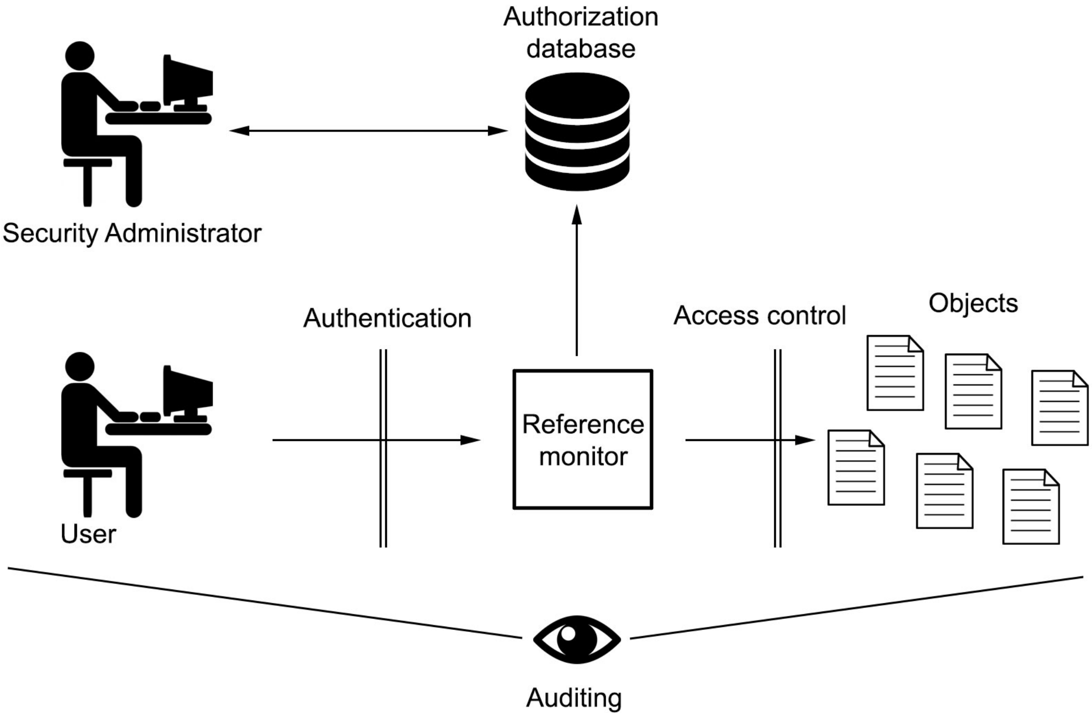


Figure 5.1. *Access control and other security services (Sandhu & Samarati 1994).*

5.2 Flow control

Flow control involves security in a data flow. A data flow occurs when a source sends some data to a destination. A flow policy specifies the channels along which information is allowed to move. In addition, this policy specifies the destinations allowed to receive data. Authorization and authentication stages are carried out in flow control.

Figure 5.2. illustrates a flow control scenario, where data is transfer by secure channels, and all the process is monitored.

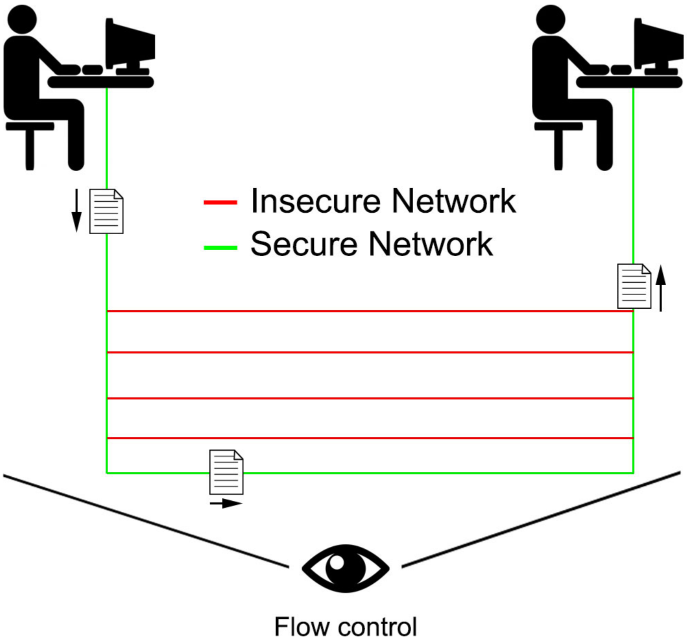


Figure 5.2. *Data flow scenario.*

5.3 Inference control

Inference controls are needed in databases, which contain sensitive information about individuals. The problem is that someone could get some of the information and reconstruct the original data through deduction, which compromises user privacy. This happens, for example, when a database about health information must be declassified for wider distribution, or when a LBS shares location data, suspending flow control policies.

Special database access queries avoid this kind of problem. One technique is the control on query set sizes and overlaps. Others are the rounding and error inoculation. Also queries based on random samples are used to solve this defect (Jagannathan & Wright 2007).

5.4 Cryptographic control

Cryptographic controls protect information stored or transmitted on insecure media, preventing the extraction of information by unauthorized parties from messages transmitted over a public channel (Diffie & Hellman 1976).

Encrypting data improves system confidentiality, but sharing a key, which no one else knows, between sender and receiver is necessary. This key can be sent by a secure channel, such as conventional cryptographic does, or using a pair of keys can be involved in the system being one of them known by everybody, public key, and the other only by the receiver, private key. Following introduces these two encryption systems.

Conventional cryptography is based on an encryption-shared key (K) that source sends, using a secure channel, to the receiver. Source converts a plaintext or unenciphered message (P) to ciphertext (C) using the encryption key, and sends it to receiver through any channel. Receiver uses the key, received from the source, to convert the ciphertext to the original message. Message could not be send using the secure channel for reasons of capacity or delay. Figure 5.3. shows this process flow.

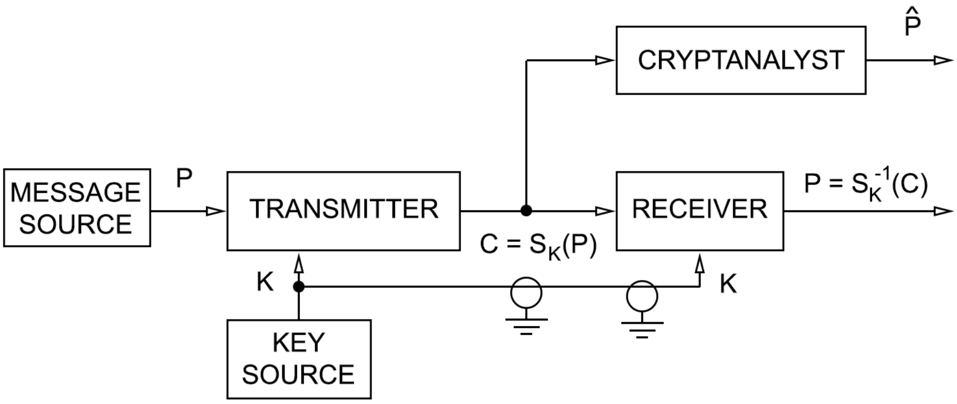


Figure 5.3. *Flow of information in conventional cryptographic system (Diffie & Hellman 1976).*

Conventional cryptographic systems imply using a secure channel that sometimes is not available. To avoid the key sharing, encryption system is modified in order to use a public-key (PBK) to convert plaintext (P) to ciphertext (C), and a private-key (PRK) to decrypt getting the original message. Decrypt is possible only using the private-key. Also, private-key can be used by the owner of it to encrypt some data, which public key can decrypt. Public-key is stored in some key storing system and this will provide it to everybody that request it, or the owner can send it through any channel since it is a public key. Figure 5.4. illustrates the data in a system that uses public and private keys to encrypt and decrypt.

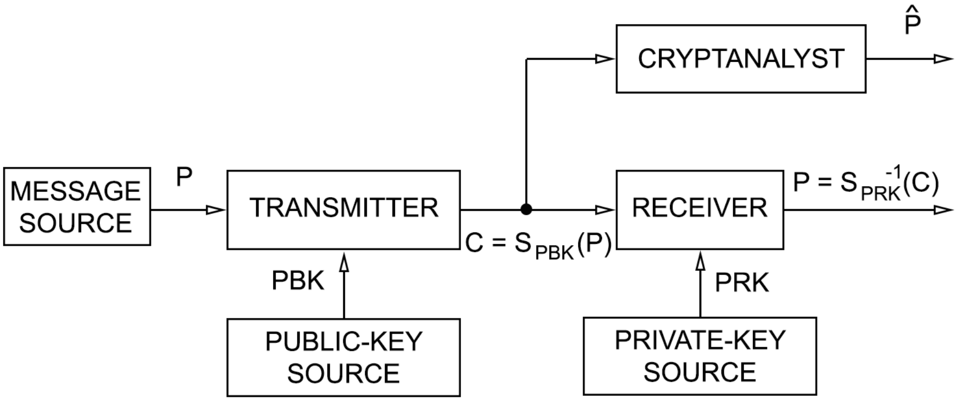


Figure 5.3. *Flow of information in a public cryptographic system (Diffie & Hellman 1976).*

At the present there are cryptographic algorithms, e.g., AES (NIST 2001) and Blowfish (Schneier 1993), and protocols, e.g., SSL (Freire et al. 2011) and TLS (Dierks & Allen 1999), developed to protect information in systems and some of them are commonly used in most of applications.

6. HELP BUTTON SYSTEM

Help Button System consists of two mobile apps developed for Android (2014a) using its native language, Java (2014). The system implements a Java web server that runs under an Apache Tomcat (2014) engine. Moreover, the system uses a PostgreSQL (2014) database used to register devices and to store information about help situations completed.

The first app, called Sender, consists of a “help button”. It has an Android widget with a button to send a help request to the system. Through this app users can inform about a help situation: the system will receive their location information and create a new help request. Users can, from a configuration interface, enter their personal information: name, health information, and picture. In addition, system provides the possibility of introducing phone numbers of known people. If there is no help available, these people will be phoned.

Second app, called Receiver, is for volunteers. People install this to receive help requests. This app is automatically opened, when a Sender presses the help button, if the receiver could assist the sender. If the receiver accepts, the app provides all the information about the sender location on a map and the provided personal information. If the receiver declines, Receiver app informs the server about this decision in order to continue with the help request, asking another receiver.

Server is in charge of linking both Android apps. Server receives the Receiver location regularly in order to have a database with the current Receiver locations to choose the best one when a help request is created. Additionally, the server stores all the information about the help request in the database and implements a time controlling system.

Both apps are currently available on Google Play. They can be downloaded from these URLs:

- **Sender:** https://play.google.com/store/apps/details?id=com.helpbutton.sender
- **Receiver:** https://play.google.com/store/apps/details?id=com.helpbutton.receiver

6.1 Use cases

The following use cases show different scenarios of the system according to the notation of Buhr and Casselman (1996). Use case maps for object oriented systems include four main blocks: Sender, Receiver, and Server. These blocks represent all the systems involved in a help situation. Plenty of senders and receivers are involved in the system, however, in the use cases these are omitted, including only the users involved in the task. The notation included in the following figures is presented in Table 6.1.

Table 6.1. *Use cases notation (Buhr and Casselman 1996).*

Symbol	Interpretation
———	**Path:** may have any shape as long as it is continuous. A path represents how data or signals flow in the system.
●	**Waiting place:** represent a starting point. In general, this means waiting for a stimulus to start the path.
▮	**Bar:** ends a path or marks a place where concurrent path segments begin or end.
◷	**Timer:** is a generalized waiting place that express the idea that there is a time limit on waiting.
□	**Box:** represent an entity involved in the system.

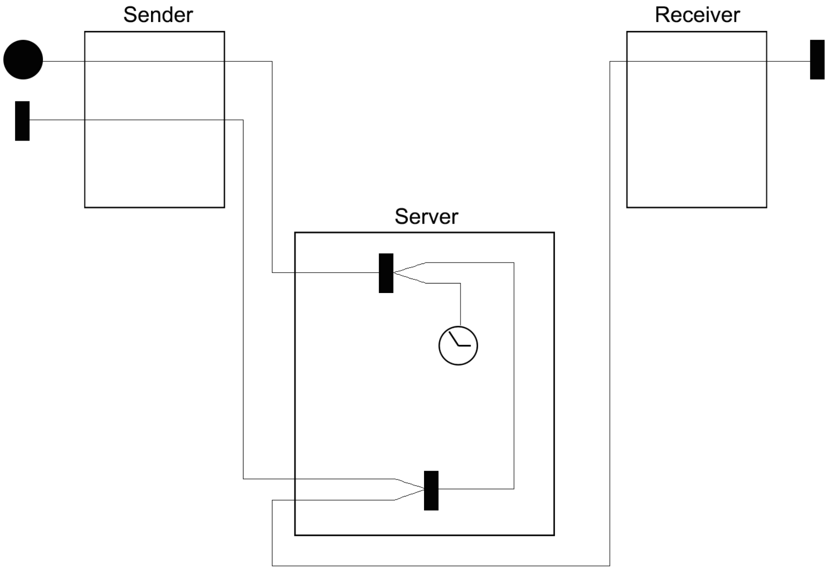


Figure 6.1. *Creating help situation use case.*

As Figure 6.1. shows, a Sender starts the new help situations. It sends the request to Server, which finds the best option for a Receiver and passes the request on. The request is sent to Receiver and the current state to Sender. Server also starts a timer to control answer time, in order to look for another Receiver if the current Receiver does not answer.

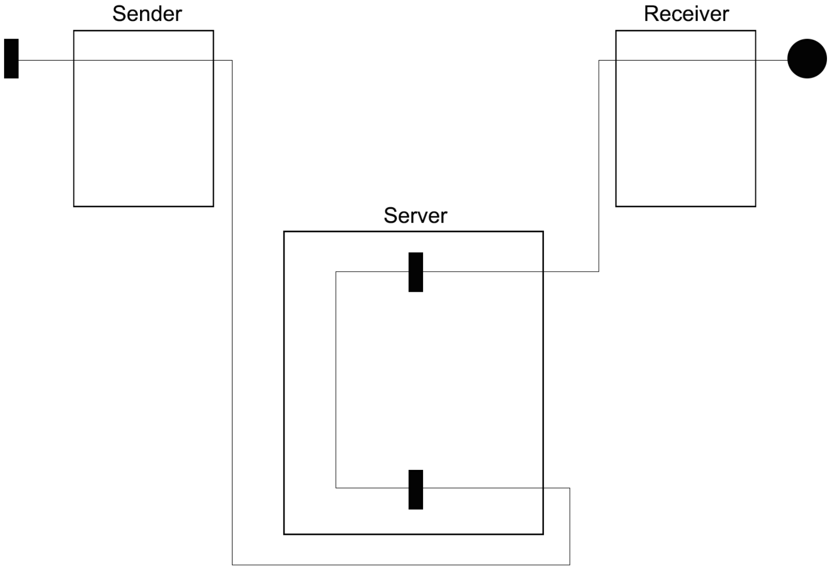


Figure 6.2. *Accepting help situation use case.*

Figure 6.2. illustrates how a Receiver accepts a help situation, after receiving the request in Figure 6.1., informing the Server. Server informs Sender about the current state of the help situation.

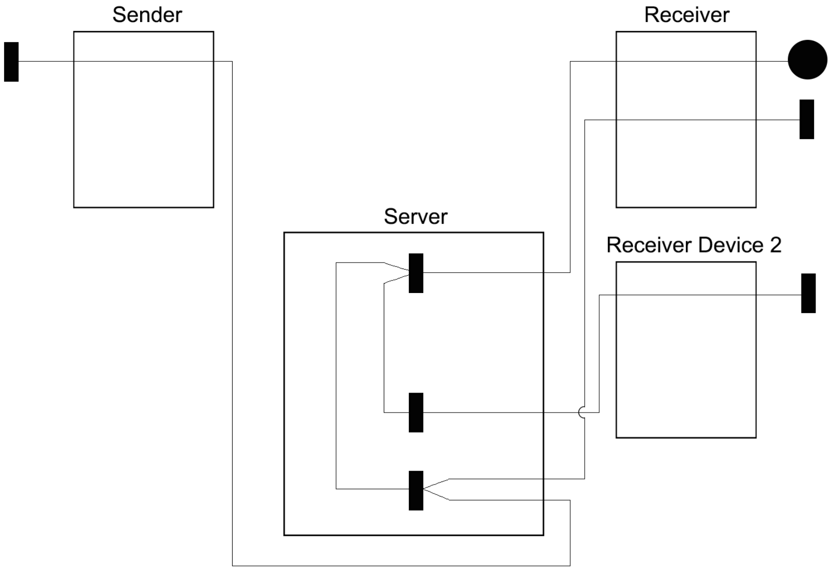


Figure 6.3. *Rejecting help situation use case.*

In Figure 6.3., after receiving a help situation request as Figure 6.1. illustrates, a Receiver can reject it sending the action to Server, which looks for another Receiver to attend to the help request. Server sends the confirmation to the Receiver rejecting the help request and including it in a list to don't ask again about the same request. When the help request is rejected, no data is shown, so it cannot be considered as a privacy issue and volunteer avoids responsibilities. In this situation, Sender is not informed, since the current situation is the same than the last, server is looking for a Receiver.

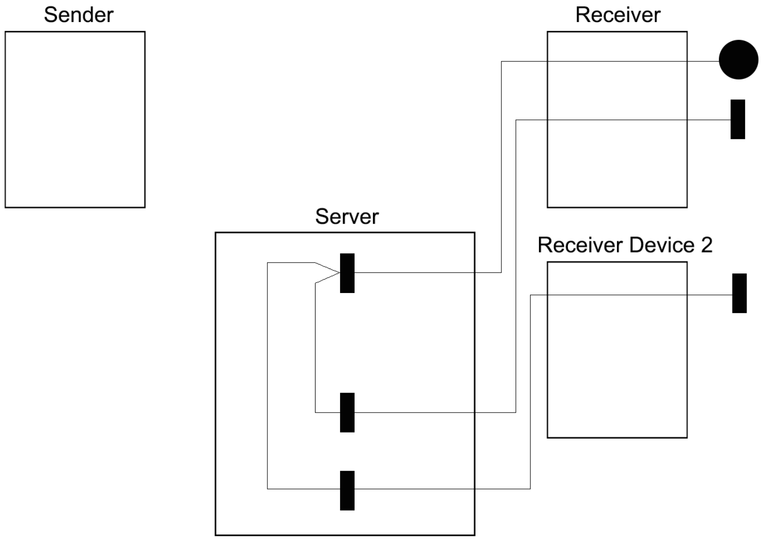


Figure 6.4. *Cancelling help situation use case from Receiver side.*

A Receiver cancels a help situation attempt, as Figure 6.4. shows, informing the Server. Server informs Sender about this event and looks for another receiver. When Server finds another Receiver, it sends the confirmation of the cancellation to the first one and the help situation request to the new one. The difference between reject and cancel is that cancel implies Receiver accepts the help request and sees requester data.

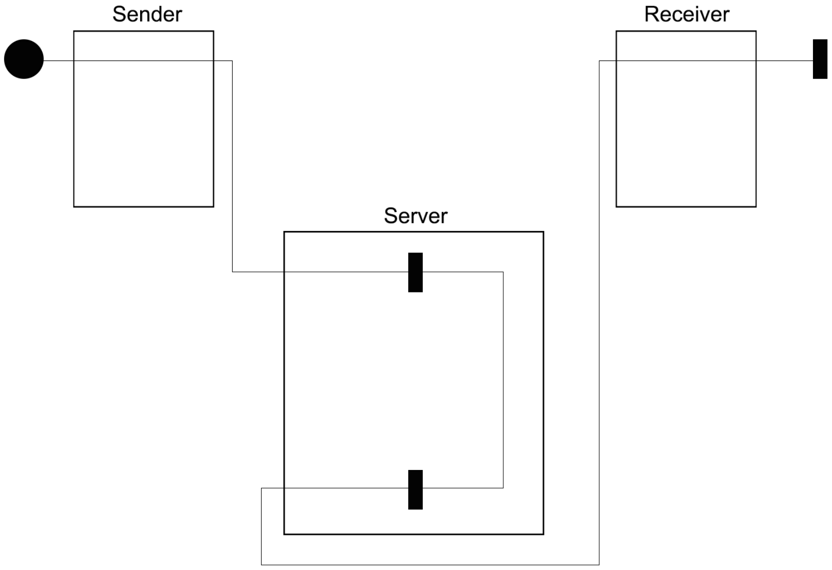


Figure 6.5. *Cancelling help situation use case from Sender side.*

A Sender cancels a help situation informing Server. Server informs Receiver about the end of the help situation. Receiver hides and eliminates the information on Sender from the receiver device. Figure 6.5. illustrates this process in a schematic way.

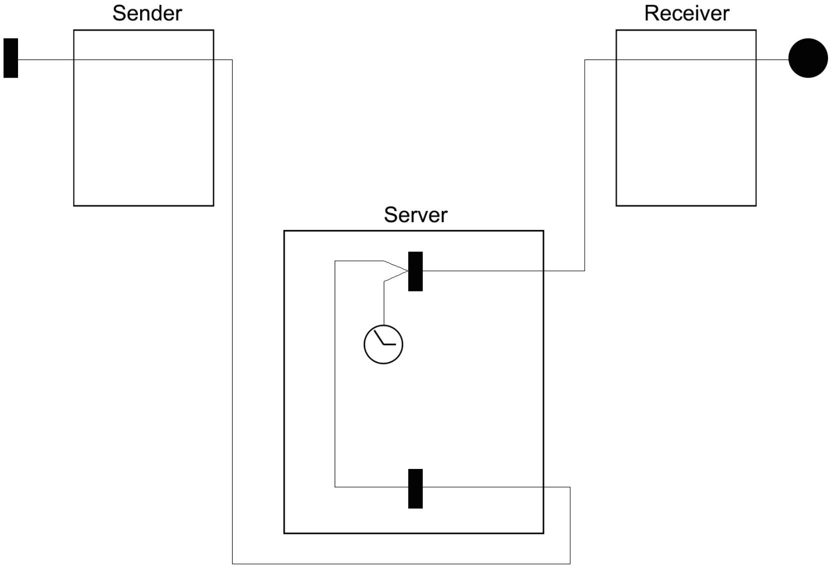


Figure 6.6. *Finishing help situation use case fron Reeiver side.*

Figure 6.6. shows the process to complete a help situation request, Receiver informing Server about this situation. Server notifies Sender in order to confirm the end of the help situation request. Server, when asksing Sender, starts a timer to restart the help situation request if Sender does not answer about the new state. Also, if Sender denies the end of the help situation, Server restarts it.

7. HELP BUTTON TECHNOLOGIES: ANDROID OS

Android is a mobile operating system based on the Linux kernel (Linux Foundation 2014) with a user interface. Source code is released by Google under an open source license and complete documentation is available on the Internet (Android 2014b).

The application of this thesis will be designed for an Android platform since the number of Android devices is the highest in the market. Figure 7.1. shows worldwide smartphone market share forecast, comparing Android, iOs (Apple 2014), Windows Phone (Microsoft 2014), Blackberry (Blackberry 2014) and other minority systems.

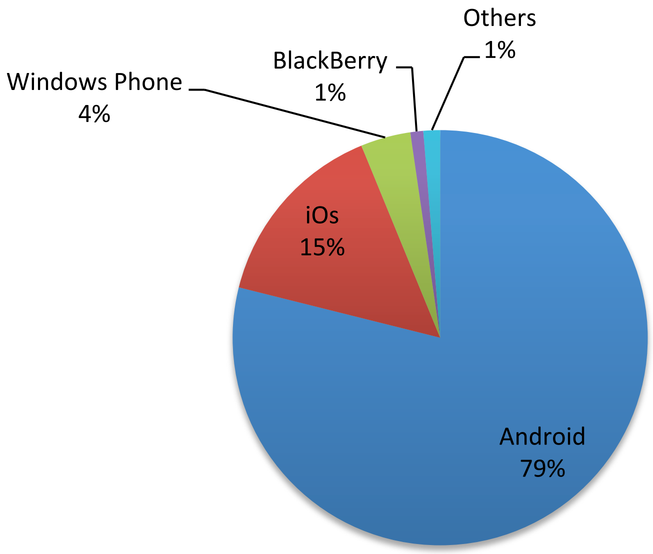


Figure 7.1. *Worldwide smartphone market share forecast, 2014–2018 (IDC 2014).*

Other alternative could be a multiplatform environment that allows running the app in several devices with different operating systems. Dallera states that implementing software on mobile devices is hard: everything is more complicated to accomplish than it is on the web or on the desktop (2011). However, a multiplatform environment brings issues such as slower execution and limited access to the device resources. Moreover, some features like GCM are better implemented in a native platform. Also, sometimes GCM is not included in multiplatform environment.

7.1 Storage

Android provides several options for saving persistent application data (Android 2014i), depending of the needs and the level of security expected. These alternatives are:

- Shared Preferences: store private primitive data in key-values pairs.
- Internal Storage: store private data on the device memory.
- External Storage: store public data on the shared external storage.
- SQLite Databases: store structured data in a private database.
- Network Connection: store data on the cloud.

Table 7.1. presents different features of each storage system in order to choose the best for the app. This evaluation is based on data availability, data security, and data capacity, meaning data security in the implicit systems that Android provides to access to this data.

Table 7.1. *Android storage systems evaluation.*

	Shared Preferences	Internal Storage	External Storage	SQLite Database	Network Connection
Availability	+ +	+ +	+	+	– –
Security	+ +	+	+	+	– –
Capacity	– –	–	+	-	+ +

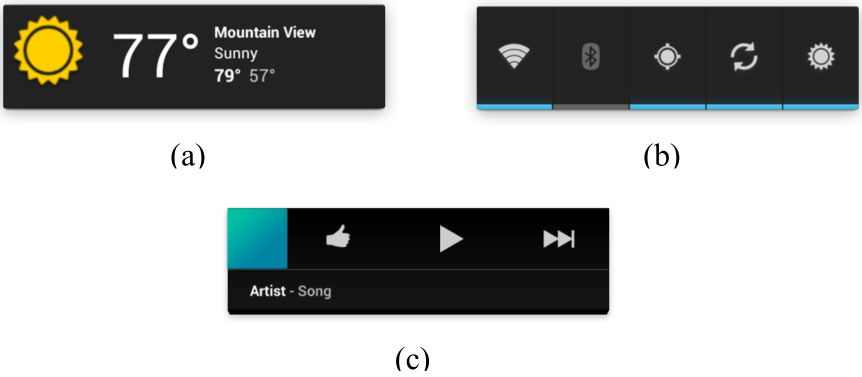
Based on the features the app of this thesis will use Shared Preferences since the amount of information stored is very low and the security that system needs is very high.

7.2 Internet connections

One common task for most Android apps is connecting to the Internet. Most network-connected Android apps use HTTP (Fielding et al 1999) to send and receive data. Android contains the standard Java network package java.net with the function HttpURLConnection (Android 2014e), which can be used to access network resources. Android also contains the Apache HttpClient (Apache Tomcat 2014) library. Both support HTTPS, streaming uploads and downloads, configurable timeouts, IPv6 and connection pooling (Oracle 2014). For Android 2.3 and later, HttpURLConnection is the best choice: transparent compression and response caching reduce network use, improve speed, and save battery (Android 2014e). HttpURLConnection in combination with HTTPS provides a secure channel to send data, increasing system confidentiality.

7.3 Widget

Android (Android 2014d) describes widgets as miniature application views that can be embedded in other applications, such as the Home screen. Widget apps need another Android app, which controls it from a remote service.



(a)

(b)

(c)

__Figure__ 7.2. (a) Information widget. (b) Control widget. (c) Hybrid widget. (Android 2014d)

All the Android resources are not available to be used by widgets. Due to this limitation, widgets are used as complement to an app in order to provide the user a quick access to some app functionality or information. Figure 7.2. illustrates the interaction resources available such as buttons, text boxes, and pictures boxes.

7.4 Google Cloud Messaging

Google Cloud Messaging for Android (GCM) is a service that allows sending data from a server to an Android-powered device, and also for these to receive messages from Google servers on the same connection. The GCM service handles all aspect of queuing of messages and delivery to the target Android application running on the target device (Android 2014f).

GCM needs to be allowed, as Program 7.1. shows, in the app configuration file called App Manifest (Android 2014c). The user should accept special permissions declared in the App Manifest when the app is installed in the device.

```
<user-permission android:name=
      "com.google.android.c2dm.permission.RECEIVE"/>
```

***Program** 7.1. Google Cloud Messaging declaration in Manifest App file.*

Also, a Google Device Token (GDT) should be obtained through a method implemented in the class *GoogleCloudMessaging* provided by Android libraries. This token is used as a reference to identify a device. It should be stored in order to connect with the device using Google services.

Figure 7.3. shows the process to get a GDT and to send messages from a server to a device using GCM. This process is initiated on the device that connects with GCM servers requesting a GDT for the app. GCM server allocates the GDT for the device and the app, sending it to the device. Once device has the GDT, it should send it to the app server in order to store it for future communications. When the server wants to send any message to the app using GCM, it sends the message to GCM servers including the device's GDT. GCM sends the message to the device using the push-notifications system implemented on Android OS.

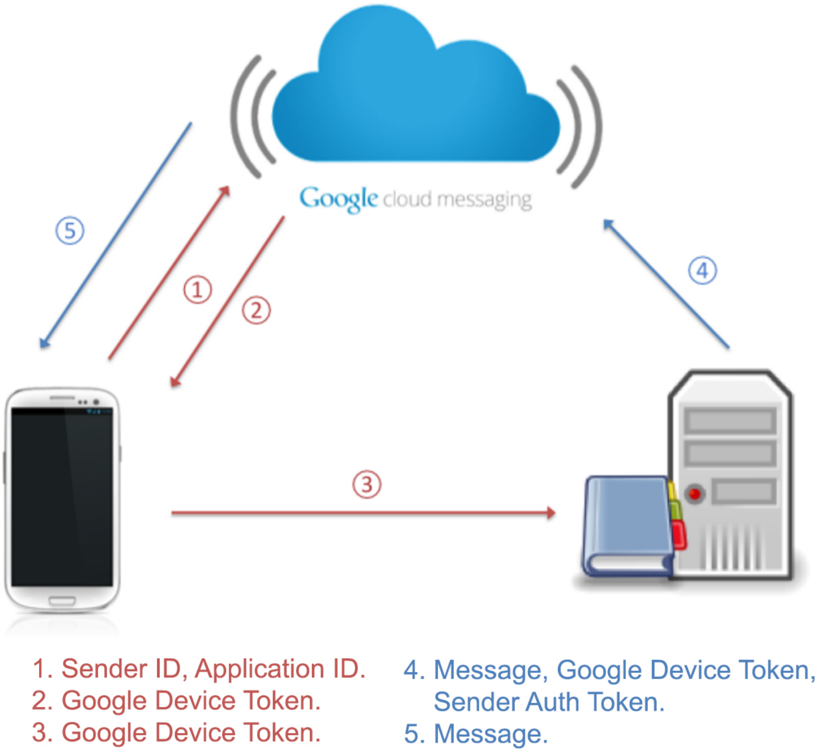


***Figure** 7.3. Google Cloud Messaging process (Siddappa 2014).*

Moreover, a Google developer account is required to use GCM and an Internet connection must be declared as Program 7.2. shows. A private identification number is provided by Google to every app, used to connect with the GCM service.

```
<user-permission android:name=”android.permission.INTERNET”/>
```

***Program** 7.2. Internet permission in App Manifest.*

7.5 Google Maps

Android provides map views using Google Maps. Google Maps for Android have features to identify locations with custom markers, augment the map data with image overlays, embed one or more maps, among others (Android 2014g).

In order to enable a map in an app, special meta-data in Program 7.3. must be declared in the App Manifest to set up the Google Play Services (Android 2014h).

```
<meta-data android:name="com.google.android.gms.version"
      android:value="@integer/google_play_services_version"/>
```

Program 7.3. *Google Play Services declaration in Manifest App file.*

Also, an API key obtained from the Android developer account must be included in the App Manifest, as Program 7.4. denotes, for getting access to Google Maps data, as well as an Internet connection.

```
<meta-data android:name="com.google.android.maps.v2.API_KEY"
      android:value="API_KEY"/>
```

Program 7.4. *Google Maps declaration in Manifest App file.*

Android Google Maps can use the user location to attach the position to the map view. This feature should also be accepted by the user in the permission list, so it should be declared in the App Manifest with the code included in Program 7.5. User location can be obtained through mobile networks or GPS signal.

```
<user-permission android:name=
      "android.permission.ACCESS_FINE_LOCATION">
```

Program 7.5. *App Manifest permission to network and GPS location access.*

8. HELP BUTTON IMPLEMENTATION

This chapter describes the implementation of the help button system. Database, Sender app, Receiver app, and Server will be described, giving technical details about the system described previously.

8.1 Database

The system uses a database with four tables: *alerts_history, locations, receivers,* and *senders.* Figure 8.1. shows these four tables, their cells, and the properties of these.

Tables *receivers* and *senders* contain receiver and sender GDTs, respectively. Both of them have two cells: *id* and *device. Id* identifies the device in the system, and *device* contains the GDT of the device used to send messages by GCM.

Table *alerts_history* stores information about the help situations that happened in the past. This table has six cells: *id, sender, sender_longitude, sender_latitude, receiver,* and *log*. *Id* identifies the item; *sender* and *receiver* are used to store the sender and receiver GDTs, respectively; *sender_longitude* and *sender_latitude* are the sender location where he or she sent the help request; and, finally, *log* contains all the steps done during the request.

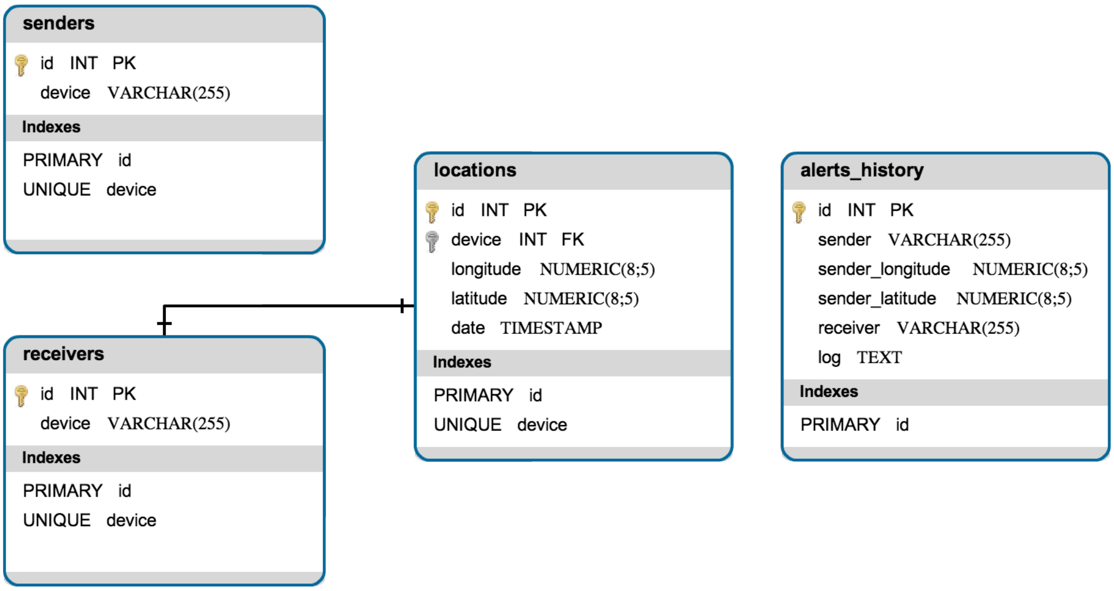
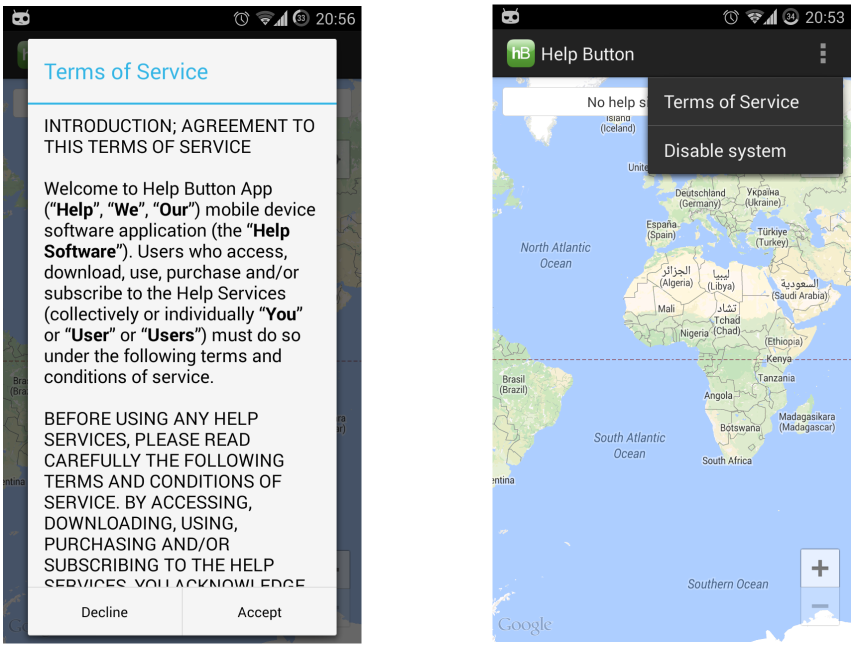


Figure 8.1. *Database graph.*

Table *locations* is used to keep the current location of the receivers in order to choose the best option for every help request. It contains five cells: *id, device, longitude, latitude,* and *date. Id* identifies the item; *device* contains the *id* of the table *receivers* (indicated in Figure 8.1. as foreign key since is a key imported from other table) that corresponds to the receiver; *longitude* and *latitude* store the current location of the receiver; and *date* is the time when the location was received in order to control how long data is stored.

8.2 Sender app

The first time that the app is run, a terms of service message is shown to inform to user about these. If the user does not accept the terms of service the application will not work. Besides, terms of services are available in the app menu to reread or withdraw user acceptance. This message is shown as in Figure 8.2.



Terms of Service

INTRODUCTION; AGREEMENT TO THIS TERMS OF SERVICE

Welcome to Help Button App (“**Help**”, “**We**”, “**Our**”) mobile device software application (the “**Help Software**”). Users who access, download, use, purchase and/or subscribe to the Help Services (collectively or individually “**You**” or “**User**” or “**Users**”) must do so under the following terms and conditions of service.

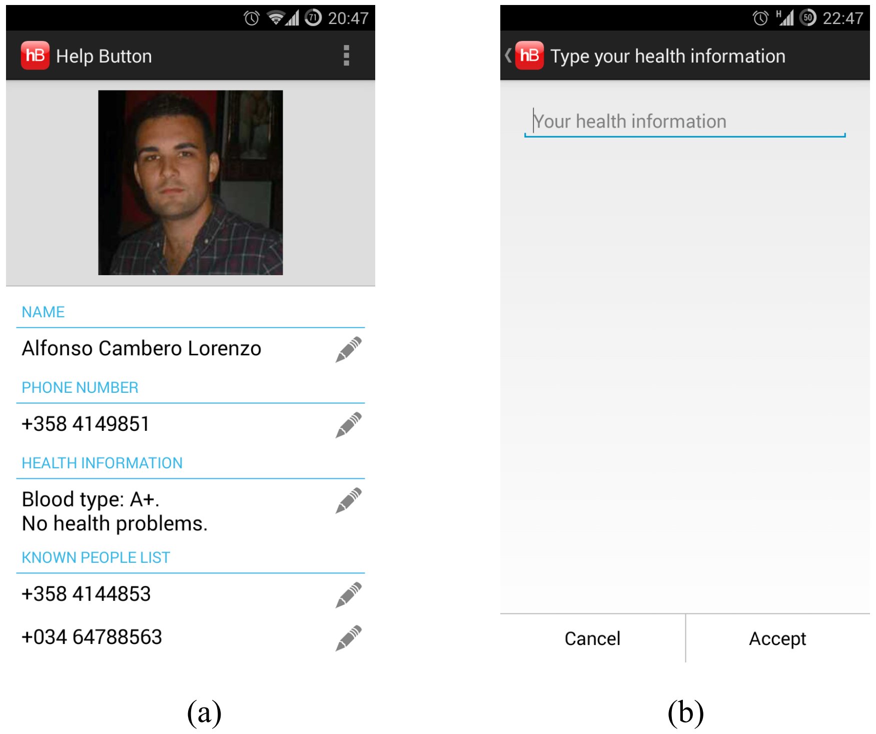
BEFORE USING ANY HELP SERVICES, PLEASE READ CAREFULLY THE FOLLOWING TERMS AND CONDITIONS OF SERVICE. BY ACCESSING, DOWNLOADING, USING, PURCHASING AND/OR SUBSCRIBING TO THE HELP SERVICES, YOU ACKNOWLEDGE

(a) (b)

Figure 8.2. (a) Terms of service message. (b) Receiver app menu to reread or withdraw user acceptance.

8.2.1 Personal information interface

Sender app provides an interface to configure personal data, such as name, picture, and health data. In this interface the user can find a picture selector, to choose a picture from the gallery of photographs of the mobile. This picture will be sent to the receiver, making recognising the sender easier. Name and health information boxes are available to inform to the receiver about the sender name and some relevant health information in order to prepare the sender for any situation.



NAME
Alfonso Cambero Lorenzo
PHONE NUMBER
+358 4149851
HEALTH INFORMATION
Blood type: A+.
No health problems.
KNOWN PEOPLE LIST
+358 4144853
+034 64788563

Your health information

(a) (b)

Figure 8.3. *(a) Personal information general interface screenshot. (b) Personal information health information interface screenshot.*

Additionally, there are some boxes for telephone numbers. One of them is for the user phone number. In this project it will be included but not stored in any database due to privacy issues. Other three telephone boxes are provided for friends or family. These are used to call them if there is no Receiver available. Figure 8.3. shows this interface, and a detail of an input data box dialog.

8.2.2 Personal information functionality

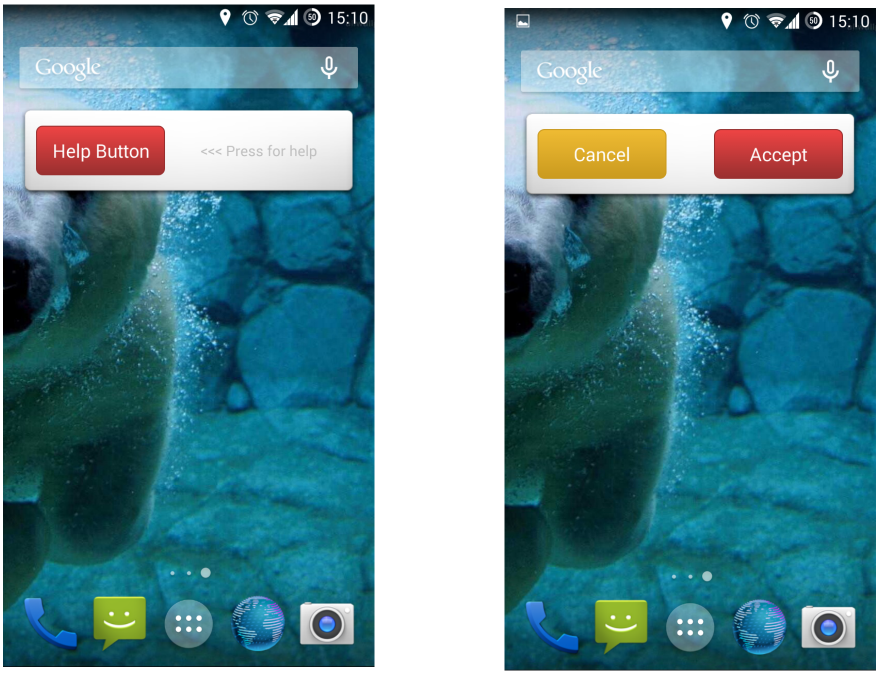
Personal information module is in charge of registering the Sender device in the system. Using the Google device token (GDT) as identifier, the device sends it by an HTTP request POST to a server servlet and this registers the device in the correct table.

The data in the text views is stored in the local memory, which only the app is allowed to access. It is stored through *sharedpreferences*, so that the data will persist across user sessions, that is, even if the app is killed. Using this storing system, app improves user privacy.

8.2.3 Widget interface

First, the widget implemented in the app was designed as a blackboard in the screen where the user could draw a right angle or some figure similar to this, activating the system. However, the implementation was not possible since Android does not allow horizontal swiping gestures in a widget, the device would understand it as a screen view change.

Finally, Figure 8.4. shows the widget developed, with a button to be pressed for sending a request. However, a single button could be pressed by mistake. To solve this problem, another button was implemented. When the user presses the button, other two buttons appear asking about confirmation or cancellation of the request. If the user presses the cancel button or waits twenty seconds without pressing any button the request is not sent and the widget returns to the original state. On the other hand, if the user presses the confirmation button a help request manager interface is opened and it begins to communicate with the server.



(a) (b)

Figure 8.4. *(a) Widget screenshot. (b) Widget screenshot after press the first button.*

8.2.4 Manager interface

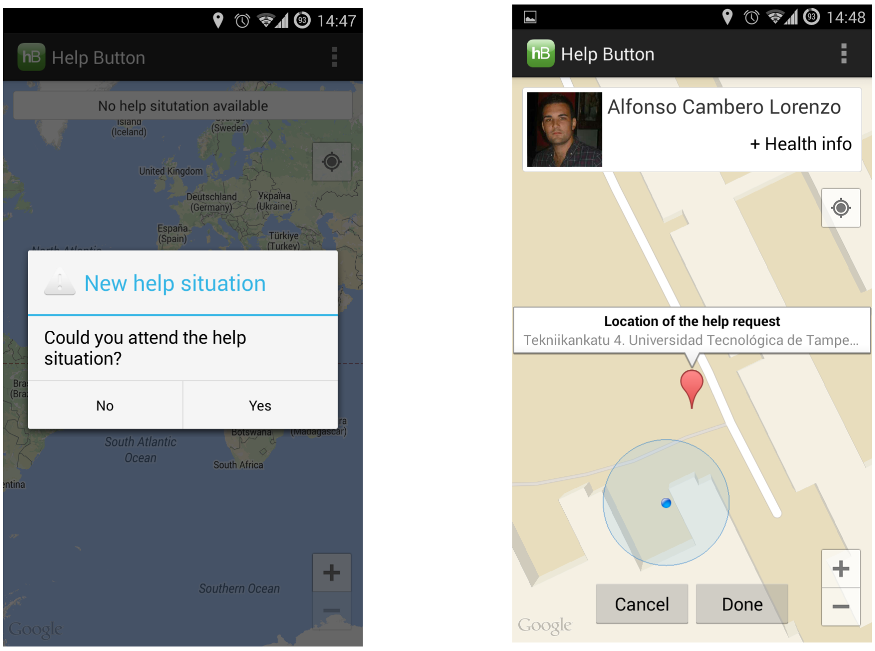
After generating a help request through the widget, a manager interface is opened automatically. In this window the user can see the state of the request. The states shown are: *sending, looking for a receiver, accepted*, and *done.* The sender has a slider button to cancel the help request when she or he wants. Besides, if a receiver informs about the end of a request, before finishing it, sender is asked in order to guarantee that request has been taken care of. If sender does not confirm, the help request continues with a different receiver. Using this mechanism, unattended request are not stored as completed.

8.3 Receiver app

The first time that the app is run, a terms of service message is shown to inform the user about these. If the user does not accept the terms of service the application will not work.

8.3.1 User interface

Receiver app provides the user with an interface to receive and visualize all the information about the sender. This includes a map, using the Google maps API (Google 2014) that shows the sender location. In addition, an information box is shown containing the sender name, picture, and health data.



No help situtation available

New help situation

Could you attend the help situation?

Alfonso Cambero Lorenzo

Location of the help request

Tekniikankatu 4. Universidad Tecnológica de Tampe...

(a) (b)

Figure 8.5. *(a) Request question. (b) Sender information in receiver app.*

When the server chooses a particular Receiver, that Receiver app receives a push notification from GCM and the user interface of this is opened automatically, vibrating and ringing, asking if the receiver wants to attend the help request. If the receiver says no, the app sends an HTTPS POST to the server with the answer. However, if he or she says yes the app not only sends the answer but the information about the request is shown in the interface opened and it begins to communicate with the server. Figure 8.5. shows a help request message and the interface of a help request situation accepted.

Moreover, two buttons are available to inform about the end of the help situation from the receiver or cancellation of it. If receiver claims successful end, as has been said before, sender receives a notification about it and a question box to confirm this event. If the receiver cancels, the server receives an HTTP POST request and can choose another receiver available.

8.4 Server

Server provides several web servlets to receive messages from senders and receivers and to link them. Also, server connects with Google Server in order to communicate with apps using push notification requests.

Server is implemented through Java classes and servlets. There are eleven servlets in charge of receiving information from the devices and two java classes that control every action in the system and database interaction. Figure 6.14 shows the classes and servlets in an UML diagram. More information about Server and its code is in the repository https://github.com/alfonsocamberolorenzo/HelpButtonServer.

The main class is called *Help.java* and is in charge of controlling everything related to a help request. This class provides a constructor to create a help object that contains all the information needed for a help situation. The help object will be stored in a HashMap until the help situation ends to keep it in the local memory and to be able to access to some help object where necessary. This class provides some public and private methods to control the help situation, such as:

- *accept* registers a receiver acceptance and informs the sender that the request was accepted.
- *calculateDistanceByHaversineFormula* calculates the distance between two points given as longitude and latitude
- *storeInfo* stores in database a log text with all the information about changes during a help request.

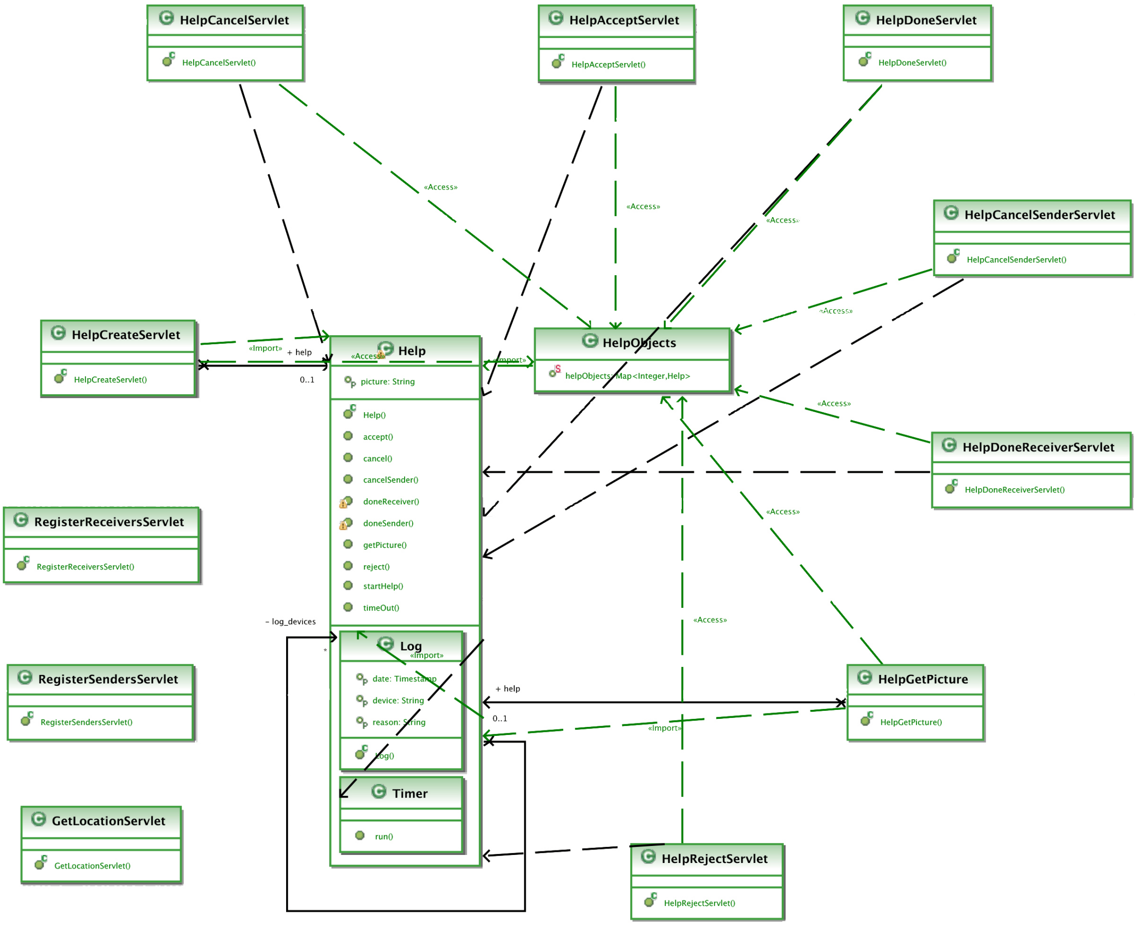


Figure 8.14. *Server UML diagram*

Help.java uses the classes *Log* and *Timer* included in this. Class *Log* provides objects to store data about every change in the help situation until the request ends when they are stored in the database. Class *Timer*, which implements Java class *Runnable,* creates a thread that automatically excludes the current Receiver so that another can be chosen, if after some seconds the request has not been accepted. This class could also be used to control how much time the situation is taking.

Class *HelpObjects* implements a HashMap to store all the active requests to be able to access to them using a reference. When some servlet executes any function, it needs the object obtained from the HashMap.

Server provides eleven Java servlets. Servlets are the Java platform technology of choice for extending and enhancing Web servers. Servlets provide a component-based, platform independent method for building Web-based applications, without the performance limitation of CGI programs. Servlets have access to the entire family of Java APIs, include the JDBC API to access a database (Oracle 2013), which is needed in the system.

Server's servlets are in charge of receiving information from both applications, server and receiver app, and doing everything needed in the help situation. These servlets are:

- *GetLocationServlet* receives receiver's location and stores it in the database.
- *HelpAcceptServlet* receives a message from receiver informing about the receiver's acceptance.
- *HelpCancelSenderServlet* receives a message from sender informing about the cancellation of the help situation.
- *HelpCancelServlet* is used to cancel a help request from the receiver side.
- *HelpDoneReceiverServlet* receives a message from the receiver informing about the end of the help situation.
- *HelpDoneServlet* receives a message from the sender informing about the end of the help situation.
- *HelpGetPicture* returns the sender's picture in a help request when receiver executes it.
- *HelpCreateServlet* is the servlet in charge of receiving sender's request to create a new help object.
- *HelpRejectServlet* is used by the receiver to reject a help request.
- *RegisterReceiverServlet* registers receiver users in the system.
- *RegisterSenderServlet* registers sender users in the system.

9. EVALUATION

In this chapter system will be evaluated according to the criteria presented in the Chapter 2, first data security, then privacy.

Data security will be evaluated according to how correctly access, flow, inference, and cryptographic controls work in the system. These will be compared to the level of confidentiality established in the data security chapter, in order to get a data security score.

Survey was a means to grasp the social concept of privacy. When social aspects are involved in technological issues, the matter has to be focused from a different point of view. Survey allows withdrawing from the problems related with society, since, in this work, required privacy level will be established according to its results.

9.1 Data security

Sender and receiver apps will be evaluated together since both apps use the same data security mechanism. They also communicate with the same server and so can be considered the same app in a data security evaluation.

Sender and receiver apps provide high-level access control because SharedPreferences feature of Android is used to store data. This feature guarantees that only the app that stores the data can access it.

HTTPS protocol ensures flow control since a secure channel is created. This secure channel is based on the data encryption provided by the HTTPS protocol. If there is no available HTTPS server and HTTP is used, data security is decreased due to lack of flow control.

Cryptographic and inference controls are not implemented in the apps. This lack decreases system confidentiality.

The confidentiality level of the apps can be worked out from the security levels obtained. Although both apps were analysed as one, the level of confidentiality can vary, since each app manages different information.

Access controls in the server to create a help request are implemented using the GDT.

To access to some help request, HTTP POST must contain a GDT valid for it (there are only two valid GDTs, those of the sender and the chosen receiver) and a unique help request reference. These provide a reasonable level of access control. However, access controls cannot be considered to be of high level since GDT could be easily forged as a real one. Flow controls are supported by HTTPS requests. As with the sender and receiver apps, the server can be considered to have a high level of data security. Inference controls are not implemented in the Server, so level of confidentiality will decrease.

Finally, cryptographic controls present the lowest level of data security because of server does not encrypt any data. The only encryption available in the system is the HTTPS protocol data encryption and the PostgreSQL database encryption.

Table 9.1. *System data security evaluation.*

	Access Controls	Flow Controls	Inference Controls	Cryptographic Controls
Sender app	+ +	+ +	?	− −
Receiver app	+ +	+ +	?	− −
Server	+ +	+ +	− −	− −

Table 9.1. Mobile app and server evaluation, related to data security evaluation.

Sender app confidentiality relies, mainly, on flow control since Sender's location is got only and sent, when flow controls works, to the server when a help request is created, without storing it in the app memory. Sender app presents a high level of confidentiality.

Receiver app confidentiality resides in access and flow control, although confidentiality is strongly based on access controls since critical data (sender's location, picture, name, and health information) is managed by the app during a help situation. As it was explained in Chapter 5, flow controls secure data when it is sent from a source to a receiver, and Receiver app needs to get all the data from the server. Moreover, that data is stored temporally in the device, when access controls are taken into account. Taking into account the safeguards, like access and flow controls (Denning and Denning 1979), and the grade of the rest, Receiver app presents a high level of confidentiality.

Server confidentiality is based on flow and access controls since data is mainly in the Server as variables or flowing between Server and the apps. Confidentiality in Server, taking into account that access and flow controls support the most important operations over inference and cryptographic, can be considered to be at medium level.

9.2 Privacy

Before analysing system privacy, privacy requirements should be established. Society strongly influences system privacy. Social aspects determine a confidentiality level that a system should ensure. Once the society indicates a level of privacy, the system will be evaluated against the level of privacy that system confidentiality supplies.

In order to define privacy level, an online survey was made (Appendix A). It was posted on social networks Facebook (Facebook 2014) and Twitter (Twitter 2014), in order to be answered by people that are used to using social networks and share information on the Internet. The survey had three principal questions about privacy level expected shown in Table 9.2., graded from one to five by its respondents. All the questions had to be answered. Results of these questions will be weighed by a value depending on the user's knowledge about privacy in the Internet and mobile apps.

Table 9.2. *Survey questions.*

	Question
Question 5	If you were a volunteer, what level of privacy would you require from an automatic system that knows your current location in order to help people with health problems?
Question 6	If you were a user requesting for help, how important would you consider the fact that the system may provide your personal information (name, picture, and health information) to other users in order to improve the system service?
Question 7	How important would you consider storing data after a help situation in order to guarantee user's security and satisfy current legislation?

The question five response of five meant that user is very concerned about her or his location privacy regardless of the use of it, and one not at all. In the question six, users rated the privacy level expected when they share name, picture, and health information in order to receive a help service. Giving a rating one in the question seven, a user completely agrees with storing data, and five that user completely disagrees. Results are shown in Figure 9.1.

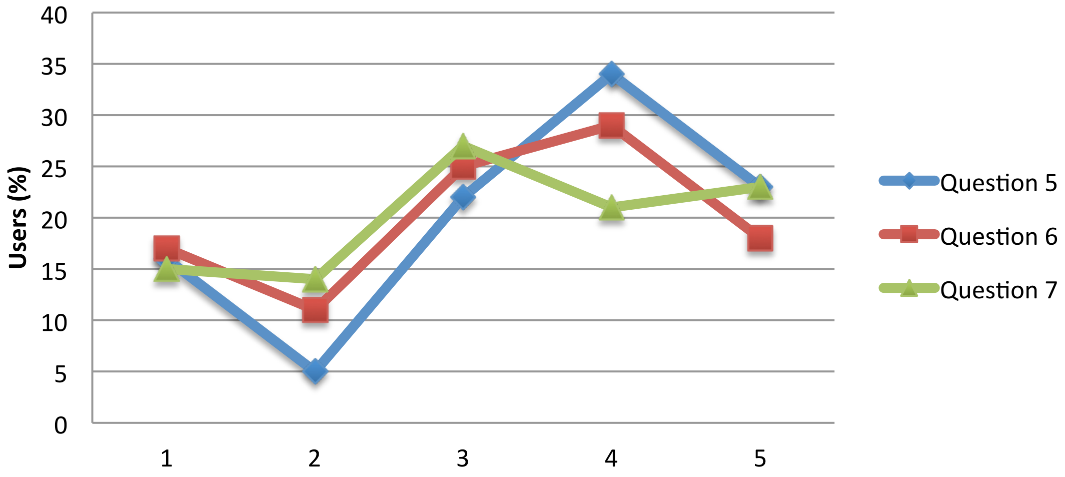


Figure 9.1*. Questions results.*

Data from Figure 9.1. can be used to calculate average and standard derivation for each of the questions. Average can be used to establish level of privacy, since it is the medium point between all the levels of privacy required by users. Standard derivation informs about how reliable is the average to be used as level of privacy.

Table 9.3. *Average and standard derivation survey results.*

	Average	Standard Deviation
Question 5	3,43	1,3353
Question 6	3,20	1,3333
Question 7	3,23	1,3548

Table 9.3. shows that level of privacy for all of the questions is between three and four. Moreover, standard deviation is relatively high. To check if these levels of confidentiality are good enough to cover as many users as possible, the cumulative distribution function (CDF) will be calculated. CDFs for question five, six, and seven are shown in Figures 9.2., 9.3., and 9.4.

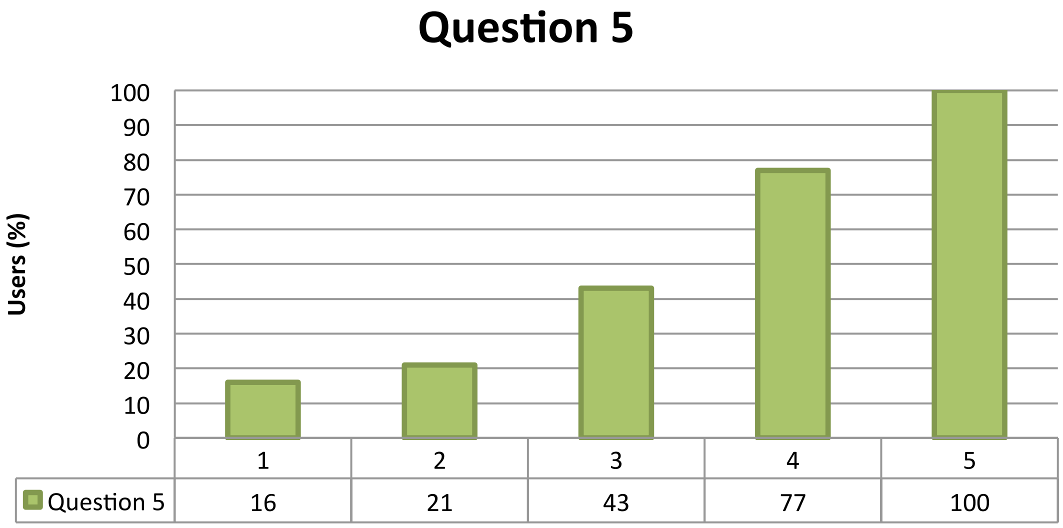


Figure 9.2. *Cumulative distribution function for question 5.*

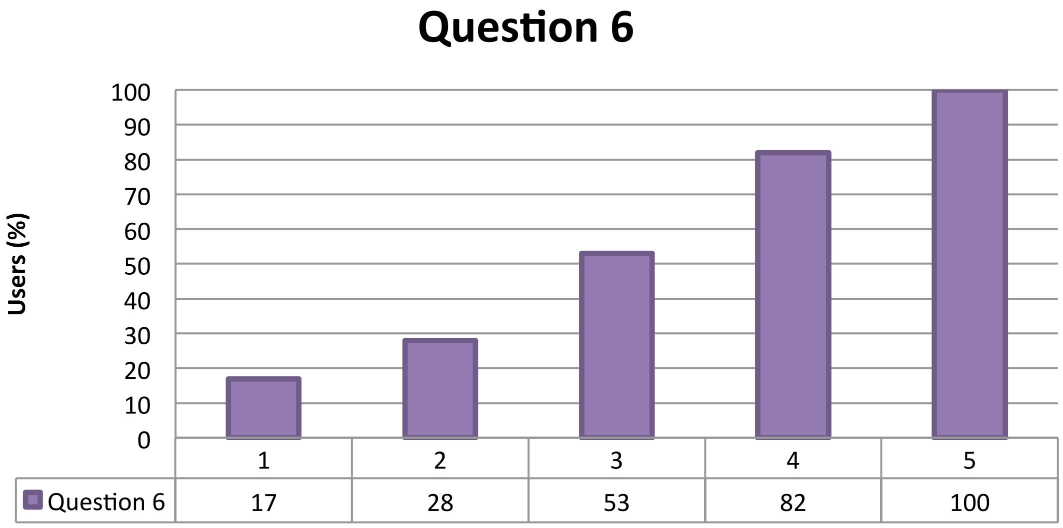


Figure 9.3. *Cumulative distribution function for question 6.*

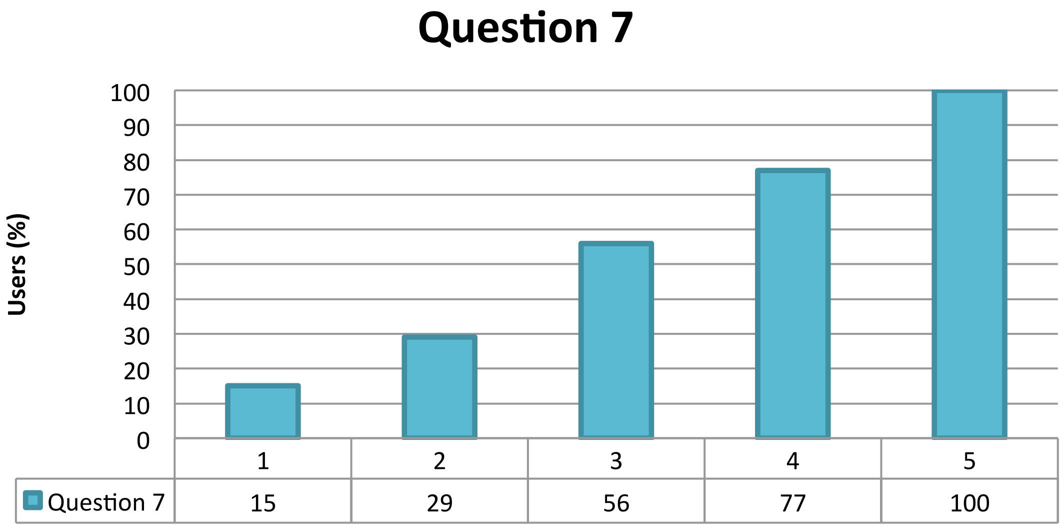


Figure 9.4. *Cumulative distribution function for question 7.*

Since satisfying every user in the system implies the highest level of confidentiality, and taking into account that it interferes with some functionalities of the system, level of privacy will be established according to survey results. This level, regardless of the privacy average, must satisfy, say, at least, 75% of users. This number of users is obtained from the results of question about system rate in the survey. Around 55% of users consider that system has a value of four or five over five. Then 37% of users consider a value of three. Taking users over three and the half of users that rated it with a three, total number is 73,5%. Rounding this value, the number of users to satisfy should be 75%, because they will be the active users of the system, who thinks that it provides an added value to the society as a help social network.

Satisfying 75% of users, Figures 9.2., 9.3., and 9.4. show that level of privacy for the three questions have to be four. From this level of privacy expected by users, level of confidentiality is obtained. In the case of privacy, both levels can be considered as proportional, so level of confidentiality have to be four over five in all the questions.

The results of the survey allow the conclusion that all the mobile apps have to provide a confidentiality level of four. Server has to provide at least the same confidentiality level, since all the data is managed by it. Server will be required a confidentiality of four.

9.3 Conclusions

After analysing confidentiality provided by data security, and getting confidentiality level expected by users, we are going to compare them in order to discover if the system's confidentiality is enough for the users' requirements.

In order to have a common representation of confidentiality levels, grades one to five from the survey will be considered symbols, like in Table 8.1. To make the translation, one to five being replaced by – –, –, 0, +, and + +, respectively.

Table 9.3. *Confidentiality summary.*

	Sender app	Receiver app	Server
Achieved Confidentiality	+	+	0
Required Confidentiality	+	+	+
Result	✔	✔	✖

As Table 9.3. shows, server does not achieve the confidentiality level expected by users, four. This is due to server not implementing strong cryptographic and inference controls. These problems should be fixed because with a level three of confidentiality, according to the survey, only half of the users are satisfied.

To implement cryptographic controls an algorithm for encryption as Advanced Encryption Standard (AES) could be used. This is a symmetric block cipher that can encrypt and decrypt information. AES algorithm is capable of using cryptographic keys of 128, 192, or 256 bits to encrypt and decrypt data in blocks of 128 bits, although blocks of 160, 192, 224, and 256 bits are also supported (NIST 2001).

On the other hand, sender and receiver apps attain the expected confidentiality. Although sender and receiver apps do not implement any cryptographic and inference control, the kind and amount of data that they manage is not so critical as that of the server, and the weight of them in the overall grade is lower. Furthermore, Android implements its own cryptographic controls, and the apps do not use a database, which would need inference controls.

System then satisfies most users according with the privacy level expected. This privacy level cannot be taken as a universal value since it depends on people and its satisfaction and concern about data handled by the system.

The increase of privacy could affect the system functionality. For this reason, after implementing the changes above, system would be ready to be published.

10. SUMMARY

This thesis analyses LBSs, linking systematically the technological and social aspects involved in the service through the concept of confidentiality. Expected privacy in LBSs is related to data security using confidentiality. Data security technologies have been analysed for knowing how they improve the level of system confidentiality. On the other hand, privacy has been analysed to know how system confidentiality modifies level of privacy.

This thesis also implements an LBS, *Help Button*, in order to use and verify the previous analysis. *Help Button* service shows how technology guarantees system privacy. Techniques shown in the theoretical analysis have been used to ensure data security. Using these techniques a level of system confidentiality has been achieved. Also, a survey has been done to establish an expected level of privacy in the system, so that a level of expected confidentiality was established. After getting both levels of confidentiality, from technological and social aspects, they have been compared to evaluate how good the system is.

Further analysis about confidentiality in mobile LBSs would be useful to complete the information provided in this work, as well as analyse to new data security techniques and algorithms for LBS privacy preservation.

In terms of the social app developed in this thesis, a list of possibilities for the near future could be:

- **Cryptographic controls:** cryptographic controls must be implemented in the server and the mobiles apps, in order to improve system security and achieve the complete functionality of this.
- **Inference controls:** inference controls should be implemented in the server in order to achieve all the data security safeguards described in the data security analysis.
- **System attack:** an external source, unknowing the system implementation, should try to attack the system to assess the resilience and discover the potential weak points.
- **Other platforms:** developing for iOs (Apple 2014), which is the second most popular mobile operating system, WindowsPhone (Microsoft 2014), BlackBerry OS (Blackberry 2014), and other minor platforms is needed to

get a larger community.

- **Survey:** designing a new survey to get a higher level of accuracy about privacy expected would be important.
- **Emergency services link:** linking the system with emergency services would add extra value to users. A reliable institution involved in the service would provide a sense of security for the senders.
- **Voice contact:** a voice contact by VoIP could improve service, allowing the receiver to contact the sender in order to know the real situation before arriving. VoIP could be the best option, avoiding mobile telephony, since the cost of a conventional call is higher than the VoIP call. It is understood assuming that all the users have a mobile Internet connection to use the system. Also, monitoring of this call by emergency services would reduce the action time of many incidents.
- **Volunteer circles:** allowing adding known volunteers to your circle, in order to choose them before some unknown person, will reduce action time.
- **Terrain analysis:** an exhaustive analysis of the locations is needed, in order to select the best option to attend to a help request, considering the real time to go someplace as influenced by, say, mountains, rivers, and borders.

REFERENCES

Android. 2014a. Android 4.4. KitKat. [Online]. Available: http://www.android.com [Accessed: 02.06.2014].

Android. 2014b. Android Developers. [Online]. Available: http://developer.android.com [Accessed: 02.06.2014].

Android. 2014c. App Manifest. [Online]. Available: http://developer.android.com/guide/topics/manifest/manifest-intro.html. [Accessed: 04.06.2014].

Android. 2014d. App Widget. [Online]. Available: http://developer.android.com/guide/topics/appwidgets/index.html. [Accessed: 03.06.2014].

Android. 2014e. DefaultHttpClient. [Online]. Available: http://developer.android.com/reference/org/apache/http/impl/client/DefaultHttpClient.html. [Accessed: 16.06.2014].

Android. 2014f. Google Cloud Messaging for Android. [Online]. Available: http://developer.android.com/google/gcm/index.html. [Accessed: 04.06.2014].

Android. 2014g. Google Maps Android API v2. [Online]. Available: http://developer.android.com/google/play-services/maps.html. [Accessed: 04.06.2014].

Android. 2014h. Google Play Services. [Online]. Available: http://developer.android.com/google/play-services/index.html. [Accessed: 04.06.2014].

Android. 2014i. Storage Options. [Online]. Available: http://developer.android.com/guide/topics/data/data-storage.html. [Accessed: 03.06.2014].

Apache Tomcat. 2014. Apache Tomcat. [Online]. Available: http://tomcat.apache.org. [Accessed: 02.06.2014].

Apple. 2014. iOs 7. [Online]. Available: http://www.apple.com/ios. [Accessed: 03.06.2014].

Bellocci, V., Genovese, S., Inguaggiato, D., and Tucci, M. 2002. Mobile Location-Aware Services: 2002 Market Perspective. Ericsson, Division Service Architecture and Interactive Solutions.

Blackberry. 2014. Blackberry 10 OS. [Online]. Available: http://us.blackberry.com/software/smartphones/blackberry-10-os.html. [Accessed: 03.06.2014].

Buhr, R. J. A. and Casselman, R.S. 1996. Use case maps for object–oriented systems. Prentice-Hall, 302 p.

Cisco. 2008. What is VoIP (Voice–over–IP). [Online] Available: http://www.cisco.com/c/en/us/products/unified~communicacions/networking_solutions_products_genericcontent0900aecd804f00ce.html. [Accessed: 08.07.2014].

Dallera, A. 2011. Why you should stay away from Appcelerator's Titanium. [Online]. Available: http://usingimho.wordpress.com/2011/06/14/why-you-should-stay-away-from-appcelerators-titanium. [Accessed: 03.06.2014].

Deering, S. and Hinden, R. 1998. Internet Protocol, Version 6 (IPv6) Specification RFC 2460. The Internet Society, 39 p.

Denning, D. E., and Denning, P. J. 1979. Data Security. ACM Computing Surveys 11 3, pp. 227–249.

Dierks, T. and Allen, C. 1999. The TLS Protocol Version 1.0, RFC 2246. The Internet Society, 80 p.

Diffie, W. and Hellman, M.E. 1976. New directions in cryptography. IEEE Transactions on Information Theory 22, 6, pp 644–654.

Downes, S. 2005. Authentication and Identification. International Journal of Instructional Technology and Distance Learning 2, 10, pp 3–18.

ESA. 2013. Galileo Navigation. [Online]. Available: http://www.esa.int/Our_Activities/Navigation/The_future_-_Galileo/What_is_Galileo. [Accessed: 05.06.2014].

European commission. 1995. D 95/46/EC. Protection of personal data, 20 p.

European commission. 2002. D 2002/58/EC. Data protection in the electronic communications sector, 11 p.

Facebook. 2014. [Online]. Available: https://www.facebook.com/facebook/info [Accessed: 12.07.2014].

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners–Lee, T. 1999. Hypertext Transfer Protocol — HTTP/1.1 RFC 2616. The Internet Society, 176 p.

Fisher, P. and Dobson, J. 2003. Who knows where you are, and who should, in the era of mobile geographic? Geography 88, 4, pp. 331–337.

Freier, A., Karlton, P., and Kocher, P. 2011. The Secure Sockets Layer (SSL) Protocol Version 3.0, RFC 6101. The Internet Society, 67 p.

Gruteser, M., and Grunwald, D. 2003. Anonymous usage of location-based services through spatial and temporal cloaking. Communication of the ACM, pp 31–42 p.

Harauz, J., Kaufman, L.M., and Potter, B. 2009. Data security in the World of Cloud Computing. IEEE Computer and Reliability Societies 7, 4, pp 61–64.

IDC. 2014. IDC Worldwide Mobile Phone Tracker. [Online] Available: http://www.idc.com/getdoc.jsp?containerId=prUS24701614. [Accessed: 03.06.2014].

Jagannathan, G., and Wright, R.N. 2007. Private inference control for aggregate database queries. National Science Foundation, 19 p.

Java. 2014. Java + You. [Online]. Available: https://www.java.com. [Accessed: 02.06. 2014].

Junglas, I. A. and Watson, R.T. 2008. Location-Based Services. Communication of the ACM 51, 3, pp 65–69.

Kalnis, P., Ghinta, G., Mouratidis, K, and Papadias, D. 2006. Preserving anonymity in location based services. Technical Report B6/06. The National University of Singapore, 12 p.

Kido, H., Yanagisawa, Y., and Satoh, T. 2005. Protection of Location Privacy using Dummies for Location-based services. Proceedings of the 21st International Conference on Data Engineering, pp 1248–1252 .

Linux Foundation. 2014. What is Linux. [Online]. Available: https://www.linuxfoundation.org/what-is-linux. [Accessed: 02.06.2014].

Mascetti, S., and Bettini, C. 2007. A comparison of spatial generalization algorithms for LBS privacy preservation. In Proceedings of the 2007 International Conference on Mobile Data Management, pp 258–262.

Microsoft. 2014. Windows Phone. [Online]. Available: http://www.windowsphone.com. [Accessed: 03.06.2014].

National Institute of Standards and Technology (NIST). 2001. Advanced Encryption Standard (AES). Federal Information Processing Standards Publications 197, 47 p.

National Research Council. 1995. The Global Positioning System: A shared national asset. Washington, DC: The National Academies Press, 263 p.

National Research Council. 2003. IT Roadmap to a Geospatial Future. Washington, DC: The National Academies Press, 136 p.

Needman, R.M., and Schroeder, M.D. 1979. The Cambridge CAP computer and its protection system. Proceedings of the Sixth ACM Symposium on Operating Systems Principles 21, 12, pp 993–999.

Olivier, M.S. 2002. Database Privacy: Balancing Confidentiality, Integrity, and Availability. SIGKDD Explorations Newsletter 4, 2, pp 20–27.ik

OMG. 2014. Unified Modeling Language (UML). [Online]. Available: http://www.uml.org. [Accessed: 04.06.2014].

Oracle. 2013. Java Servlet Technology Overview. [Online]. Available: http://www.oracle.com/technetwork/java/overview-137084.html. [Accessed: 28.05.2014].

Oracle. 2014. Connection pooling. [Online] Available: http://docs.oracle.com/javase/jndi/tutorial/ldap/connect/pool.html. [Accessed: 10.07.2014].

Parzen, E. 1962. On Estimation of a Probability Density Function and Mode. The Institute of Mathematical Statistics 33, 3, pp 1065–1076.

PostgreSQL 2014. PostgreSQL: The world's most advanced open source database. [Online]. Available: http://www.postgresql.org. [Accessed: 02.06.2014].

Rescorla, E. and RTFM, Inc. 2000. Hypertext Transfer Protocol Secure – HTTP/1.1 RFC 2818. The Internet Society, 7 p.

Sandhu, R. S. and Samarati, P. 1994. Access Control: Principles and Practice. IEEE Communications Magazine 32, pp 40–48.

Schiller, J., and Voisard, A. 2004. Location-Based Services. Morgan Kaufmann Publishers, 262 p.

Schneier, B. 1993. Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish). Fast Software Encryption, Cambridge Security Workshop Proceedings, pp 191–204.

Siddappa, D.C. 2014. Unwinding ADF. [Online]. Available: http://deepakcs.blogspot.fi/2013/06/adf-mobile-push-notifications-with.html. [Accessed: 17.07.2014].

Twitter. 2014. [Online]. Available: https://about.twitter.com/ [Accessed: 12.07.2014].

United Nations. 1948. The Universal Declaration of Human Rights. [Online]. Available: http://www.un.org/en/documents/udhr. [Accessed: 06.06.2014].

University of California at Irvine. 2014. Privacy and Confidentiality. [Online]. Available: http://www.research.uci.edu/compliance/human-research-protections/researchers/privacy-and-confidentiality.html. [Accessed: 06.06.2014].

US Department of Commerce. 1999. Data Encryption Standard (DES). Federal Information Processing Standards Publication 46–3, 22 p.

W3C. 2011. CGI: Common Gateway Interface. [Online]. Available: http://www.w3c.org/CGI. [Accessed: 04.06.2014].

W3C. 1998. MD5 Hash Algorithm. [Online]. Available: http://www.w3.org/TR/1998/REC-DSig-label/MD5-1_0. [Accessed: 02.06.2014].

Zwillinger, D. and Kokoska, S. 2010. CRC Standard Probability and Statistics Tables and Formulae. CRC Press, 49 p.

APPENDIX A: PRIVACY SURVEY

The survey was done in English and Spanish for covering different cultures and getting more accurate results. Survey consisted of the following:

- Question 1: Gender.
- Question 2: Age.
- Question 3: Nationality.
- Question 4: Rate your knowledge (from 1 to 5) about privacy in the Internet and mobile applications.
- Question 5: If your were a volunteer, which level of privacy (from 1 to 5) would you require to an automatic system that know your current location in order to help people with health problems?
- Question 6: If you were a user requesting for help, how important would you consider (from 1 to 5) the fact that the system may provide your personal information (name, picture, and health information) to other users in order to improve the system's service?
- Question 7: How important would you consider (from 1 to 5) storing data after a help situation in order to guarantee users' security and current legislation?
- Question 8: Rate the system (from 1 to 5) about your own opinion.

Seventy persons from twenty to sixty-three years old and several countries (Spain, Latvia, Singapore, France, and Brazil) answered the survey. Complete results for the questions were:

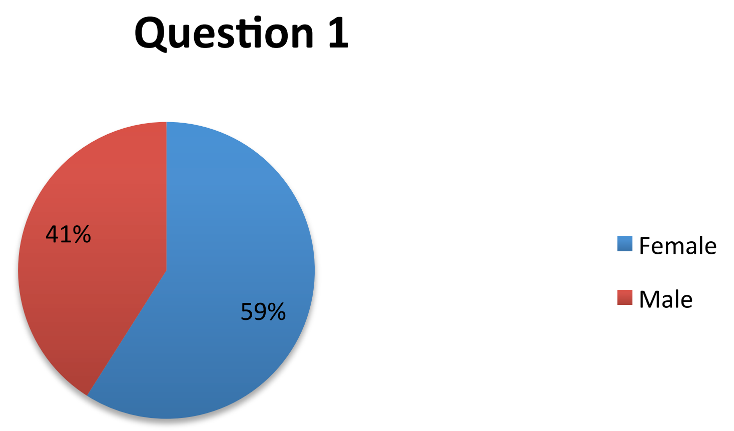


Figure A.1. *Question 1 survey results.*

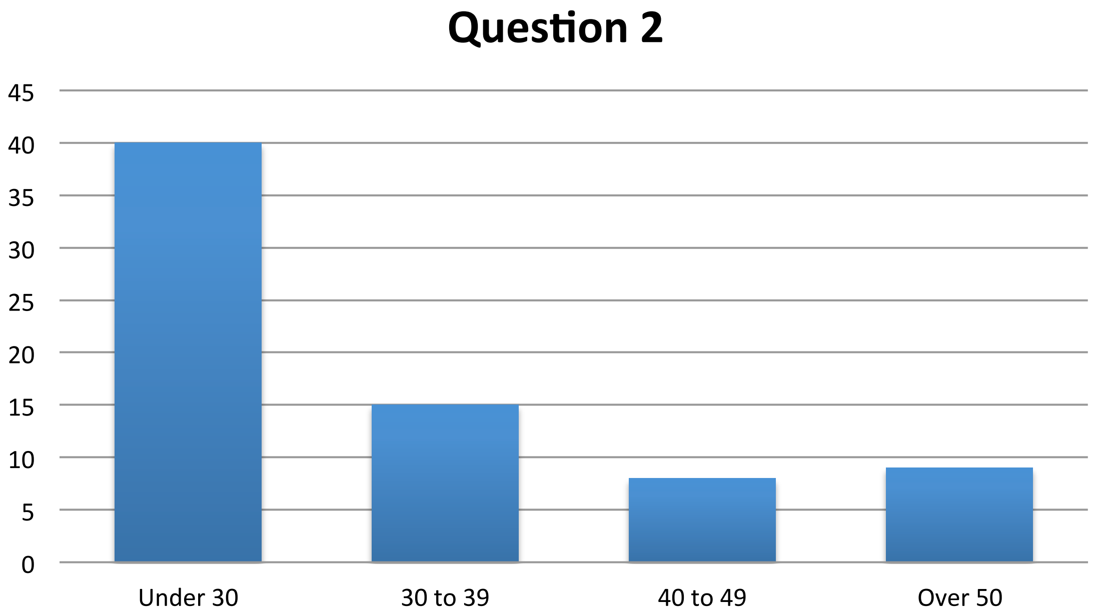


Figure A.2. *Question 2 survey resutls.*

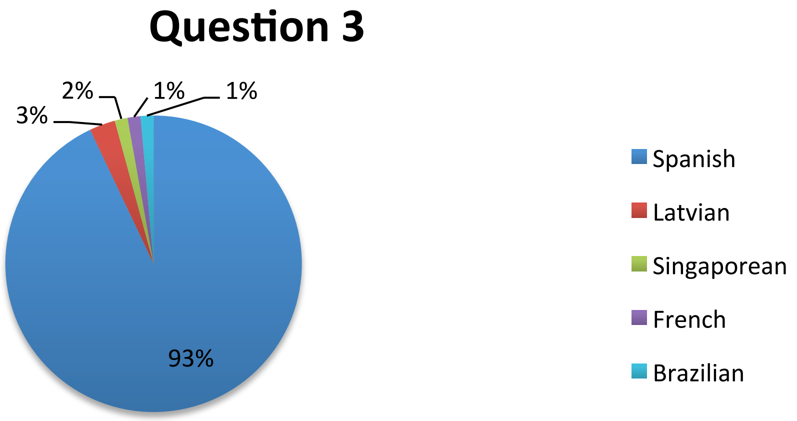


Figure A.3. *Question 3 survey resutls.*

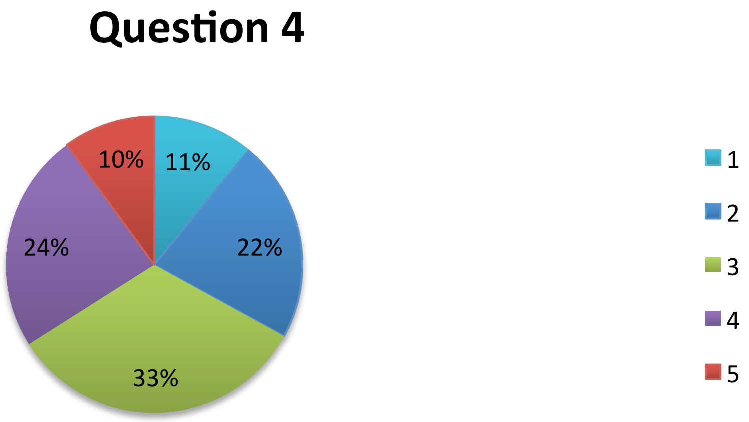


Figure A.4. *Question 4 survey resutls.*

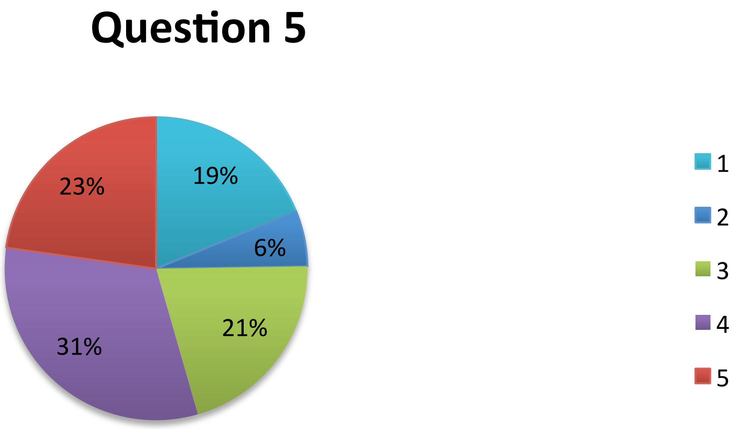


Figure A.5. Question 5 survey resutls.

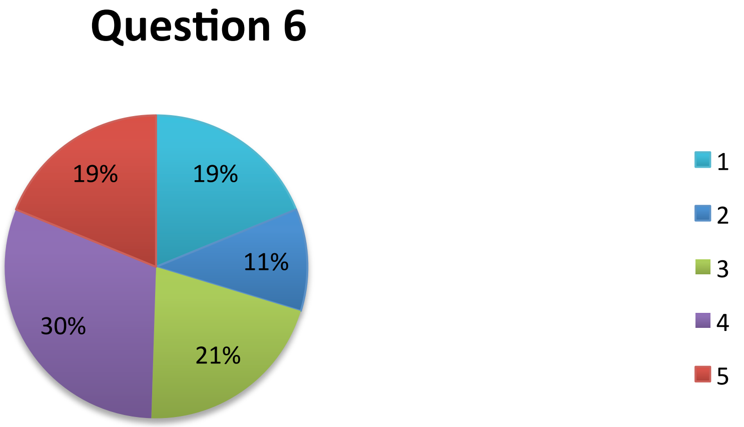


Figure A.6. Question 6 survey resutls.

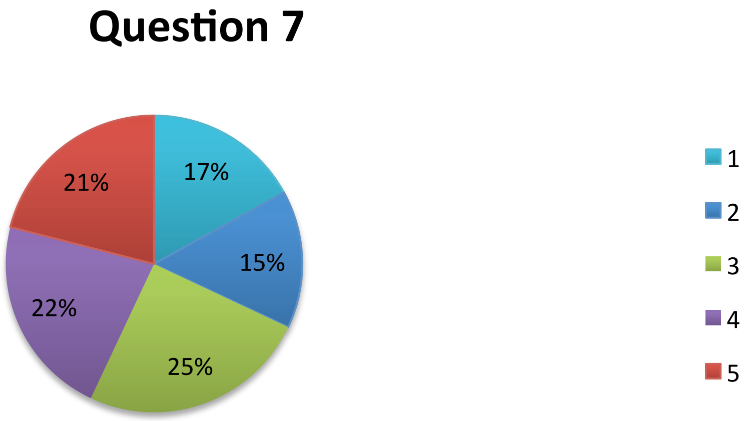


Figure A.7. Question 7 survey resutls.

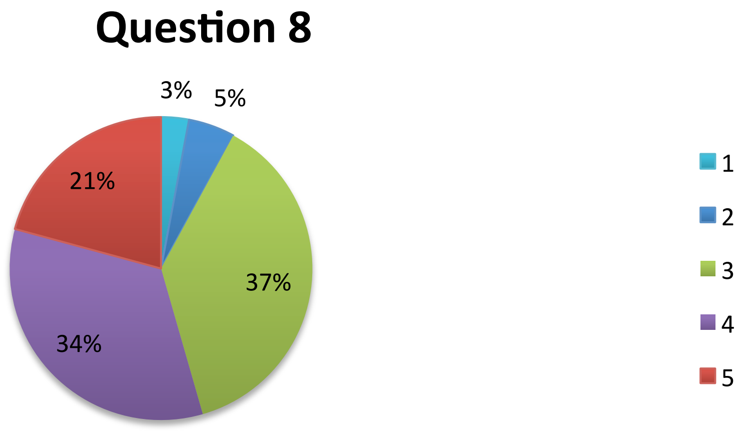


Figure A.8. *Question 8 survey resutls.*