Mobile cloud healthcare systems using the concept of point–of–care

Hazzaa Naif Alshareef
MSc, BSC. (HONS) COMPUTER SCIENCE
108138281

Thesis submitted for the degree of Doctor of Philosophy

NATIONAL UNIVERSITY OF IRELAND, CORK

Faculty of Science

Department of Computer Science

January 14, 2017

Head of Department: Professor Cormac J. Sreenan

Supervisors: Dr. Dan Grigoras

Research supported by the Saudi Electronic University in Saudi Arabia

Contents

I, Hazzaa Naif Alshareef, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.

Hazzaa Naif Alshareef

This thesis is dedicated to my son, Naif, and my daughter, Najat. This work is for, and because of, you and all the generations to come. It is dedicated to all our journeys in learning to thrive.

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr Dan Grigoras, for his invaluable guidance, encouragement and support. Dr Grigoras always valued my opinions, trusted in my abilities, and made me feel that we were partners, which encouraged me to reach for higher goals. I truly appreciate the freedom and flexibility I was given in my research pursuit.

In addition to my supervisor, I would like to thank Prof. Cormac J. Sreenan, Head of the Department of Computer Science, for his tremendous support and help during my postgraduate studies. Furthermore, I would like to express my thanks to the rest of my thesis committee: Prof. Traian Muntean, Aix-Marseille University, and Dr John Herbert, for their encouragement, insightful comments, and challenging questions.

I would like to thank my dear wife Norah for supporting and encouraging me to achieve this degree and my children, Naif and Najat, for their patience and understanding. I hope that from my journey you learned about believing in yourself, being dedicated, and pursuing dreams.

In addition, I am grateful to my parents and my siblings, Arwo (and her two sons Mohaned and Mohamed), Hattan, Sarah (and her two lovely daughters Fatimah and Maryam) and Ali, for their quiet but steady encouragement. They have been a constant source of love, support, and encouragement during the challenges of study and life. I am truly thankful for having you in my life.

A special thank you goes to my two research partners in the mobile cloud research area, Michael O'Sullivan and Aseel Alkhelaiwi, for sharing their ideas and providing feedback. I also want to extend my thanks to my best Irish friend, Ultan Neville, and my colleagues in the MISL lab, especially Mary Noonan and Jason Quinlan, for their wonderful collaboration.

Finally, I will not forget to thank my Saudi friends, Nasser, Ibrahim, Aziz, and Ahmed (and his three lovely children Mohamend, Mohand and Ritaz) in Cork, who shared the difficulty of being abroad and far from home and supported me during my study journey.

List of Figures

List of Tables

List of Algorithms

Abstract

Recent years have witnessed a rapid growth in delivering/accessing healthcare services on mobile devices. An example of a health practice/application that is benefiting from the mobile evolution is m–health, which is aimed at providing health services to mobile devices on the move.

However, mobile devices have restricted computational and storage capacity, and run on batteries that have limited power. These limitations render m–health unable to run the demanding tasks that may be required for accessing/providing health services. The mobile cloud has recently been proposed as a solution for dealing with some of the limitations of mobile devices, such as low storage and computing capacity. However, introducing this solution into the m–health field is not straightforward, as the integration of this technology has specific limitations, such as disconnection issues and concerns over privacy and security.

This thesis presents research work investigating the ability to introduce mobile cloud computing technology into the health field (e.g., m–health) to increase the chances of survival in cases of emergencies. This work focuses on providing help to people in emergencies by allowing them to seek/access help via mobile devices reliably and confidently, as well as the ability to build a communication platform between people who require help and professionals who are trusted and qualified to provide it. The concept of point–of–care has been used here, which means providing as much medical support to the public as possible where and when it is needed.

This thesis proposes a mobile cloud middleware solution that enhances connectivity aspects by allowing users to create/join a mobile ad–hoc network (MANET) to seek help in the case of emergencies. On the other side, the cloud can reach users who do not have a direct link to the cloud or an Internet connection. The most important advantage of combining a MANET and a mobile cloud is that management tasks such as IP allocation and split/merge operations are shifted to the cloud, which means resources are saved on the mobile side.

In addition, two mobile cloud services were designed which have the aim of interacting with users to facilitate help to be provided swiftly in the case of emergencies. The system was deployed and tested on Amazon EC2 cloud and Android–based mobile devices. Experimental results and the reference architecture show that the proposed middleware is feasible and meets pre–defined requirements, such as enhancing the robustness and reliability of the system.

Chapter 1

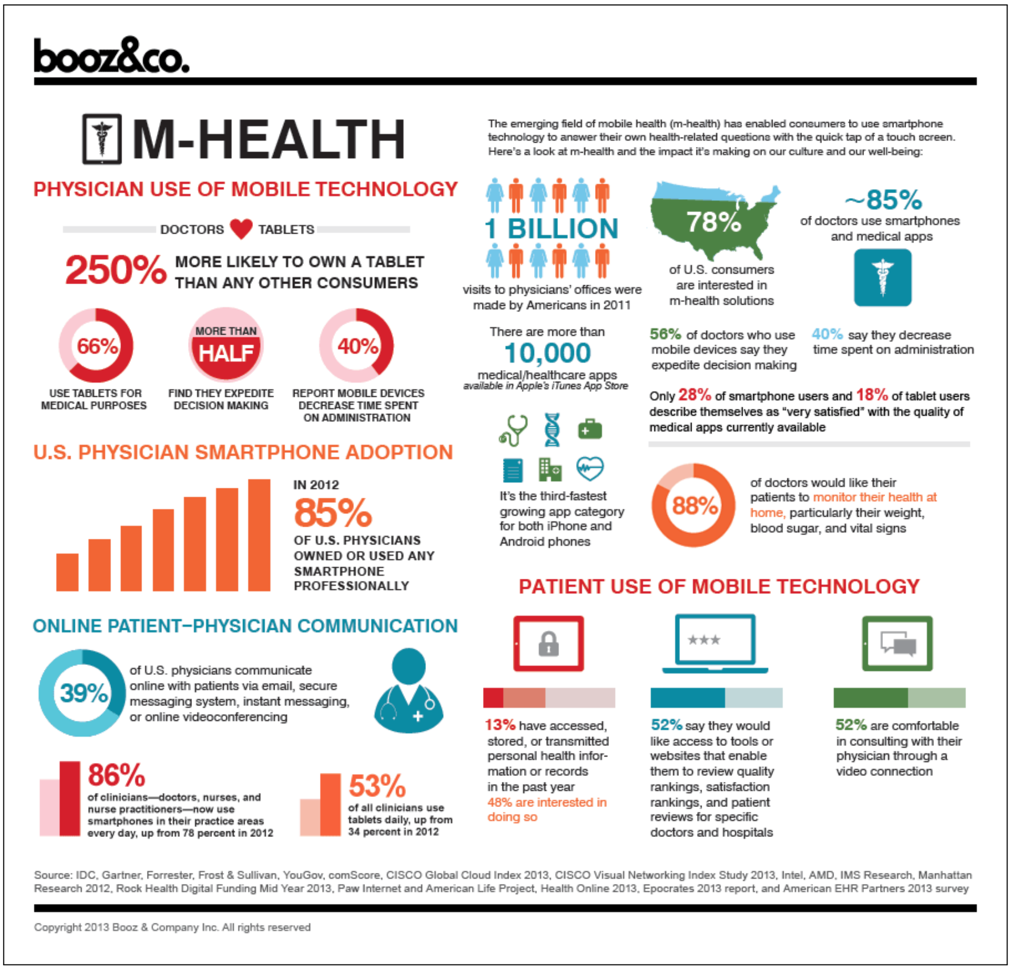
Introduction

1.1 M–Health

Recent years have seen an increase in the use of mobile devices such as smartphones, tablets, and smart–bands in people's lives. Features offered by these types of devices, such as ease of use and being wirelessly–enabled, allow people to access services that can improve their quality of life. One of the most important aspects of life that can be accessed through mobile devices are health services, whereby users have the ability to track their health status 'on the move', such as by tracking physical activities (e.g., walking and running) and monitoring body status (e.g., heartbeat rate). From the other side, medical professionals and centres can use mobile devices to provide better healthcare services to the public, as doctors, for example, can access patients' records and laboratory results on the move instead of looking up printed charts or files. This reduces the time needed to deliver healthcare to patients. Furthermore, users can communicate with medical professionals in cases of emergency to discuss a medical problem, which could, in turn, relieve pressure on emergency departments. These practices are known in the health sector as m–health.

M–health is an abbreviation of 'mobile health', which means to provide/deliver healthcare services on mobile devices such as smartphones, tablets and wearable devices to minimize shortcomings in the traditional medical framework in an efficient way. M–health has gained increased attention from researchers in both the medical and information technology (IT) fields for the purpose of reducing the pressure on hospitals, as well as cutting the cost of health services to make them available to everyone.

According to GreatCall [4], the m–health market will reach $26 billion by 2017 and, by the same year, half of smartphone owners will have downloaded mobile health apps. Information and figures related to m–health have been provided on the GreatCall site to answer the following question: “Is Mobile Health the Future?”



booz&co.

M-HEALTH

The emerging field of mobile health (m-health) has enabled consumers to use smartphone technology to answer their own health-related questions with the quick tap of a touch screen. Here's a look at m-health and the impact it's making on our culture and our well-being:

PHYSICIAN USE OF MOBILE TECHNOLOGY

DOCTORS ❤ TABLETS

250% MORE LIKELY TO OWN A TABLET THAN ANY OTHER CONSUMERS

66% USE TABLETS FOR MEDICAL PURPOSES

MORE THAN HALF FIND THEY EXPEDITE DECISION MAKING

40% REPORT MOBILE DEVICES DECREASE TIME SPENT ON ADMINISTRATION

1 BILLION visits to physicians' offices were made by Americans in 2011

78% of U.S. consumers are interested in m-health solutions

~85% of doctors use smartphones and medical apps

There are more than 10,000 medical/healthcare apps *available in Apple's iTunes App Store*

56% of doctors who use mobile devices say they expedite decision making

40% say they decrease time spent on administration

Only 28% of smartphone users and 18% of tablet users describe themselves as “very satisfied” with the quality of medical apps currently available

U.S. PHYSICIAN SMARTPHONE ADOPTION

IN 2012 85% OF U.S. PHYSICIANS OWNED OR USED ANY SMARTPHONE PROFESSIONALLY

It's the third-fastest growing app category for both iPhone and Android phones

88% of doctors would like their patients to monitor their health at home, particularly their weight, blood sugar, and vital signs

ONLINE PATIENT–PHYSICIAN COMMUNICATION

39% of U.S. physicians communicate online with patients via email, secure messaging system, instant messaging, or online videoconferencing

PATIENT USE OF MOBILE TECHNOLOGY

13% have accessed, stored, or transmitted personal health information or records in the past year 48% are interested in doing so

52% say they would like access to tools or websites that enable them to review quality rankings, satisfaction rankings, and patient reviews for specific doctors and hospitals

52% are comfortable in consulting with their physician through a video connection

86% of clinicians—doctors, nurses, and nurse practitioners—now use smartphones in their practice areas every day, up from 78 percent in 2012

53% of all clinicians use tablets daily, up from 34 percent in 2012

Source: IDC, Gartner, Forrester, Frost & Sullivan, YouGov, comScore, CISCO Global Cloud Index 2013, CISCO Visual Networking Index Study 2013, Intel, AMD, IMS Research, Manhattan Research 2012, Rock Health Digital Funding Mid Year 2013, Paw Internet and American Life Project, Health Online 2013, Epocrates 2013 report, and American EHR Partners 2013 survey

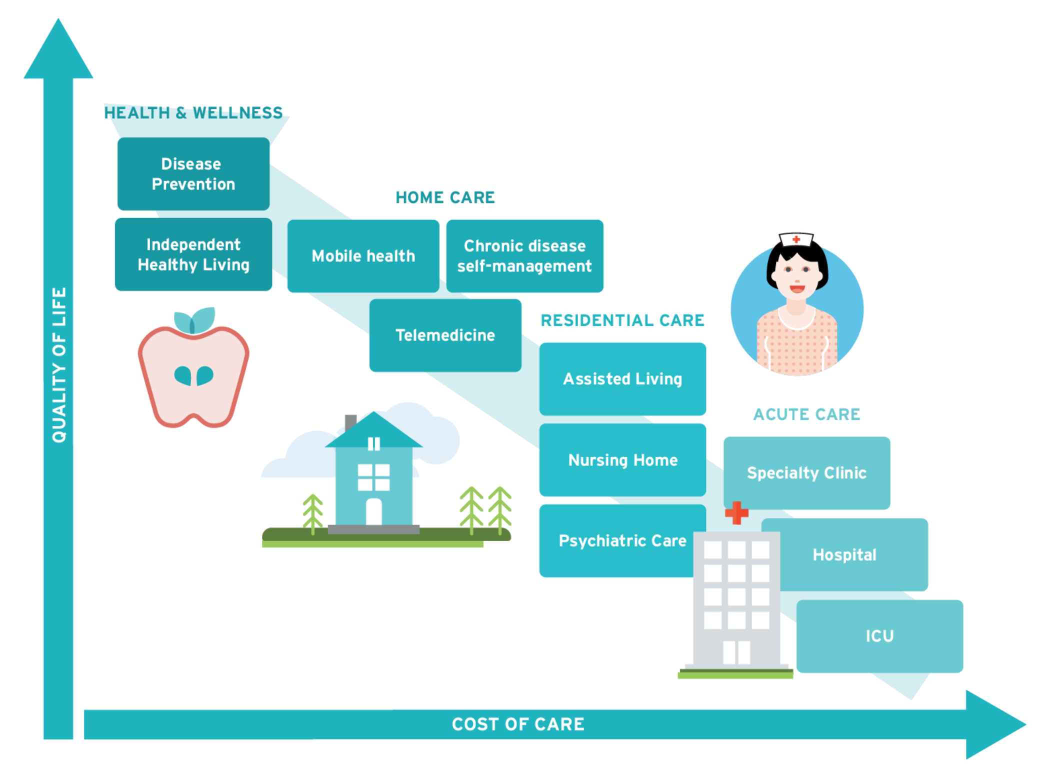
Copyright 2013 Booz & Company Inc. All rights reserved

Figure 1.1: Figures and Numbers Regarding M–health [1]

The above infographic for 2013 (see Figure 1.1) shows that more than 80 per cent of physicians in the US owned or used a mobile device professionally (with an increase of 6 per cent year on year) and more than half of them found that using mobile devices accelerated their decisions, while 40 per cent reported that it decreased administration time. With regard to communication, almost 40 per cent of physicians communicated online with patients, such as by using email, messaging and online video calls. In addition, 88 per cent of doctors would like their patients to monitor their health at home, such as their weight, blood sugar and vital signs.

The figure also shows that around 80 per cent of US consumers were interested in m–health solutions. In order to cope with this demand, more than 10,000 medical apps are available in Apple stores, which is the third–fastest–growing app category for both iPhone and Android devices. Thirteen per cent of patients had accessed, stored, or transmitted personal health information in the past year, whereas 48 per cent were interested in doing so. Interestingly, more than 50 per cent were comfortable consulting their physician through a video connection.

M–health solutions can play an important role in the idea of aiming to deliver care to the patient, instead of a patient going to the care as traditionally happened [5]. Figure 1.2 shows that m–health can reduce costs as well as improve quality of life.

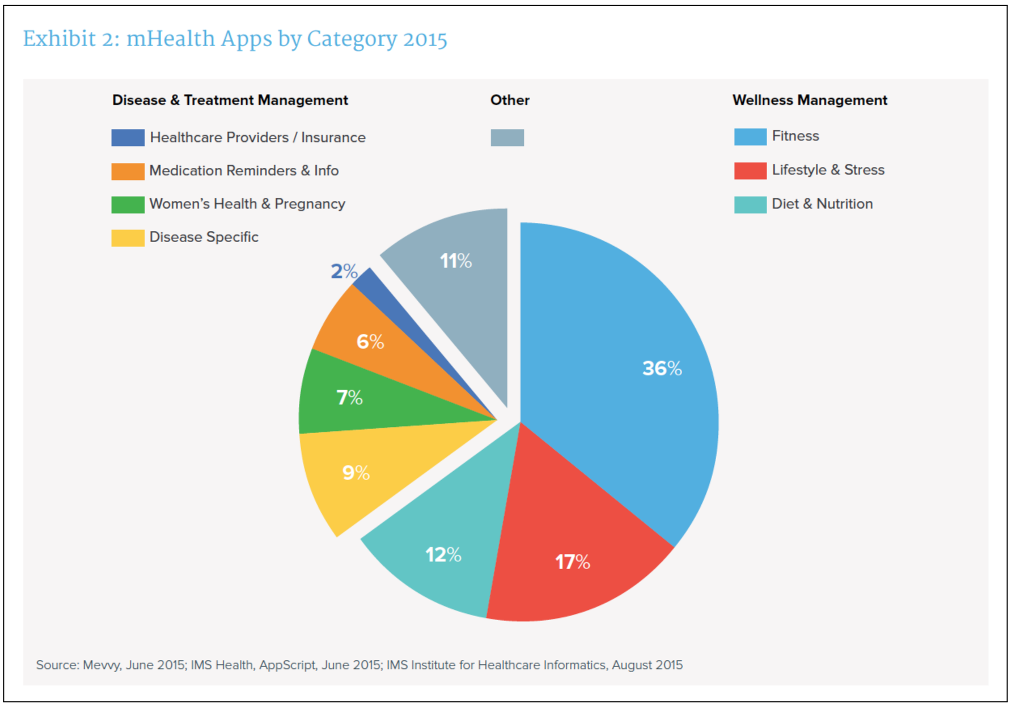


Source: Intel Across Healthcare 2013, MaRS Market Intelligence 2014

Figure 1.2: ‘SHIFTING LEFT’ to Reduce Cost and Improve Quality of Life [2]

Examples of m–health services could be creating electronic medical records (EMR) and making them accessible by mobile devices, establishing a communication link (e.g., a video session) between a patient who is facing a medical issue and a doctor who can provide health support to that patient, or retrieving useful information in the case of an emergency, such as the shortest route to the nearest medical centre.

A recent report [3] shows that the total number of m–health apps has increased, more than doubling over the past two years. Around two–thirds of the apps available concentrate on general health issues, such as fitness, lifestyle, stress, and diet, while the rest focus on other aspects, such as specific health cases, relevant medication, and women's health and pregnancy (Figure 1.3).



Source: Mevvy, June 2015; IMS Health, AppScript, June 2015; IMS Institute for Healthcare Informatics, August 2015

Figure 1.3: M–health Apps by Category [3]

1.2 Integration of Mobile Cloud to Enhance M–Health

Mobile devices have restricted computational and storage capacity and run on batteries that have limited power. These limitations render mobile devices unable to run the demanding tasks that may be required for accessing health services. Therefore, a solution is needed to allow users to utilize their preferred devices, such as smartphones and tablets, to access or deliver health services with complex and highly computationally heavy tasks being executed, not solely on mobile devices, but also on other devices and in different places. The most suitable solution is to shift heavy computational tasks to remote devices that offer better performance and then return the results to mobile devices without the latter having to do the processing.

This is exactly what happens in mobile cloud computing: a health service or application is hosted in a cloud centred on the large data repositories of hospitals, and mobile devices use this service remotely over the Internet, whereby the cloud executes requests and allocates the resources needed to achieve the results (Figure 1.4).

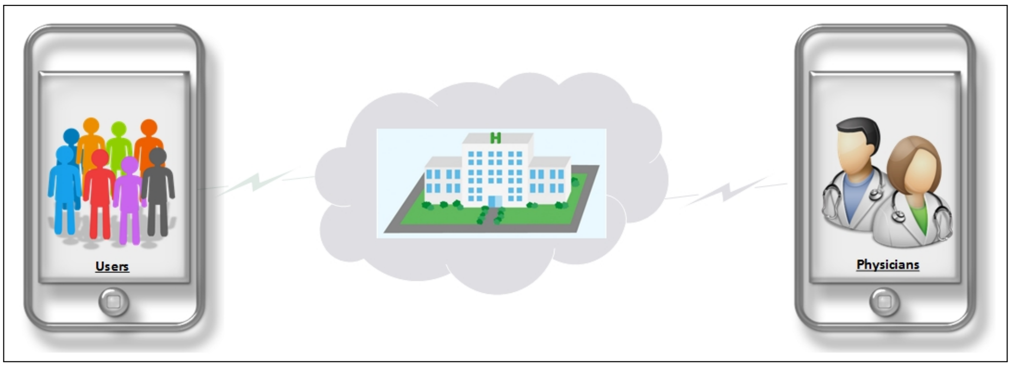


Figure 1.4: Mobile Cloud Platform Hosted by a Hospital and Accessible by Mobile Users

In relation to m–health, the mobile cloud can help health providers to cope smartly with consumers' needs, such as delivering health services swiftly as well as ensuring the smooth management of users' data and accounts. According to a recent study by Philips [6] (the official health technology sponsor of SXSW Interactive 2015), cloud computing adoption in healthcare is estimated to raise by 20 per cent annually until 2017, when it will reach a market size of $5.4 billion.

Introducing mobile cloud computing solutions into the health sector is not a straightforward solution, however, because it has specific limitations, such as disconnection issues and concerns over privacy and security. Therefore, this thesis examines these kinds of limitations and discusses the possibility of introducing mobile cloud computing technology into the health sector, with the aim of providing a robust, trustworthy system that can deliver healthcare services in an easy and reliable way to mobile users in emergencies.

1.3 Motivation and Research Goals

This section describes the motivation and goals of the research. A number of topics are also mentioned that are considered as relevant to the research.

1.3.1 Motivation

People are now using mobile devices to access health services more than they did before. In addition, health providers are aiming at delivering healthcare services to people on the move to reduce pressure on hospitals and other health centres. Moreover, providing health support and information at people's fingertips will improve their quality of life, which should reduce costs resulting from dealing with diseases such as diabetes and heart–related issues.

However, this improvement can also be essential, in some cases, such as in cases of emergencies, where a life can be saved if swift care is provided. Furthermore, better results can be gained if a fast but well–directed action has been taken.

In some cases, the nature of the location where an accident has taken place can present a further challenge, such as a user not being able to reach a health department to report an accident because of being in a remote area with no Internet or cellular coverage.

Overall, integration of the mobile cloud in the health sector, particularly m–health, has been widely seen as a promising solution for delivering healthcare support to people anytime anywhere, as well as dealing with most issues concerned with mobile devices, such as limited computational and storage capability. Furthermore, introducing the concept of point–of–care will improve the delivery of healthcare support to people who are in need or facing an emergency, by delivering healthcare support to the public 'on the move' to deal with the mobility issue and respond swiftly to increase the chance of survival.

These advantages have motivated this research in the development of innovative software architecture to provide *fast response* healthcare support *on the move* with the help of the mobile cloud to people in emergency situations using the concept of point–of–care.

1.3.2 Research goals

- To determine a means of enhancing the delivery of healthcare information (or advice) on mobile devices.
- To expand the point–of–care concept to everybody, anywhere, when they are in distress.

- To identify how care can be mobilized for people in emergency situations wherever these occur, as well as notifying health providers and establishing communication links (if needed) between all the parties concerned in delivering health services.
- To design and deploy a robust and secure mobile cloud system that can assist people in medical emergencies wherever and whenever they experience problems, as direct access to medical professionals at any time is key to this system.

1.4 Challenges and Solutions

Designing and implementing a system that delivers healthcare services on the move generates a number of challenges. These challenges can be divided into three main groups, which are detailed below. Some of these challenges become as requirements for the system proposed in this thesis, as shown in chapter 2.

1.4.1 Challenges related to technologies in use

Mobile cloud computing technology was introduced to offer better outcomes (such as high performance and high availability) to healthcare services. In fact, cloud computing still suffers from certain shortcomings, such as issues related to security, privacy, reliability, and availability. However, security and privacy are the most important issues because the technology is required to deal with people's medical data, which are very sensitive and have to be protected and secured.

Therefore, the proposed system has to provide a strong security mechanism that ensures that the content of medical data is stored securely in the cloud using high–level protection strategies, such as data encryption. The system also needs to ensure that both cloud services and stored data can only be accessed by appropriate/authorized people to minimize the possibility of attack. This requires ensuring access to the system using a high–level authentication technique, such as a one–time password, an SMS–authentication scheme or one of the biometric authentication methods, which offers a high degree of protection.

1.4.2 Challenges related to users who participate in the system

In addition to users who are seeking health services and want to ensure their sensitive medical data are protected and secured, there are other kinds of users: those who provide health services to the public and have specific requirements to help them deliver care or support to people in emergencies. These requirements include ease of use, fast response, and high availability. It is also understood that providing health support to the public using an out–of–hospital method places more pressure on healthcare professionals [7].

Using the concept of volunteering is a key solution to this issue, whereby a medical professional, or anyone who is qualified, authorized and able to deliver healthcare to the public, has the choice of participating or not. The system also has to allow medical professionals to change their status if required; for example, from “Available” to “Offline” or “Dealing with a case”.

Finally, the system has to have a form of management that ensures all participating users are treated equally in the case of redirecting medical requests. Carrying out a survey to discover medical professionals’ views on using such a system could also be useful.

1.4.3 Challenges related to the collected data

In order to provide a healthcare service to someone, certain data have to be collected, such as current location, medical history (including body monitoring data, e.g., heart rate and blood pressure), and personal information (e.g., name and age). This raises ethical issues regarding how this type of information will be used and who will access the data.

Collecting this type of information is critical for the proposed system to provide proper help to that person. For example, the system has to be aware of any allergies or special medical requirements (e.g., heart–related issues or asthma) before redirecting the case to an available medical professional.

Another example is related to tracking technology, as the system has to identify the current location of the person seeking help in order to connect him/her to medical staff who are in the vicinity. Furthermore, the system has to track medical professionals in order to redirect emergency cases that are close to their location.

Therefore, collecting this type of data is very important and might even be life–saving. However, the system also has to provide a high level of confidence to users that their data will only be used for medical purposes and they have the right to accept or reject this type of information being shared with other medical parties, such as hospitals or insurance companies.

1.5 Summary of Thesis Contributions

The following is a list of the contributions to the existing knowledge made by this thesis:

- An analysis of related works integrating the mobile cloud computing paradigm in healthcare services (e.g. m–health).
- A mobile cloud middleware solution, which enhances the connectivity and reliability of mobile cloud healthcare applications. The main idea of this middleware is that it allows users to create and join mobile ad–hoc networks (MANETs) in order to use them in emergencies, whereby the cloud takes on the role of managing these networks, such as Internet Protocol (IP) allocation and resource dissemination.

  However, this model can also be extended to help sessions that are executed between MANET members to be registered, saved and resumed when needed for the purpose of saving the progress of such a session if one of its parties suddenly leaves the network. As a result, the session is resumed from the point at which the interruption occurred, which, in turn, leads to better usage of mobile device resources, particularly battery life.
- A novel mobile cloud service that offers people who are experiencing medical emergencies while on the move the possibility to ‘look up’ doctors or nurses who are located within their proximity and who can respond more quickly than the emergency services. Furthermore, this service establishes the details of a communication link that is managed by the cloud, as well as starting a chat session to exchange text, photographs, and files. SMS messaging ability is also provided as an alternative method for reaching cloud services and seeking medical help in emergencies.
- The design of another mobile cloud service that makes the best use of social media applications, such as Twitter, in emergency and risk management.

Risk and emergency teams can receive in a matter of seconds data that can inform their decisions when an emergency has affected areas under their management. This service allows users to provide on–the–ground information regarding such an event, as well as early notification to people who are in the vicinity of an emergency situation. Users' requests are matched to a set of pre–defined labels that will help rescuers gain a clearer understanding of the situation.

1.6 Thesis Structure

The remainder of the thesis is organized as follows.

- Chapter 2, "Motivation Scenarios and Requirements", outlines a number of real–world scenarios to clarify the concepts behind this research, model the system architecture and assess the benefits people may gain from it. This chapter also provides what is needed to enhance users' experience in emergencies.
- Chapter 3, "Background and Related Research", presents relevant background and related work, as well as an overview of the topics considered and the goals of this work.
- Chapter 4, "Introducing Mobile Cloud Technology into M–Health to Deliver Better Care/Support in the Case of Emergencies", describes the researcher's initial approach to introducing mobile cloud technology into m–health in order to deliver better care/support in the case of emergencies.[1]
- Chapter 5, "Managing a Mobile Ad–hoc Network in the cloud to support m–health applications", presents the MANET—cloud model, which aims to provide robust and reliable connections to mobile users in the case of emergencies by assigning the management role of MANET networks to the cloud, such as IP allocations and monitoring members' connectivity. An illustration of a framework that allows active sessions between MANET members to be saved and resumed is also presented.

[1] An article was published in a non–peer–reviewed publication that was written for "outreach" purposes. The article was entitled "Introducing mobile cloud technology into m–health to deliver better care/support in case of emergencies" [8] and was published in the Boolean [9] magazine for UCC. The Boolean is an annual collection of short papers in which doctoral students describe their area of research and some of their main findings. These articles are journalistic in nature and are written to be accessible to a non–specialist audience.

Elements of this chapter were published in:

  - “Mobile ad–hoc network management in the Cloud” [10], which was presented at the International Symposium on Parallel and Distributed Computing (ISPDC) in Porquerolles in France in June 2014.
  - “Robust cloud management of MANET checkpoint sessions” [11], which was presented at the ISPDC’14 in Limassol in Cyprus in June 2015 and a journal version [12] which was published in Concurrency and Computation: Practice and Experience in 2016.

- Chapter 6, “Emergency Help Services”, introduces two mobile cloud services that allow help to be sought from the cloud in the case of emergencies, based on the aforementioned MANET–cloud model. Parts of this chapter were published as:

  - a conference paper [13] that was presented at an international conference on cloud computing technologies and applications (CloudTech’15) in Marrakesh in Morocco in June 2015 and a journal version that has been accepted (but not yet published) by the *International Journal of High Performance Computing and Networking.*
  - another conference paper that was presented at the ISPDC’15 in Fuzhou in China in July 2016.

    An extended journal version of the contents of this paper is currently under development and will be submitted to *Concurrency and Computation: Practice and Experience* at a later date.

- Chapter 7, “The Reference Architecture”, matches the design and implementation of the proposed system (including the MANET–cloud model and the two mobile services) with previously presented scenarios/applications, including how the new framework could make an improvement in each situation. Furthermore, discuses the reference architecture and revisits the requirements that were defined on chapter 2.

- Conclusions and future work are presented in Chapter 8.

1.7 Publications

The chapters of this thesis are composed of the following publications:

1. Hazzaa Naif Alshareef and Dan Grigoras. Using Social Media and the Mobile Cloud to Enhance Emergency and Risk Management. Proceedings of the IEEE 16th International Symposium on Parallel and Distributed Computing (ISPDC), pp. 92–99, July 2016.

2. Hazzaa Naif Alshareef and Dan Grigoras. Swift personal emergency help facilitated by the mobile cloud. Int. J. High Performance Computing and Networking, “Accepted for publication, June 2016”.

3. Hazzaa Naif Alshareef and Dan Grigoras. Robust cloud management of MANET checkpoint sessions (extended). Concurrency and Computation: Practice and Experience, 2016: doi: 10.1002/cpe.3816.

4. Hazzaa Naif Alshareef and Dan Grigoras. First responder help facilitated by the mobile cloud. Proceedings of the IEEE International Conference on Cloud Computing Technologies and Applications (CLOUDTECH), pp. 1–8, June 2015.

5. Hazzaa Naif Alshareef and Dan Grigoras. Robust cloud management of MANET checkpoint sessions. Proceedings of the IEEE 14th International Symposium on Parallel and Distributed Computing (ISPDC), pp. 66–73, June 2015.

6. Hazzaa Naif Alshareef and Dan Grigoras. Mobile ad–hoc network management in the cloud. Proceedings of the IEEE 13th International Symposium on Parallel and Distributed Computing (ISPDC 13), pp. 140–147, June 2014.

Chapter 2

Motivation Scenarios and Requirements

2.1 Introduction

This chapter presents a number of scenarios for events that might occur in people's day–to–day lives. This chapter also defines a number of requirements based on these scenarios that will need to be considered in the proposed system design and implementation, which are presented in chapter 4. The roles and services of the system's main users, such as mobile devices and cloud computing, are also discussed.

2.2 Scenarios

The idea behind presenting the following scenarios is to clarify the concepts of this research, model the system architecture and assess the benefits people may gain from the proposed system.

2.2.1 Remote areas

In some countries, such as Saudi Arabia, driving along highways might be dangerous because of the long distances involved or the quality of the roads. Therefore, travelling between cities by car can result in emergency cases, particularly in remote areas.

One scenario is that of an individual driving along a highway; he/she has an accident, in which his/her leg is broken. Unfortunately, he/she is in an area in which there is no health centre nearby. However, he/she can use his/her mobile device to get help. An example of help could be notifying people nearby, requesting medical information on how to treat emergency situations, or reporting the accident to a road safety department or ambulance service. In addition, if someone discovers the accident, the passer–by may be offered a procedure on how to deliver appropriate help, even if he/she is not a medical professional.

In these types of situation, a system that allows users to reach the medical services using a mobile device is required, while also taking into account the poor network coverage that might exist. The system should also offer a way of notifying other drivers/people on the same road who could provide help, as well as being able to request medical assistance from the nearest emergency centre or provide directions to that centre.

2.2.2 Crowded places

When a large number of people are present in the same place to perform the same activity, such as watching a football match or attending a concert or religious event, there is a strong chance that emergency medical cases will occur. The following two sections present two such scenarios.

2.2.2.1 Al–Hajj (pilgrimage to Mecca)

In this project, Al–Hajj (pilgrimage to Mecca) is considered as such an event. Al–Hajj is a religious duty, whereby every able Muslim is obliged to perform a pilgrimage to Mecca at least once in his/her life [129]. In this event, hundreds of thousands of people are present in the same area for a short period, performing virtually the same actions [130].

There has been a review [131] of the most common health issues that occur during the Al–Hajj event. Some of these issues are related to the medical history of the pilgrims and others can occur because of the environment and the particular location [132]. Some researchers [133] have analysed the difficulties and risks related to Al–Hajj in general and designed a system to deal with some of these issues. Another paper [134] presents a mobile–based tracking service to help pilgrims in emergencies.

For example, imagine that someone who has a certain type of medical history, such as heart disease, requires healthcare urgently. In a crowded place, reaching such a person using the regular emergency services will be difficult or might even, in some cases, be impossible.

However, a doctor or nurse might be in the crowd not far from that individual and, therefore, could immediately be available to help. Alternatively, the person in need, or someone around him/her, could use a mobile device to obtain directions to the nearest medical centre. Therefore, this kind of system is able to link users who are looking for care with those who are able to provide it, and is potentially life–saving.

2.2.2.2 Shopping centres

Another scenario related to people's daily activities is that of visiting shopping centres. Emergency cases might occur in such places, particularly in relation to children. For example, a child might lose his/her family or injure him/herself, prompting the need for an urgent response. Another example could be related to the failure of the electricity or lighting systems of these places, whereby a large number of people may be involved and require help, such as needing directions to the nearest exit.

In this scenario, a system that allows parents to track their children's location in the case of losing them in the darkness and resulting confusion, as well as establishing a communication link between them, is required.

Another feature that could be added to the system is the ability to disseminate useful information, such as directions and instructions that would help emergency crews to rescue people who are in distress.

2.2.3 Residential areas

Emergencies can also occur in the home, from cases of people falling downstairs to serious fires that can destroy the whole house. Assuming now that such an emergency has been detected in the home using a modern home–monitoring device.

Generally, this will trigger an alert to one of the emergency departments, such as the ambulance or firefighting services.

However, it might be that a professional in close proximity is available and can, therefore, provide help before an ambulance or fire engine arrives.

Therefore, a multi–notification system is needed: one that automatically searches for doctors or nurses in the vicinity of an emergency situation whilst also providing useful information such as the shortest route to the location. This notification and triggering of an alert at the relevant emergency centre would happen simultaneously.

2.3 Requirements

This section starts by defining the role of the system's users and target services. The system requirements are then presented and discussed.

2.3.1 Roles

2.3.1.1 Mobile devices

Users can use mobile devices to:

- Request healthcare services from the cloud, such as searching for a doctor who is near his/her location.
- Set up a MANET to share services and data and maintain robust communication with the cloud.

2.3.1.2 The cloud

Cloud computing resources are used here for two main purposes:

- Hosting and providing healthcare services, such as simple medical information, notifying relevant departments and so on.
- Managing a MANET and keeping communication active.

2.3.2 Target services

Mobile devices have useful functions and services that could help in delivering and supporting healthcare systems.

For instance, location coordinates collected by mobile devices could be considered as one of the most important services that are specific to mobile devices and can play an important role in delivering a healthcare service, particularly in cases of emergency.

In addition, elements or services such as microphones (e.g., for voice recording), cameras (for taking photographs or recording a video), and storage can also be involved in delivering healthcare services in emergencies.

Smartphones and tablets could also be used here, as well as their built–in sensors, such as the device's use of fingerprints for authentication and identification purposes. Another example is the use of heart rate and gravity sensors to allow the collection of initial users' medical data in the case of an emergency.

Alternatively, these built–in sensors could be used to trigger an alert if the user is experiencing an emergency and cannot request a service manually, such as sending a request to an emergency department if the phone of an elderly user has detected a fall using gravity sensors.

The storage ability/service of the cloud would mainly be used in delivering health support and care to people in emergency cases. For instance, medical reports could be saved in the cloud where an update or access to these records would be important in an emergency and would lead to better understanding of the case of the user who requested aid.

However, the strong computing ability of the cloud itself could also be used. In other words, heavy and long tasks could be shifted to the cloud instead of executing them in mobile devices, which would save mobile device resources such as battery life.

2.3.3 System requirements

This section discusses the functional and non–functional requirements of the proposed system.

2.3.3.1 Functional requirements

- The system should provide, via different network technologies (e.g., cellular and Wi–Fi), access to its services, as well as using SMS messaging between users and the system.

This will help users to reach the proposed services using a low–cost link, such as over free Wi–Fi Internet access. On the other side, the system can be reached using SMS messaging if no Internet connection is available. As a result, the availability of and ability to access the system will be increased.

- The system should support live conversations between users, such as voice or video communication. This would be useful in the case of delivering health advice to a user in a remote area, for example.

  Another example could be the system allowing live communication between experts who are involved in delivering care to an emergency case.

- The system should support real–time interactions. This is one of the most important requirements of such a system when planning to deal with emergency medical cases because users who are in an emergency situation need a quick response from the nearest emergency department. Any delay in the response to a call for help might lead to a lower chance of survival or make the case worse.

- The system should support different forms of multimedia (e.g., text, photographs, voice, and videos) to enable better understanding in the case of sending health advice to a user facing an emergency.

  For instance, the system could send a video demonstrating what the user needs to do to stop bleeding. On the other side, a user who requests aid can take a picture of what he/she is experiencing and send it to the system to get help, such as in the event of a road traffic accident.

- The system should deliver useful information in emergency circumstances, such as emergency telephone numbers depending on the user's location (e.g., 999 or 911). It will be more valuable if the system enables communication between someone who can deliver aid and another user who needs it, such a father who is trying to help his/her child.

  In this case, useful information can be delivered while waiting for emergency help to arrive, such as an ambulance and its crew.

- The system should involve health providers (e.g., doctors, hospitals and ambulances) if they are needed to ensure that a high level of healthcare will be delivered, as well as reducing the mistakes that may result from emergency situations, such as when unqualified individuals try to provide help which may cause further harm to the injured user or medical case.

In general, all emergency cases should be redirected to health centres such hospitals or emergency departments as soon as these cases are detected by the system or received as a request from a user.

- The system should store user details, including name, date of birth (DOB), current location, state and medical history. One example is to implement an EMR solution as part of the system. Therefore, the system should maintain all users' medical–related data, such as laboratory results (e.g., X–ray or blood test results) and medical history, as well as personal details to be used whenever they are needed. For example, if a user requests aid from the system, the system can redirect the request to the nearest medical centre and attach the user's personal and medical data for the purpose of medical professionals being aware of any allergies or past medical issues, such as heart–related problems.

- The system should contain information relating to medical centres, including name, address, availability, and speciality. All of this information should be organized in the system database to allow access when required. For instance, if a user faces an emergency case while he/she is driving on a highway, the user can retrieve information regarding the nearest medical centre from the system, such as its location on a map, telephone number or even start live communication (e.g., a video call).

- The system should validate doctors' details with the relevant health department. This will be part of the level of care that the system guarantees to deliver to its users. In other words, only qualified and trusted medical staff are allowed to provide healthcare or support to the public. For example, if a user is in a crowded place, such as a football match, feels unwell and wants to discuss this concern with a doctor or nurse, the system can help provide details of a volunteer doctor or nurse to that user, with the ability to start a communication session between the two parties. This example can be very attractive if the chosen doctor or nurse is located just a few steps from the user who makes the request.

- The system should allow users to create peer–to–peer networks, such as a MANET, for the purposes of allowing users to share a service and content locally and making the system robust. In the case of crowded places in particular, the ability to reach the system services might not be available or not possible for all users.

Therefore, users can either reach the system services through neighbouring links or request help/aid from another member of the MANET.

- The system should monitor the MANET's usage and its members' behaviour for two main reasons:
  - To ensure the network will not be used in the form of an attack that slows the entire system.
  - To allocate resources to each network as soon as it is active and release any of the resources allocated to terminated networks.
- The system should gather information about every member of each MANET, including link strength, mobility level, and the number of one–hop neighbours. This will help the system to communicate and choose the best link in terms of quality, whereby this link will be used to reach other members if doing so is impossible to do directly. Another benefit could be finding missing users in a crowed place, such as during Al–Hajj, enabling pilgrims to be located or notified based on the information collected by the system.
- The system should manage MANETs effectively and smartly, such as by merging networks that have the same purpose or are located in the same area or splitting a MANET that is congested and has a high number of users, as splitting operations will reduce traffic and allow smooth interactions between members.

2.3.3.2 Non–functional requirements

Responsiveness: the system should react immediately to users' requests to avoid any further harm occurring to them or those involved in an emergency situation.

Scalability: the system should handle any increases/decreases in workload efficiently, such as managing a high number of active MANETs. In general, the system should take further action, such as merging two or more networks into one MANET or splitting a busy network into two or more separate MANETs.

Security: the system should deploy robust security mechanisms, including:

- Protecting users' data by introducing strong encryption schemes.

- Monitoring users' behaviour for the early detection of possible attacks.

Reliability: the system should be reliable in terms of providing more ways of accessing the deployed services, such as through the use of a Wi–Fi or cellular network.

Privacy: the system should protect users' data from any unauthorized access.

Accessibility and availability: the system should be accessible and available anywhere, anytime, using mobile devices.

Usability: the system should support multiple languages (internationalization), which will be useful in reducing the pressure on people who can request aid and communicate using the system's main language. This will be feasible in some scenarios, such as a football match or a religious event, as users of many nationalities might attend.

Quality: the system should deliver relevant medical information in a manner suited to the requested material and guarantee a high level of care and support to its users.

Resource use minimization: (e.g., battery life and data) the system should take into account that most users are mobile. Therefore, minimization strategies should be considered at all times, such as reducing the size of images and videos sent.

Trustworthiness: the system should ensure that all the medical users who register are trustworthy and qualified to deliver medical help to the public. This will reduce the possibility of medical errors, as well as increase the level of trust in the whole system.

Data integrity: the system should ensure referential integrity in medical data, particularly data that are intended to be exchanged/transferred over intermediate nodes.

Capacity: the system should cope with the large number of requests/transactions coming from a high number of users.

2.4 Chapter Summary

From the scenarios outlined in this chapter, it is clear that there is a need to design a system that delivers healthcare services on the go for emergency situations.

Considering these scenarios helped in the design as well as in outlining the challenges that might occur for the proposed system.

In addition to the functional and non–functional requirements mentioned previously, the system has to take into account a number of important requirements.

First, providing reliable connections between mobile devices and the system (e.g., the cloud), as well as connections between the users themselves, such as the one between a user who requests emergency help and a doctor who was chosen to provide that help.

Second, ensuring connections are secure enough and meet users' preferences, as well as offering protection from attacks, such as attempted access by an unauthorized person.

Third, responding to emergencies has to be achieved quickly; therefore, the system should ensure that real–time and fast communication is provided to users.

Fourth, since dealing with emergencies and medical cases requires collaboration between medical parties such as doctors and hospitals, as well as the dissemination of medical data or any other useful information that could be involved in delivering help to people who need it, the system should allow collaboration between people who are involved in providing medical support. The system should also provide an easy mechanism for the dissemination of data and information that could lead to better understanding of a given emergency or to better decision making.

To summarize, the information in this chapter played an important role in the system design that is presented in the next chapter and is used as a means of reference to check the middleware and services proposed in chapter 7.

The next chapter reviews existing technologies and related projects for providing a clearer path that leads to better designing of a middleware system that is aimed at enhancing healthcare services using mobile cloud services, as is discussed in detail in chapter 4.

Chapter 3

Background and Related Research

3.1 Introduction

Delivering medical aid to the public in the case of an emergency will save people's lives, or at least ensure that no further harm comes to them. To achieve this goal, existing technologies and projects have to be reviewed in order to build a path that leads to a mature design that can fit efficiently into real–world scenarios. This chapter is divided into three main parts, as detailed in the following list.

1. Background information about mobile cloud computing (MCC) technology (known as the mobile cloud) is presented, highlighting the limitations of mobile devices and how the cloud infrastructure could help reduce these drawbacks. This section attempts to answer the following question: *"What is the mobile cloud?"*

2. Next, the m–health model is discussed, outlining its limitations and some of the existing systems, as well as responding to the following questions: *"How can a mobile cloud solution help?"* and *"What is the relationship between m–health and the mobile cloud?"*

3. Two related technologies are then presented and discussed:

   (a) Mobile ad–hoc networks (MANETs), including management issues and relevant existing systems. The main reason for presenting a MANET is because it is the basis for developing the infrastructure for the proposed system and can enhance the reliability and availability of the whole system. In other words, the aim is to create a robust infrastructure that would permit communication when direct communication between the mobile device and the cloud is not possible.

(b) Social media networks, as the usage of this type of network has increased recently in the case of delivering care and support in emergencies. The purpose of reviewing this type of application is to develop the service that is part of the proposed system and integrates social media networks to improve the delivery of healthcare support in the case of emergencies.

3.2 Mobile Cloud Computing

Two areas first have to be reviewed and discussed (mobile computing and cloud infrastructure) before reviewing MCC. The following sections explore these two areas and provide a comprehensive understanding of their limitations and benefits.

3.2.1 Mobile computing

Mobile computing (MC) technology uses mobile devices (such as mobile phones, tablets and personal digital assistants [PDAs]) anytime, anywhere to access/deploy services or applications. According to Johnson and Maltz, *"Mobile computing is a technology that allows for the transmission of data, voice, and video via a computer or any other wireless enabled device without having to be connected to a fixed physical link"* (p. 2) [14].

Using mobile devices to seek services over a wireless link brings benefits to its users, including communication (anytime, anywhere), entertainment (e.g., gaming), safety (e.g., delivering support in emergencies), and easy access to information and the Internet [15] [16].

However, there are some limitations to using this technology. The most noticeable limitations are lack of security, extra cost (e.g., accessing the Internet over cellular links), availability, reliability, and limited resources (e.g., battery life, storage and computing capacity) [17].

3.2.2 Cloud computing

Cloud computing (CC) [18] is a new technology that is usually based on data centres with high levels of computing and storage capacity. It has recently attracted both new developers and the owners of existing systems and, furthermore, represents a new direction and the next generation of the IT industry. For example, *"This model creates a brand new opportunity for enterprises"* (p. 75) [19].

In one paper [20], more than 20 different definitions of cloud computing from a variety of sources are compared in order to produce a more mature definition of CC. However, it is believed that the most efficient definition of CC is that provided by the National Institute of Standards and Technology (NIST): *"Cloud computing is a model for enabling ubiquitous, convenient, on–demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"* (p. 6) [21].

There are advantages to deploying this technology, which include high availability, scalability, easy access (over the Internet), and the reduction of risks (e.g., hardware failures) and maintenance expenses [22]. In addition, it has an elastic (pay–as–used) model, whereby users pay for what they have used with the ability to extend services as they need them, particularly when they are on the move [23]. On the other hand, there are some issues with using this technology. The most important issues are lack of privacy, security, and data integrity [24].

Choosing the most appropriate cloud design will lead to greater benefits for a company that is about to implement cloud infrastructure instead of building an IT environment independently. Therefore, an understanding of the available models and services is required.

One paper [25] discusses cloud computing architecture, which is based on four layers:

1. *Hardware layer:* responsible for handling hardware resources (servers, routers, switches, power and cooling systems).

2. *Infrastructure/virtualization layer:* responsible for constructing a pool of computing and storage resources.

3. *Platform layer:* responsible for offering better performance when applications are directly installed on virtual machines (VMs).

4. *Application layer:* responsible for managing cloud applications. These applications help the cloud to improve performance and availability, as well as reducing operating costs.

Moving to deployment choices, the cloud can be deployed as one of the following:

1. *Infrastructure as a Service (IaaS):* delivering all the infrastructure needs of subscribers.

2. *Platform as a Service (PaaS):* providing operating system management and software development frameworks.

3. *Software as a Service (SaaS):* allowing access to applications over the Internet.

Interestingly, this paper [25] describes these options from a business perspective. The authors combine the first two services (IaaS and PaaS) into one group, as these are delivered by the same provider.

With regard to cloud types, there are three main types of cloud computing [26]:

- ***Public cloud*** accessible to anyone. There are issues with using this type of cloud, such as lack of security.

- ***Private/internal cloud*** suitable for a single company. The main idea behind this is to solve the security issues that can be found in the public cloud.

- ***Hybrid cloud*** a combination of the two types above to solve some of their issues. The better the configuration of this type of cloud, the greater the benefit to users.

Amazon Web Services (AWS) has rolled out another type of cloud, called the Virtual Private Cloud (VPC), which is a mix of the public and private clouds. It is aimed at providing a secure and seamless bridge between an organization's existing IT infrastructure and the Amazon public cloud. This type of cloud is public because it still uses computing resources for the general public and virtually private because the connection between IT legacy and the cloud is secured through a virtual private network, and a set of 'isolated' resources for the VPC is dedicated to AWS.

3.2.3 Mobile cloud

From the literature, the definitions of MCC [27] can divided into two classes:

First: mobile device resources (storage and computing) are involved. This brings advantages, such as accessing multimedia and sensor data without the need for large network transfers, efficient access to data stored on other terminals, and spread ownership and maintenance hardware costs.

Second: shifting the storage and computing processes from mobile devices to cloud computing. This leads to solving mobile device limitations, such as saving battery life. Some authors, such as O'Sullivan and Grigoras [28], use this as a definition of MCC. This thesis also uses this definition.

The most comprehensive definition of MCC, based on the second definition above, is as follows: *"a model for transparent elastic augmentation of mobile device capabilities via ubiquitous wireless access to cloud storage and computing resources, with context–aware dynamic adjusting of offloading in respect to change in operating conditions, while preserving available sensing and interactivity capabilities of mobile devices"* (p. 1) [29].

3.2.3.1 Differences between the mobile cloud and the traditional cloud

From the two previous definitions of the mobile cloud, Table 3.1 clarifies two key differences between cloud computing and the mobile cloud:

Table 3.1: Differences Between CC and MCC

	Cloud computing	**Mobile Cloud Computing**
Aims	Delivering services globally without outlining where these services are kept or how services are provided.	Enhancing mobility, which means users can obtain services via mobile devices.
Resources	Usually based on personal computers (PCs) with high computing and storage ability.	Possibility of building clouds from mobile devices for data storage and computing.

3.2.3.2 Architecture

One paper [30] explains the main architecture for MCC in Figure 3.1, which shows that:

- Devices are connected to mobile networks over base stations, such as access points, satellites, or base transceiver stations (BTS). These stations are responsible for providing and managing communication between devices and networks.
- Mobile users' requests and information are sent to central processors that are attached to servers that deliver mobile network services.
- Mobile providers can manage mobile network services to provide services to users.
- Users' requests are then transmitted to the cloud over the Internet.

In the cloud, all requests are processed and the results sent back to the user who made the original request.

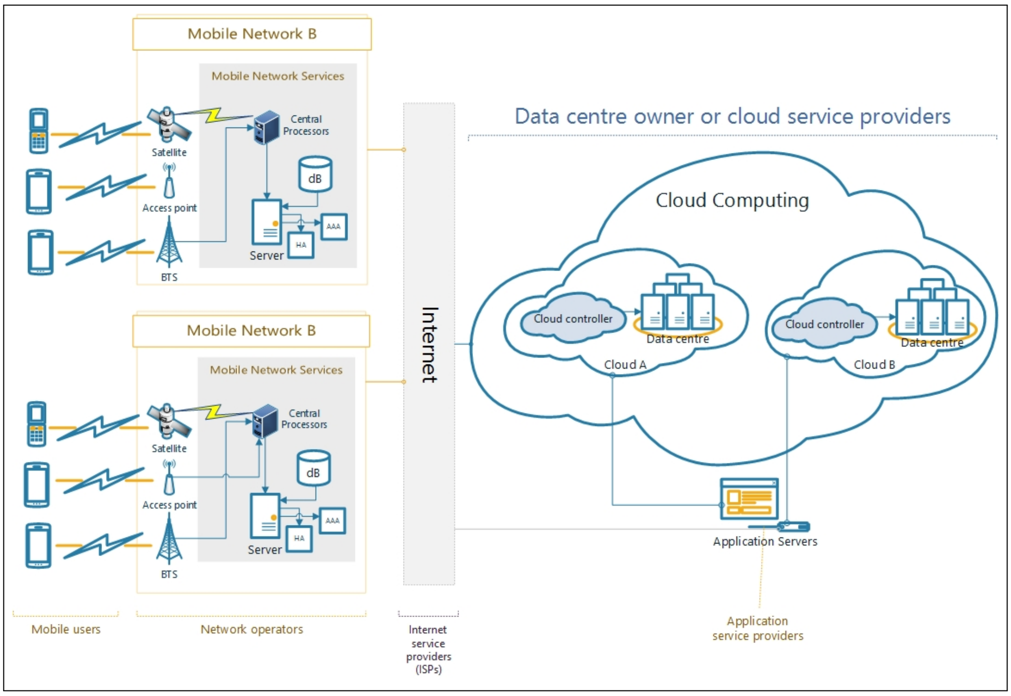


Figure 3.1: Mobile Cloud Computing Architecture

3.2.3.3 Challenges and benefits

One group of authors [31] have explored challenges that might arise with respect to the constituent parts of MCC (mobile computing and traditional computing) when MCC is implemented, such as issues concerning application and network requirements and the security of mobile computing.

With regard to cloud computing, the paper mentions security, lack of standards and performance as challenges that will have an impact on MCC implementation. Other authors [32] have considered the following challenges that MCC might face:

- Dealing with mobile device limitations, such as low capacity (in computing and storage) and limited battery life.
- A variety of mobile device operating systems, such as Android, iOS, and Windows Phones.
- Using mobile devices as cloud components will affect cloud performance.
- Addressing security attacks inside and outside the cloud [33].

However, the authors also list benefits that could be gained from deploying MCC, including:

Enhancing mobility: whereby MCC could improve users' experience by maintaining information about users' locations, contexts, and the services requested.

Sharing resources: members sharing their resources leads to resolving mobile device limitations. As a result, improvements in accessing applications and services are achieved.

Overcoming of some CC issues: an example of which is deploying CC with long wide area network (WAN) latencies. Issues are found when a mobile user executes a resource–intensive application on a distant high performance server. One solution could be instantiating customized services software on a nearby computer, then accessing this service over a wireless local area network (LAN).

3.2.3.4 Research directions

The above–mentioned paper [32] points out a number of open issues in MCC that need to be resolved, such as:

Low bandwidth: increased usage of MCC is inhibited by bandwidth limitation. Hence, optimal and efficient ways of allocating bandwidth should be in place. The paper presents two technologies that might improve and address the bandwidth issue: 4G and femtocells.

Network access management: a better network access mechanism will lead to an improvement in link performance and improve bandwidth usage. The authors mention cognitive radio as a solution, whereby unlicensed users can access a spectrum that is allocated to licensed users.

Quality of service: when mobile users are about to communicate with the cloud, they may face some difficulty, such as congestion, disconnection, and signal attenuation. The paper presents a number of possible proposals to solve these issues, such as CloneCloud [34].

Pricing: since MCC integrates both mobile services and cloud servers, this means that there are two different providers with different service and customer management, payment methods and prices. Therefore, a way of addressing price issues should be considered.

Standard interface: : a mainly web interface is used for communications between mobile users and the cloud. The paper finds that a web interface is not suitable because it is not designed for mobile devices and compatibility between different devices could be an issue. Therefore, a standard protocol should be developed. The authors anticipate that HTML5 might be a solution in the future.

Service convergence: because of the high usage of MCC, single cloud computing might not be enough to serve users' needs. Therefore, the idea of sky computing might be introduced. Sky computing is a computing model that groups more than one cloud computing provider to offer a large distributed infrastructure. Likewise, mobile sky computing might be introduced in the same way as CC and MCC.

3.2.3.5 Existing projects

Two of the most well–known existing projects (Hyrax [35] and VM–based cloudlets [36]) are presented in the following subsections.

Hyrax

Hyrax is a platform derived from Hadoop [37] as a MapReduce system [38]. It aims to allow the construction of a mobile cloud for Android smartphone applications with distributed data and computation.

In other words, the creation of a mobile cloud from Android–based smartphones for the purpose of dealing with the limited storage and computational ability of mobile devices.

Three technologies are used:

1. *MapReduce:* a programming model for processing and generating large data sets.
2. *Hadoop:* an open source implementation of MapReduce that is used for large–scale data processing. It is designed to interact with data stored in a distributed file system.
3. *Android OS:* a well–known Google operating system.

To examine the advantages and disadvantages of Hyrax, a multimedia search and share application called Hyraxtupe was developed. Hyraxtupe allows users to browse videos and pictures that are stored on a network of phones and to search by time, location and quality. A web application is used as a client interface.

VM–Based Cloudlets

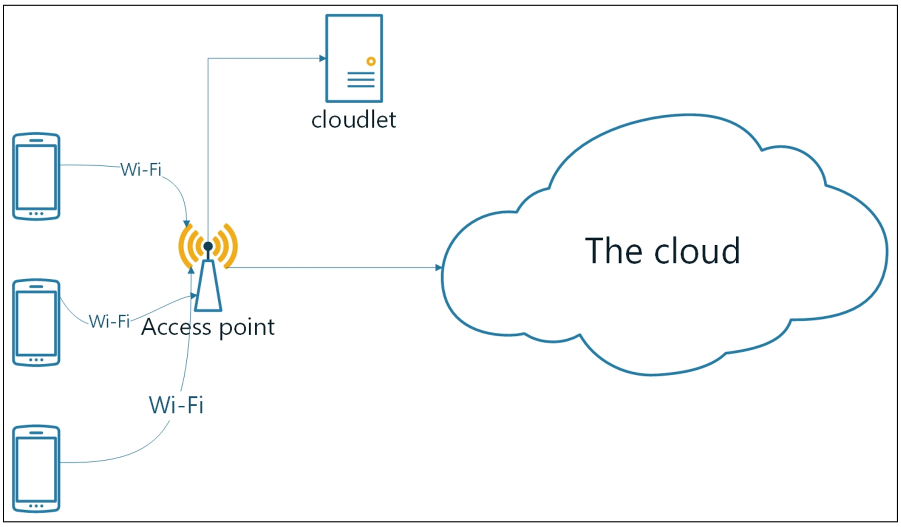


Figure 3.2: Cloudlet Approach

One example of solutions for mobile clouds is cloudlets, as proposed by Satyanarayanan et al. [36] and defined as “trusted, resource rich computers in the near vicinity of the mobile user (e.g. near or co–located with the wireless access point)” (p.1) [39].

The idea behind this brings a self–managing (typically small) cloud infrastructure close to mobile devices to allow them to access cloud services through a one–hop Wi–Fi link, such as near a broadband access point at home or in a café (see Figure 3.2). Here, cloudlets are not intended to take the role of the cloud, but to deal with delays that may result from accessing cloud services.

Technically, the hardware of each cloudlet runs virtualization software that is responsible for running/managing VMs, which are created and owned by users (usually a small number of users) and allow the creation of a VM that rapidly initiates customized service software on a nearby cloudlet to be accessed over a wireless link. These VMs are kept in the cloudlet machine, in which a small file (around 100MB) that contains the users’ profiles and settings is stored on the mobile device.

An overlay–VM technique is used here to manage this operation. As a result, when the user connects to the cloudlet again (by being near the cloudlet location, such as at home or in the office), a full VM is created to allow users to access the cloud service. The cloudlet can perform jobs to serve the user, whereby heavy and intensive tasks (particularly those that do not require real–time communications) will be forwarded to the remote cloud.

As mentioned earlier, cloudlets are aimed mainly at deal with the latency that may occur when users access cloud services by deploying a cloudlet one hop away from mobile devices. As a result, real–time applications/tasks are executed/run on the cloudlets more quickly than executing/running these applications in the remote cloud data centre. Furthermore, users are allowed to customize their own applications and profiles that are stored in the VM–overlay, as mentioned previously. However, the most noticeable limitations of this approach are as follows.

Availability

Cloudlets might not be available near the user’s location. In one paper [40], cloudlets are provided as mobile devices to deal with availability issues. However, this scheme is less powerful than the regular cloudlets (such as using a fixed PC) because it relies on mobile device resources.

VM–related limitations

Delays may arise with the creation of a full VM on the cloudlet.

Network–related limitations

When multiple users are using the same network to access the cloudlet, an issue such as a bottleneck can occur. Furthermore, the throughout system will be effected when a cloudlet intends to handle a large number of requests at the same time.

However, in one paper [41], a protocol is designed and discussed that aims to increase the system throughout by effectively and efficiently handling requests coming from mobile devices based on the current workload of each resource in the cloudlet.

Mobility–related limitations

These include disconnections and saving the progress of sessions.

With regard to healthcare, the authors of one paper [42] proposed a system based on the cloudlet approach with the aim of enhancing the end–to–end cost of the large–scale body area network (BAN) [43]. In other words, the system is proposed in order for collected/monitored BAN data to be available to the end users (or server) in a reliable manner. The paper captures the power consumption and delay resulting from transmitting data from BANs to cloudlets and to the regular cloud. It was found that power consumption and latency are improved (compared with transmitting collected data to the remote cloud) when the number of cloudlets is increased because this increases the Wi–Fi coverage area.

To summarize, using a cloudlet approach can offer real–time interactive responses through low–latency, high–bandwidth, one–hop wireless access. The cloudlet infrastructure is similar to Wi–Fi access, in that the main challenge is managing the cloudlet framework.

3.3 Mobile Health

Introducing mobiles into the healthcare sector has resulted in a new term: 'm–health'. In general, m–health [44] is defined as the use of mobile devices, such as smartphones and tablet, for delivering health services and information.

However, a more comprehensive definition of m–health is: *"mobile computing, medical sensor, and communications technologies for health care"* (p. 405) [45]. Table 3.2 provides other definitions of m–health from various sources.

Other authors [46] believe that introducing mobiles into healthcare can:

- make healthcare more accessible, affordable and available.
- improve the decision–making and production processes.
- facilitate information access, enhance workflow and promote evidence–based practice.

As stated in the above paper, the unique characteristics of mobile devices, such as location–based services, ease of accessibility and usage, low cost and the ability to connect to the Internet, have made these types of device attractive to the health field. Furthermore, the paper discusses in detail a number of factors that play a crucial role in m–health solutions, which include, but are not limited to, the following:

- Being 'consumer centred', which means focusing on health consumers' needs.
- A high level of care has to be guaranteed.
- Allowing the involvement of all parties, including patients, providers (medical staff, such as doctors and nurses), and healthcare organizations, such as hospitals.

With regard to research outcomes in m–health, one paper [53] comments that these are mostly *"a multitude of small pilot projects, particularly in low– and middle–income countries"* (p. 393). The paper presents seven key recommendations that have the goal of avoiding programme duplication and reducing resource wastage, as follows:

1. M–health needs to develop an evidence base.
2. M–health systems should be interoperable with existing e–health initiatives.
3. M–health should adopt and implement the same standards already present in e–health.
4. M–health should take a participatory approach.
5. M–health should promote equity in health.

6. M–health programmes need to move towards sustainability.
7. M–health needs to focus on health, not on technology.

Table 3.2: Definitions of M–health from Various Sources

Definition	Source
Emerging mobile communications and network technologies for healthcare.	Istepanian and Lacal [47]
A subset of eHealth using mobile devices to deliver health services to the patients.	Mechael [48]
Refers to the application of embedded wireless devices to track health related parameters.	Bardram et al. [49]
A personalized and interactive service which provides ubiquitous and universal access to medical advice and information to any users at any time over a mobile platform.	Akter et al. [50]
Medical and public health practice supported by mobile devices, such as mobile phones, patient monitoring devices, personal digital assistants (PDAs), and other wireless devices.	World Health Organization [51]
Using communication such as PDAs and mobile phones for health services and information.	UN Foundation [52]

3.3.1 M–health applications

M–health applications and solutions are being provided and tested in a variety of health domains, such as diabetes, asthma and smoking cessation. *"From patient monitoring and diagnostics to more efficient medical education and communication, smartphones serve a vital role in the practice of medicine today"* (p. 2) [54]. However, it is still unknown whether these solutions make any improvement or reduce disease risk [55].

Interestingly, these solutions can be deployed/run in special medial mobile devices or today's smartphones, such as iPhone– or Android–based mobile devices. With respect to smart device applications, the British Red Cross [56], for example, has an official app that offers step–by–step advice using pictures and videos, tests that assess knowledge of first aid, and provides emergency numbers.

Another example is the HelpMe app [57], which plays a loud alert in emergency situations to draw nearby people's attention, as well as sending an auto–SMS (Short Message Service) help message to a pre–defined friend or family member.

One of the practices of m–health is the use of wearable medical devices to check aspects of people's health, such as detecting medical/health abnormalities in elderly people [58]. Using wearable devices is part of the out–of–hospital schemes that have been given a high level of attention in the literature. In one study [59], the authors claim that wearable technology is the *"next big thing"* in tech industries, while others [60] provide a survey of wireless body area networks and highlight some applications with a special interest in patient monitoring.

There is also a wide range of wearable devices and body sensors on the market that allow users to track their activity, monitoring aspects such as walking, running, sleeping, and heartbeat rate [61]. With regard to emergencies, wearable devices can be used to identify a personal emergency, such as a heart rate that is acting abnormally [62].

3.3.2 Existing systems and projects

One Malaysian project [63] consists of healthcare information delivery using SMS and Multimedia Messaging Service (MMS) messages to people who are in emergency situations. The system offers a number of features: locating the nearest healthcare centre by SMS, searching for doctors by SMS, and broadcasting SMS messages containing helpful information. Interestingly, in the case of an emergency, first aid staff have the ability to communicate with hospitals using MMS messages to obtain recommendations about a patient's medical state (these messages might contain text, photographs or videos).

One author [64] outlines difficulties faced by pilgrims and authorities during the Al–Hajj (pilgrimage) seasons, such as the identification of pilgrims (who might be lost, dead or injured), emergency situations, guiding lost pilgrims to their camps, and controlling crowds. This paper proposed a system that works as follows: it assigns an ID to every pilgrim, then confirms their locations and sends these to a web server (if the connection is lost, location information is stored temporarily on the phone until the signal returns). When the locations are received, the system stores them in a database with the pilgrim's ID. Retrieving locations for a particular pilgrim can be done by accessing the system's website and then typing the pilgrim's ID.

Another study [65] presents an advanced system that provides help in the case of an emergency that serves users from the place of the event to the hospital, called a Comprehensive Emergency System (CES). The system involves a number of health parties, such as first aid departments, emergency departments, and medical record centres.

The paper describes the framework of the system as follows:

- The system is triggered when a request is received in the Main Centre System (MCS) via a simple mobile application where location is detected automatically.
- The MCS sends a request to the ambulance closest to the emergency location. To reduce delay, the ambulance is given a limited time in which to accept the request. If the request is not accepted, it is sent to another ambulance; if the job is accepted, the MCS provides directions to the location of the emergency.
- When the ambulance arrives, staff can use the patient's fingerprints to retrieve medical records. They then search for appropriate hospitals, depending on the patient's condition, distance, availability and the speciality of the hospital. When a hospital is chosen, information about the patient is sent. Ambulance staff can also consult doctors on the move, if needed.

Hospitals prepare what they plan to do for the incoming patient based on the information that is received. Some authors [66] have discussed the design and integration of social networking applications with cloud computing and mobile technologies to provide a system that allows users to communicate during emergencies or disasters. A Facebook app is proposed in the paper to design a Personal Emergency Preparedness Plan (PEPP), which aims to allow users to collect geographical information, pictures and videos, along with any other data that may be useful in these kinds of situations. As shown in the paper, non–medical professionals (such as friends or family members) can be connected to the user to provide help in emergency cases.

Elsewhere, a web service called CliniCloud [67] presents a medical kit that allows people to track their state of health using smartphones and store their data in the cloud. Users then have the ability to contact a service called Doctor on Demand [68]. This service allows users on the move to start a video call with a doctor. This service is only available in the US, however, and the system requires additional medical diagnostic equipment, which results in higher costs.

An Android–based emergency alarm and healthcare management system has also been presented [69]. The system aims to provide two main functions to users: a trigger alarm to friends or hospitals, and a life reminder feature to remind the user to take medicine on time. The authors discuss the benefits of integrating health systems on mobile phones, these advantages being portability, open operating systems, the use of the Global Positioning System (GPS) and the ability to detect events using a gravity sensor. The proposed system was tested on Android mobile phones and laboratory–based servers.

However, this system only allows interactions with hospital–based medical staff, whereas the system proposed in this thesis benefits from the facility to interact with medical staff who are not on a hospital site, such as those who are in a vehicle or in a shopping centre. The system is also part of a hospital, whereas the idea proposed in this thesis is to implement a stand–alone system that allows interactions with all the parties who could be involved in an emergency.

A mobility management system for mobile cloud computing, M^2C^2, has also been proposed [70]. It aims to create a local cloud(s) in emergency locations to help emergency providers reach cloud services. This is similar to the idea of cloudlets (as shown previously in section 2.2.3.5). Operations such as executing mobile–based applications, collecting sensor data, and submitting wearable device readings happen locally, via Wi–Fi or 3G links, in what is called an Emergency Response Vehicle (ERV). This kind of vehicle offers functionality, such as low latency data processing, storage and access. The local cloud then communicates with the public cloud on behalf of the users. According to the paper, more than one ERV can operate from the same emergency location and, therefore, a mechanism for selecting the most appropriate ERV in terms of networking and delay is tested and discussed by the authors. The authors claim that using their approach will benefit emergency services by reducing delays. Channelling activity through these emergency locations can be more than twice as fast as connecting directly to the public cloud.

The M^2C^2 system differs from the approach proposed in this thesis in that it is mostly provided to professionals, such as ambulance crews or firefighters, whereas the proposed system aims to provide health services to normal users (e.g., the public), as well as support to medical staff. The proposed system supports as wide a range of users as possible to deliver the right medical treatment to people who are experiencing an emergency.

A cloud–based system that can be installed in an ambulance to provide health support on the move has also been suggested [71]. The proposed system offers video conversations over a cellular network between a first aid crew at the scene and doctors who are located in a hospital. Doctors can view/update patients' medical records, retrieve directions for the shortest route to the nearest hospital, and prepare the emergency department for what the patient needs based on information from the first aid crew. The authors claim that using this system will save lives by providing the right treatment at the right time and by taking into account mobility and road–traffic issues.

However, using this system requires ambulances to have special equipment. Furthermore, the selected specialists cannot provide any further help until the ambulance has arrived at the hospital if there are connection problems and a call cannot be made. Communication can also only be made with doctors who are present in the hospital.

In another paper [72], the design of an online social mutual help system using the mobile cloud is discussed. The main aim of this paper is to improve the feasibility of delivering help on a multi–tenant framework. The authors propose a helper recommendation algorithm that processes help requests, providing the best possible support based on user affinity and using data mining and machine learning to capture users' behaviour. Once a recommendation is found, it will be pushed to or pulled by the user, depending on how active the user is.

Interestingly, the proposed framework provides a RESTful web service application programming interface (API) for better integration with third–party systems. The framework was built and evaluated on a physical machine and showed improvement in the success rate of the mutual help process when the aforementioned algorithm was introduced.

3.3.3 Discussion

After reviewing m–health related research, two main question are asked here:

1. What is the relationship between m–health and the mobile cloud?
2. How can the mobile cloud help m–health?

From one side, when healthcare services are provided to mobile users, which is the main purpose of m–health, great benefits can be achieved, such as easy and quick access to medical resources [73].

On the other side, mobile devices have limited resources, such as battery lifetime, and suffer from a lack of reliability [74]. Mobile cloud computing can help deal with some of these limitations, by, for example, shifting heavy computational processes to the cloud [75]. Mobile devices can also store and access data in the cloud to improve reliability and storage capacity [76]. Therefore, MCC has been found to be a suitable solution to known m–health shortcomings, such as small storage capacity and medical human error [77]. Ultimately, the benefits of mobile cloud solutions in m–health can lead to improved support and care delivery, enhanced availability and reliability, and allow a wider dissemination of data and other useful information [78]. From here, the second question can be considered. In other words, once m–health solutions are run on mobile devices, any improvement in the mobile devices themselves will lead to an improvement in m–health as well.

In the literature, Chang et al. identify high–level requirements for providing healthcare services using cloud–computing technologies [79]. The paper reviews and analyses some of the existing IT–enabled healthcare ecosystems. In addition, the authors give their views on the imperatives for cloud computing research in supporting future IT needs for healthcare.

Other authors [80] have reviewed the significance of and opportunities for implementing cloud computing in the healthcare field. The authors discuss the benefits of introducing the cloud paradigm to healthcare areas, such as on–demand self–service, ubiquitous network access, resource pooling, rapid elasticity, and a pay–per–use model. The paper also provides several protection methods that can be utilized to deal with security and privacy issues.

To summarize, m–health can benefit from the cloud paradigm in many respects, such as on–demand self–service, ubiquitous network access, resource pooling, rapid elasticity, a pay-per-use model, and virtualization [81]. Recently, cloud computing has been playing an important role in enhancing the development of healthcare information systems [82]. Furthermore, cloud–based health systems can provide easier access to medical data to reduce time and cost, as well as allow collaboration on and sharing of medical resources, information, and files.

M–health has a unique set of requirements, as well as limitations such as lack of security and privacy, due to concerns regarding dealing with medical data [50]. However, it is clear that m–health also shares the same benefits and limitations of mobile devices. Therefore, any solution that helps mobile devices to deal with their limitations will have a significant impact on m–health.

3.4 Related Technologies

Two main technologies are reviewed in this section: mobile ad–hoc networks (MANET) [83] and social media applications [84]. The former technology is used as the infrastructure for the proposed system, which is presented in detail in chapter 4. Reviewing the latter technology will help create a mobile cloud service that makes the best use of a social media application in emergencies (as in chapter 6). This mobile service is one of the proposed system services aimed at enhancing system availability, as well as allowing the fast dissemination of relevant data and information in the case of an emergency.

3.4.1 Mobile ad–hoc networks (MANET)

As shown previously, a MANET is considered here for the purpose of creating a robust infrastructure that allows users who do not have a direct connection to the cloud to reach cloud services through neighbouring links.

MANET management is a complex task, generally carried out by a middleware system running on mobile devices and, therefore, consuming their resources [85] [86]. Specifically, MANET events, such as when a new device joins, the sudden departure of a device, a network split or the merging of networks, are difficult to handle.

Indeed, the process of IP allocation in order to avoid duplicates or waste consumes much of a mobile's resources, particularly energy, as it generally involves all the MANET members. A network split due to the mobility of devices can lead to session termination between peers that now belong to departing sub–networks.

One paper [87] presents a number of network models that are based on the idea of mobile multi–hop ad–hoc networking. The paper refers to them as MANET–born models because they are built on the lessons learned from MANET research. The authors believe that there is a lack of attention to application and middleware in the MANET field that has led to a reduction in MANET spread.

In a summary of the paper, the authors forecast that the presented models will be mixed together and/or combined with infrastructure–based networks to create what they call *"the multi–paradigm era"* (p. 95).

3.4.1.1 Published similar work

In the literature, there are some solutions for dealing with the complexity of managing a MANET. For instance, Zhang et al. propose a secure geo–social networking system that uses a Wi–Fi–based multi–hop MANET [88]. The main goal of the paper is to present a system that allows users who are located in the same place to communicate with each other using a Wi–Fi network. They can share content (e.g., photographs and videos), chat and form relationships. The paper provides a multi–hop MANET platform for mobile devices called MoNet, in which two applications are designed: Wi–Face and Wi–Market. Cloud services are mentioned in the paper but not for MANET management, and no implementation is presented.

In another project [89], a framework called the Mobile–Cloud is introduced, which aims to address trust management, secure routing, and risk management issues in MANETs. The framework transforms traditional MANETs into a new service–oriented communication architecture, in which each member becomes a service node that can be used as a service provider or service broker. In general, each member of such a MANET has to provide sensing data about the device itself (such as battery state and processor type), and neighbouring mobile nodes (such as address, link quality and number of hops).

As a result, the cloud will have an overview of each MANET operation that can be used to provide information to mobile devices for decision making (routing a request, for example). The framework is intended to handle all processing and data collection in a centralized way to turn all communication from one–to–many to one–to–one. Clearly, the paper requires each mobile device to have an Internet connection able to connect to instances in the cloud. That is not possible for most MANETs.

3.4.1.2 Wi–Fi Direct

Recently, Android was provided with a new service called Wi–Fi Direct [90], which allows users to create one–to–one or one–to–many communication without the need to set up an infrastructure. Wi–Fi Direct uses a peer–to–peer (P2P) framework, in which one device acts as group owner and any other device can be a peer to that device. It relies on a technique called a Software enabled Access Point (SoftAP), which makes a wireless antenna work as both access point and client [91].

Wi–Fi Direct is offered in most new devices, such as mobile phones, cameras, printers, PCs, and gaming devices. According to one project [92], there were more than 1,400 different devices certified for Wi–Fi Direct as of October 2012.

Android developers have provided a Wi–Fi Direct feature in Wi–Fi–enabled devices: Android 4.0 (API level 14) or later. Programmatically, developers can use P2P and a Wi–Fi manager library to build all necessary pieces to use this feature. The benefits of this are enhanced mobility – Wi–Fi Direct devices can connect anytime, anywhere to each other directly, since a Wi–Fi router or AP is not required – and ease of use, as there is no need for pre–configuration. All that users need do is detect nearby devices, then send a connection request and secure the connection.

However, only the most recent devices support Wi–Fi Direct. The group owner can be a bottleneck, particularly if the number of peers is high. Using this feature requires interaction among users: each user has to choose a device with which to connect and that chosen device has to accept the request before establishing communication.

3.4.1.3 Serval mesh

An Australian group is working on a project called Serval Mesh [93], which aims to enable devices to perform self–organizing P2P networks with gateways to the Internet when available. Users can either send text messages to a particular peer or to all peers (broadcasting) in the same area, and/or calls can be made to any peers using the Voice over IP (VoIP) [94] technique.

The purpose of this project is to provide widespread communication in areas where the infrastructure is damaged, such as in the case of a natural disaster. A custom routing protocol that behaves in a similar fashion to the optimized link state routing (OLSR) protocol has also been used [95]. An Android app is available on the Android Market [96] and consists of two components: the Batphone and Serval DNA (distributed numbering architecture). The Batphone is the user interface; Serval DNA is the core networking, encryption and file–sharing component.

The benefits of this project are that it is open source, free to use, carrier independent and all communication is done using Wi–Fi links. However, devices need to be rooted for full features. It works without rooting if peers are connected to the same network. The entire software is still in the experimental stage.

3.4.1.4 MANET Manager app

The MANET Manager app [97], which is part of a larger project called Smart Phone Ad–hoc Networking (SPAN) [98], is composed of several components. These are the MANET service, the modular MANET routing protocol framework, and a layer of managers for security, session management, and reliable transmission. The app provides a choice of several reactive and proactive routing protocols through the modular routing protocol framework [99]. The app components work together to provide a Java observer interface that other apps can implement. The apps can then interact with the network that the service creates.

3.4.1.5 Configuring a MANET manually

By modifying the Wi–Fi configuration file manually, some developers have managed to enable the kernel to get ad–hoc network working: the Wi–Fi chip is changed from its normal mode to an ad–hoc mode.

In practice, users have to add ad–hoc network details (service set ID [SSID], priority and ID) in a wpa_supplicant.conf file, and then set the ap_scan equal to 2 to perform ad–hoc networking.

The benefit of this approach is its direct and easy way of allowing ad–hoc networking. However, mobile devices will work in either an ad–hoc mode or a normal mode; the two modes cannot run at the same time. Moreover, this method does not work on all Android devices, as it requires superuser access.

3.4.1.6 MANET sessions

In a traditional MANET, if two nodes are engaged in a session and one of them departs suddenly, their communication is aborted. The session is no longer active, work is lost and, consequently, the energy of the batteries has been wasted. Therefore, solutions and techniques for dealing with MANET active session breaks have to be reviewed, as this will lead to better management of the MANET itself, as well as saving mobile resources. One possible solution is to introduce checkpoint techniques.

In one paper [100], different approaches to setting up checkpoints in MANET are presented, such as checkpointing and a rollback recovery scheme for cluster–based multi–channel ad–hoc wireless networks [101] and checkpointing using a flooding method [102]. The paper takes into account MANET properties such as mobility and device limitations [103]. The paper defines a checkpoint as *"a fault–tolerant technique that is a designated place in a programme at which normal processing is interrupted specifically to preserve the status of information necessary to allow resumption of processing at a later time"* (p. 31). Therefore, if a failure occurs, the computation can be restarted from the most recent checkpoint, instead of repeating the whole task from the beginning.

Nicolae and Franck [104] provide a checkpoint–restart solution to deal with the dynamic and long runtime of IaaS clouds. The solution is intended to reduce storage space and computational overheads. The idea concerns attaching VM disc images to the checkpoint protocol at the user level to accelerate resumption of the application process. A number of similar projects that use fault tolerance are also discussed by Elnozahy et al. [105]. The system presented, known as BlobCR, periodically saves the global status of an application to maintain stable storage and then restarts the application from an intermediate point if interruption occurs. The authors claim that checkpoints of disc–only snapshots are more compact and faster than those for the whole status of a VM instance.

A system involving 'follow me sessions' is proposed by Handorean et al. [106], whereby a session follows the client and can be continued with another peer if disconnection occurs. As a result, the client is connected to a service instead of to a server. The paper assumes that the new provider node offers the same functionality. The checkpoint technique is used to record the execution state for resuming a session from an intermediary point, preventing processor time being wasted. Two types of migration are provided: weak migration, whereby a session can be run from the beginning (not resumed), and strong migration, in which a session is stopped, transferred, and resumed. The latter is more powerful but is also more expensive to perform in terms of network resources and delays.

Overall, the idea of creating checkpoints in the MANET area is not new. While many of the research papers deal with recovery from failures occurring to a whole system [107] [108] [109], other papers examine the overheads resulting from creating checkpoints that may put more pressure on mobile device resources (e.g., storage capacity and battery usage) [110] [111] [112].

However, the project proposed in this research study is interested in dealing with any failures that may occur during a session that is active between two nodes that are members of the same MANET. Thus, when the session becomes active again after a break, it can resume from the point that was reached before the break. This procedure can be carried out with the help of cloud services to deal with the issue of the unstable storage capacity of mobile devices, as well as avoiding additional power consumption. Another benefit of introducing cloud computing is that sessions can be resumed using another mobile device because all the information has been linked to the user rather than a device.

3.4.2 Social media in emergencies

As stated previously, social media applications need to be reviewed in order to understand how this technology can improve healthcare delivery, particularly in the case of emergencies. Social media is fast becoming one of the most popular communication/news sharing applications used by the public.

In practice, some emergency/disaster–related organizations have set up accounts on Facebook and a number of hospitals have Twitter accounts. Groups of experts in fields such as medicine and technology have also been established on LinkedIn [113]. It is clear that usage is increasing year by year, with Twitter, in particular, providing a communication platform during emergency events [114].

Twitter has attractive features, including a free service, online access, it is mobile friendly, messages stay queued until delivered, it offers better search and classification abilities using hashtags in the body of tweets, and supports tracking features. Twitter also makes the best use of bandwidth, by sending short (140—characters) messages, as well as allowing the wide and fast distribution of information.

With regard to emergency management, one paper [115] provides a list of criteria that any 21^{st} century emergency system should have, such as low cost, power efficiency, ease of use, being mobile friendly, the ability to receive, generate, provide, and direct useful and critical information from a variety of sources, and GPS ability. The authors believe that Twitter matches/meets the criteria for an ideal emergency communication system that aims to provide help to the public in the case of an emergency. As proof of concept, the paper examines a number of large–scale events, such as the California fires and Sichuan earthquake, highlighting the use of Twitter in these situations.

Issues that can occur during the course of using social media are also discussed in the paper, such as spam, lack of privacy, misuse of tracking features, and impersonation. According to the paper, many of these issues are general symptoms of online networks, in which lack of privacy and trust is one of several features for which social media is known.

3.4.2.1 Possible usage of social media

In general, there are two main directions for using social media in emergencies:

1. **Passive**, such as disseminating useful information to the public or receiving feedback from people who recently received help.
2. **Active (systematic)**, which includes building live communications and issuing warnings to the public, receiving victim requests in the event of risk and emergencies, monitoring social media activities, and using the streams and multimedia exchanged in social media (for example using uploaded images) for estimating what is required to respond to an event or assessing the level of the damage caused.

However, not many medical and emergency centres are yet able to use social media actively. It is believed that less than 10% of the total usage of social media is active, while the rest is passive [116].

3.4.2.2 Benefits and challenges of social media integration

Using social media can benefit any system of emergency and risk management in any of four phases: (1) preparation — coordinating activities among stakeholders; (2) response — assigning response teams to areas in need; (3) mitigation — sharing current status and locations; and (4) recovery — providing a communication platform, particularly when regular networks are limited or down [114]. Furthermore, social media could create a platform on which all the entities engage almost in real time, including the public and the risk and emergency management centres or teams. Another benefit of social media noted in the literature is its efficiency: *"Social Media can help efficient communication to a large audience and well–targeted groups of people, with fewer resources and efforts than other communication media"* (p. 115) [117].

However, there are a number of challenges to using social media in risk management, such as the real–time analysis and monitoring of social media streams. In one paper [118], the authors discuss processing and managing social media data concerning emergency events. The paper defines challenges according to two main high–level categories: (1) scalability: recording millions of messages and storing multimedia objects (e.g., images and videos); and (2) content: social media and microblogging messages particularly are brief and informal, which may lead to poor results.

Processing multi–language data could also be a concern. The paper presents some approaches, including classification methods, to make data more researchable and meaningful, and a sub–documenting method to assist in making predictions or decisions from the data collected.

Another critical issue in using social media in emergency and risk management is trust, which can be compromised in the case of false alarms or the employment of sarcasm. According to one paper [119], although sarcasm is a well–studied phenomenon in some sciences, such as linguistics, it is still difficult to assess in the text–mining literature because of its complexity.

The paper presents a mechanism for detecting sarcasm in Twitter messages. The authors compare the performance of human–based classification and an automated form and find that an automated technique could be as good as a human agent. However, according to the paper, both approaches are still difficult and neither performed very well.

3.4.2.3 Existing systems and projects

A research team working with the Crisis Coordination Centre (CCC) in Australia to improve emergency management and crisis coordination have proposed a system that collects and analyses tweets regarding an emergency event [120]. The system aims to enhance emergency awareness by enabling responses to emergency warnings, near–real–time notification of emergencies, and first–hand reports of the impact of an event.

The authors believe that if information collected from social media regarding an event is extracted and analysed properly and rapidly, this could improve the level of situation awareness. The paper mentions the amount of Twitter data (around 30 million tweets) captured during the earthquakes in Christchurch, New Zealand, in September 2010 and February 2011.

The authors found that, when earthquakes or aftershocks occurred, people actively broadcast information, sympathy, and other messages on Twitter.

In order to deal with real–time and high–volume text streams, the paper adapts and optimizes various data–mining techniques, including burst detection, text classification, online clustering, and geotagging, for the purpose of the early detection of sudden emergency events and exploring/monitoring the events identified.

Another example of using social media in managing emergency events is dealing with public panic that might occur [121]. The authors discuss the use of Twitter by the Centre for Disease Control and Prevention (CDC) in the USA when the first patient in the country was diagnosed with the Ebola virus in September 2014. The CDC provided live Twitter chat to address uncertainty and dispel misunderstandings regarding the virus. The authors collected around 2,155 tweets containing the hashtag "#CDCchat". The data were then processed using SAS Text Miner 12.1 software [122] because the program provides the ability to parse and extract information from text, reliably filter and store that information, and assemble tweets into related topics for inspection and to gain insights from the unstructured data. Based on the results from the analysed data, the paper finds that there were eight mutually exclusive topics, among which the greatest concerns were regarding the virus itself, its lifespan and symptoms, and how it is transmitted.

In another paper [123], a system that aims to enhance communication channels in the case of an emergency, as well as assisting victims, is proposed. In this research, Twitter was used as a form of social media application. According to the paper, quantitative analysis can be used to examine the status of an emergency event. For example, if the number of tweets increases during a certain time, this indicates that a change has occurred. The authors also find that there is a relationship between the development of an event and the nature of the data content, which they categorize into three types: precautions, warnings and reports. In other words, the greater the severity of an event, the more intense the data content will be. The paper uses a flood that took place in Jeddah in Saudi Arabia in 2011 as a case study. A unique hashtag was defined and 30 Twitter users were involved. A collaboratively developed map was provided for use by the public highlighting the flooded area. This map was created by a number of volunteers and utilized data collected from Twitter.

In this research work [124], a Knowledge as a Service (KaaS) framework is proposed for the purpose of providing better disaster management by taking advantage of cloud computing resources. The main goal of this research is to facilitate improved and informed disaster decision-making and so reduce the impact of disasters on lives and property. The proposed framework is intended to gather, store, and share useful information regarding a disaster to be used in better management of such events.

3.4.2.4 Discussion

As mentioned previously, social media apps allow citizens to provide valuable on–the–ground information that would benefit an emergency management system by providing a clearer picture of an event and the level of damage [125]. Social media can also help rescuers to find victims much more quickly and effectively, either when those people send a help request via social media or when a concern is raised by a family if a loved one is missing in an event such as a flood or a storm [126].

Social media offers the quick and wide dissemination of information. However, this could be one of the most noticeable disadvantages of using social media, as rumours can also be spread very quickly and widely [127].

There are also a number of challenges that make it difficult for social media to be (a) trusted and (b) suitable for emergency and risk management. For instance, social media apps are designed for a purpose, that is, to allow users to communicate, whether for the exchange of information or as a form of entertainment, by telling friends about what they do and do not like.

To summarize, using this sort of app in emergency and risk management would require some modifications. A key modification is enabling the location service on a mobile because it would be difficult to reach someone who is seeking help without knowing his/her location.

3.5 Chapter Summary

This chapter provided a review of four main research topics: the mobile cloud, m–health, mobile ad–hoc networks and social media applications. The first research topic was reviewed to provide a better understanding of the mobile cloud, including possible implementation, the benefits it can offer, and challenges that might occur. Similarly, the second topic, m–health, was reviewed to provide an overview of m–health applications, advantages and limitations.

As this work is intended to introduce mobile cloud solutions into m–health, a number of papers and existing systems were presented and discussed. It is clear there is a large degree of interest in the integration of m–health and the mobile cloud [128].

The third topic was reviewed for the ultimate purpose of this research work, which is to build an infrastructure or middleware for a new system that can help mobile device users to reach available services reliably that are hosted by the cloud. In other words, it is clear that using a MANET can lead to an increase in system availability and reliability by allowing a member of a MANET to access cloud services through a neighbouring link. On the other side, the cloud can reach a higher number of users in the same way. Furthermore, the dissemination of data and information can be extended using a MANET and the cloud idea, for example if the cloud wants to broadcast useful information regarding an emergency event.

Messages can be sent from the cloud to all users who have a link to the cloud but, more interestingly, users who do not have a direct link to the cloud but can connect to a MANET can receive this broadcast message over one of the neighbouring links. Finally, useful data can be shared locally between each MANET member, such as an interactive map showing both the affected and the safe areas of a city that has, for example, recently been flooded. If the map is large and there is a possibility of a break in the link, a checkpoint technique of saving the progress of a session, such as downloading a map from a peer, can be saved to be resumed when both users intend to accomplish the session again. In the latter case, a number of checkpoint techniques were reviewed in this chapter for the purpose of designing a feasible solution to deal with sessions breaking between MANET members.

Moving to the fourth and last research topic that was reviewed in this chapter, which was social media applications and their possible usage in emergencies, a review and the background information provided in the chapter showed that social media apps have attracted attention for their potential to provide some form of support and healthcare in the case of an emergency. This chapter examined possible solutions that could use social media to provide health support to people in emergencies, taking into account the benefits and challenges that may be gained or faced because of this type of integration.

This led to the idea of creating a service that was hosted in the cloud and made accessible to mobile users which would make the best use of social media applications (e.g., Twitter). For example, by broadcasting useful information to the public through a social media network or collating on–the–ground information regarding an event such as a flood that has occurred in certain areas of a city.

In summary, mobile devices allow people to have access to the Internet on the move. On the other side, cloud computing allows the deployment of services that are accessible over the Internet. Therefore, introducing the mobile cloud to make improvements in the health sector, particularly m–health, would benefit all kinds of users (e.g., patients and physicians).

The following chapter, chapter 4, discusses the initial design of the research system based on the topics reviewed and considered in this chapter.

Chapter 4

Introducing Mobile Cloud Technology into M-Health to Deliver Better Care/Support in Case of Emergencies

4.1 Introduction

The literature review has demonstrated that many factors contribute to making m–health difficult to implement. This ultimately has a detrimental impact on the ability to improve it and accomplish its goal, which is to deliver low–cost healthcare to anyone, anywhere, anytime. Most of these factors are related to mobile device limitations, such as limited storage and computational resources, as well as lack of security and privacy.

Therefore, the research in this thesis focuses on the design and implementation of mobile cloud healthcare middleware that delivers at the point of care, in order to provide a robust and reliable mobile health system that can offer fast and low–cost care to the public when someone is experiencing a health emergency.

The following sections discuss the initial design and architecture of the proposed system, including the performance requirements and the type of users who might interact with the system.

4.2 System Model

This section discusses the design overview and the system's main components and presents the type of users who might interact with the system.

The system can be divided into five main components: (1) the cloud, where all the services are hosted; (2) the MANET–cloud middleware that is proposed by this research, with the aim of enhancing the m–health system in emergencies by providing reliable and robust communication between users who seek services and users/centres who can deliver it; (3) a wireless communication medium, such as Wi–Fi or a cellular network, including SMS messaging; (4) users who communicate with the cloud to seek or provide healthcare services, and include patients and professionals; (5) a supportive service, such as a social media network, that is not part of the system but is connected to it in order to provide a unique feature, allowing, for example, the system to send and receive SMS messages.

Figure 4.1 displays these components graphically and the following subsections present each component in turn.

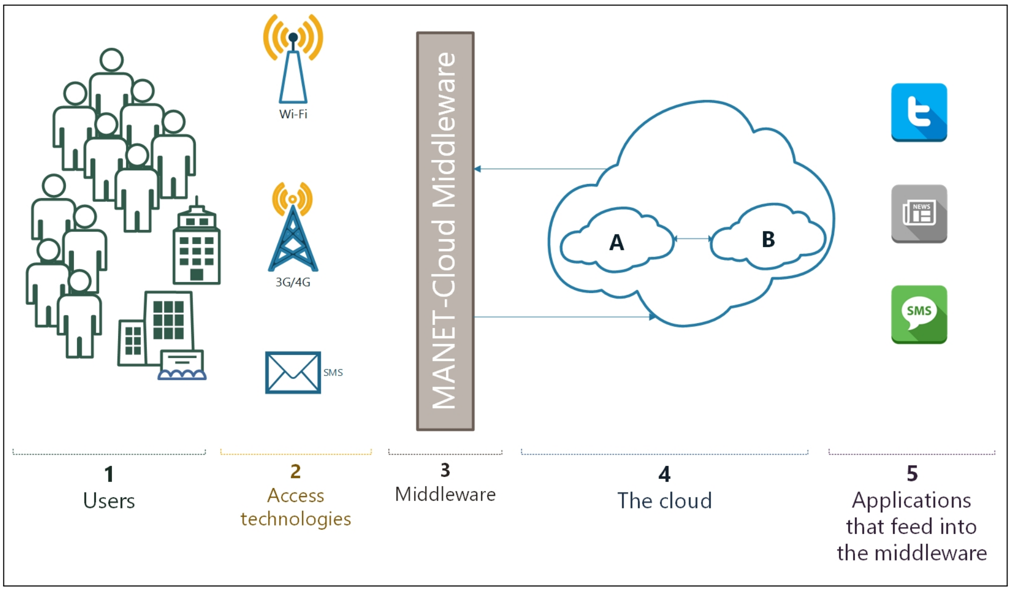


Figure 4.1: System Overview Model Including all Major Elements

4.2.1 The cloud

As stated previously, mobile cloud computing can help mobile devices to deal with certain limitations, such as restricted storage and computational capacity, by shifting heavy tasks so that they can instead be processed in the cloud.

However, the cloud in the proposed system might be deployed/managed by a main hospital in a city or centralized by the Ministry of Health, as in Saudi Arabia.

The main purpose of this cloud is to host all the services and make them accessible to mobile devices anytime, anywhere. For instance, the cloud might hold directions to all the hospitals in a particular country, so that when a user needs to retrieve the nearest medical centre to his/her location, a request can be made to the cloud to retrieve directions.

Another example could be storing the EMRs of patients, as well as managing these records so that only authorized personnel, such as a doctor who is currently dealing with the user's case, can access them.

Usually, the cloud is made accessible publicly — as in a public cloud. However, since dealing with sensitive medical data can raise security and privacy concerns, the cloud can be split into two: a private cloud to hold sensitive data, and a public one for general purposes, such as sending a help request to retrieve useful information in emergency situations.

4.2.2 MANET–cloud model

This section presents the design of model that is aimed at improving system connectivity and robustness by allowing users to create/join peer–to–peer and self–organizing networks, such as a MANET. Furthermore, these networks or MANETs are managed by the cloud to overcome some of the issues relating to implementing a MANET–type network, such as IP allocation and split/merge operations.

The model will help users to reach the cloud, not only directly through the Internet, but also in seeking help from the cloud through a neighbouring link. On the other side, the cloud can reach a large number of users using the same method: that of communicating with a user through one of his/her neighbours.

An example of the benefits that can be gained from this method is that the cloud could extend a broadcast message that contains a piece of useful information regarding the spread of a disease to as high a number of recipients as possible.

Furthermore, the members of each MANET can exchange/communicate locally when the normal infrastructure is down or unavailable.

For example, users can help each other to deal with a hazard such as a flood that has hit the area where those people are.

Another example could be exchanging a piece of information regarding an event that all the MANET members are attending, such as a football match or when performing Al–Hajj.

With regard to m–health and emergencies, a MANET member can seek help from one of the other members whereby he/she can even redirect the request to the cloud or even meet the user who requested the help in order to provide assistance.

This kind of assistance will be significant if the member who is trying to provide help is medically trained, such as a doctor or nurse, and the requested user is in a difficult situation (e.g., having suffered a heart attack). Here, a swift response can be achieved using this method.

Deployment location

There are three main locations where this model can be implemented:

- In mobile device space, which means it will be installed on the actual devices of the users.
- Near to mobile devices, at the edge of a network, such as a local server or a proxy that is connected to the same network as the mobile devices (e.g., home broadband). The idea is similar to the use of cloudlets, which were reviewed in chapter 3.
- Near to the cloud, deployed on the edge of the cloud and able to share its resources.

In comparing the three options above for where the implementation of the new model should take place, it is clear that the first option will lead to a negative impact on mobile device resources, such as draining battery life.

However, the next option overcomes this issue by deploying/running this model on a stand–alone server that is connected to the same network as the users.

This brings another advantage, such as a decrease in the time needed to receive something from the cloud because this server is usually next to the mobile device and connected to the Internet via wired cables.

Using the second option can be of help when the number of users is low because the possibility of a bottleneck will occur if the number of users (and requests) increases rapidly. From here, the third option can be considered, whereby the model will make the best use of the high storage and computational capacity of the cloud.

In other words, when the model is implemented close to cloud resources, an improvement in the cost of mobile device resource usage can be achieved, such as a reduction in the total storage and computing resources needed.

Based on the above comparison and the requirements outlined in chapter 2, the model presented in this thesis is to be deployed near the cloud in order to make the best use of its resources.

Performance requirements

As mentioned previously, the main purpose of creating MANET–cloud model is to support m–health applications in emergencies at the infrastructure level. Therefore, the following list introduces and discusses the performance requirements for the proposed model.

Robustness

One of the most important requirements for introducing this model is to make the system more robust by supporting it in coping with errors during execution and with erroneous input, as well as the ability to deal with mobility aspects such as disconnections and link breaks. Another advantage that can be gained from MANET–cloud model increasing system robustness is that this provides alternative paths for communication, such as using neighbouring links. This also helps the system to minimize bandwidth usage when all that is required by members is to connect in order to interact with others.

Response time

As the system is intended to deal with medical cases in emergencies, response time is very important and must be considered in the system design. The proposed model must minimize the time needed to respond to emergencies, including addressing latency between the users and the cloud, as well as the time needed to process tasks/requests, such as searching for a suitable doctor near the requested user's location.

Security and privacy

The nature of the task of integrating a MANET–type network and a mobile cloud raises security and privacy concerns. Therefore, the model must apply a mechanism that ensures a reasonable level of protection of users' data and protect the whole system from risks or attacks.

4.2.3 Communication medium

As one of the choices made during this research, mobile devices will reach the cloud over the Internet through wireless links such as Wi–Fi or Bluetooth. For example, a user can send a request to the cloud using a smartphone that is connected wirelessly to home broadband.

More interestingly, a wearable device (e.g., a heart rate monitor) can send an alert to the cloud directly if connecting to the Internet is possible, such as connecting to home broadband. However, the use of a cellular network is also considered, such as accessing the Internet via 3G/4G links or even exchanging SMS messaging with the cloud.

4.2.4 User types

Generally, the proposed system defines three main users who might be communicating with the cloud, as the cloud holds all the digital services proposed and enables sharing/communication between all parties in order for them to seek or provide health services.

The following subsections present each user type, together with the protocols each user needs in order to access the system.

Patients

One group of users might be patients who are looking for medical services to be delivered to them (or to someone near them, e.g., parents and a child) in emergency situations based on their needs.

With regard to access protocols, this type of user might access the system by one of the following means.

Account management

Users might access the system to create a new account to store their medical data or login to an existing account to update their profile or check stored medical data, such as blood test results.

These users will have to provide valid credentials (e.g., username and password) to be allowed to access stored data such as EMRs or personal profiles.

Seeking help

Users could access the system to obtain help in the case of emergencies. For example, to search for the nearest medical centre or retrieve useful information relevant to the requested case.

This type of access results from a query executed on the database, whereby if there are relevant or matching data on the system database, these will be sent to the request user in a different format, and could include text, images, and videos. Figure 4.2 shows a step–by–step chart of how users can send a query to the system to retrieve information relevant to their case.

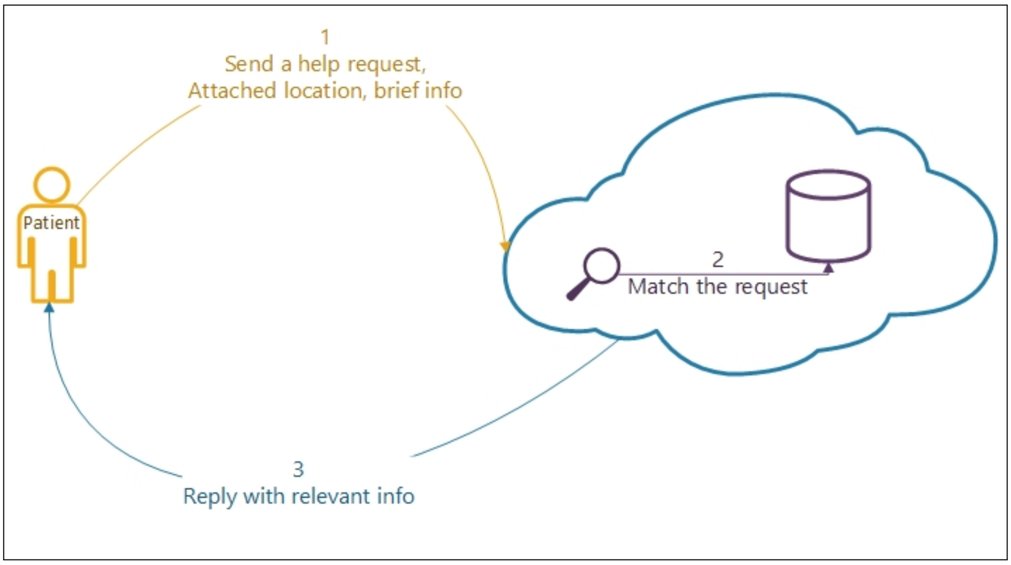


Figure 4.2: Patients Accessing the System to Seek Medical Help in Emergencies

Starting a conversation with a medical professional

Users could also contact the system to start a conversation (e.g., voice or video) to discuss a medical concern that they have. This type of access results in a look–up of the medical directory stored on the system to find a suitable person who can interact with the requested user.

After starting a conversation or session between the two users, the system monitors the status of this session until its termination for the early detection of any break in the communication links. Figure 4.3 shows how this access can be achieved.

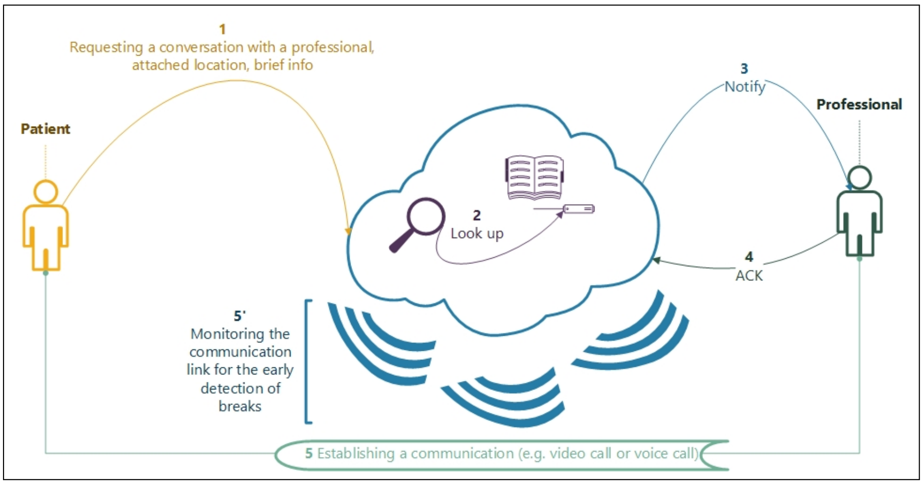

1
Requesting a conversation with a professional, attached location, brief info
Patient
3
Notify
Professional
2
Look up
4
ACK
5'
Monitoring the communication link for the early detection of breaks
5 Establishing a communication (e.g. video call or voice call)

Figure 4.3: Patients Accessing the System to Start a Conversation with a Medical Professional to Discuss an Emergency Case

Triggering an alert

In this case, users cannot access the system directly but are pre–allowed this access. In other words, a user can link a wearable device to his/her account in the cloud to allow the device to interact with the cloud without user involvement.

An example could be the medical sensors or wearable device of an elderly person living alone. When abnormalities in his/her medical status are detected by these devices, an alert is sent to the system (directly or through a mobile phone, for example). The system then redirects this alert to the nearest emergency centre. Figure 4.4 shows this access in graphic form.

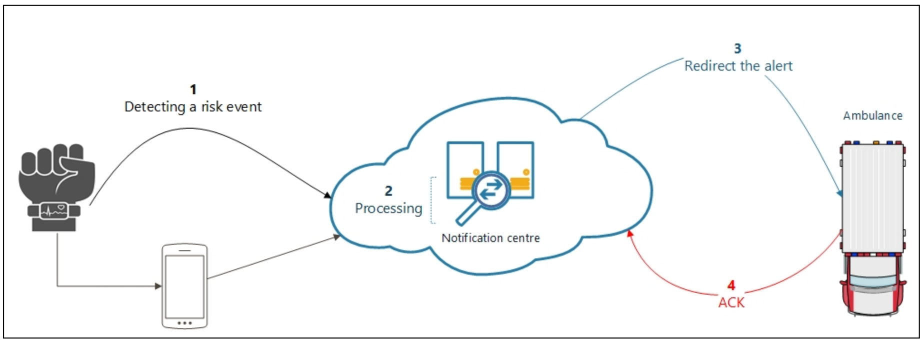


Figure 4.4: System Access Protocol for a Wearable Device to Trigger an Emergency Alert

Professionals

Another type of system user is anyone who is qualified and registered in health departments and can provide help in the case of an emergency, such as doctors, nurses, and ambulance crews. This kind of user can access the system for different purposes, such as the following.

Account access

Doctors or nurses can access their accounts in the system to update personal information, such as their qualifications and areas of expertise, and, more importantly, their status (e.g., being unavailable to deal with another case). Put simply, a professional needs to provide the correct credentials to be allowed to access the system services and update his/her current status. The protocol is, in general, similar to the account management protocol presented in the previous section.

Contacting an expert

A professional might need to contact an expert to discuss the emergency case he/she is handling. This type of access requires the professional to provide information to help the system find the most suitable expert to consult.

Once an expert is found, a communication link is established, such as a voice or video link, which is monitored by the system to detect any breaks. Figure 4.5 presents this access protocol.

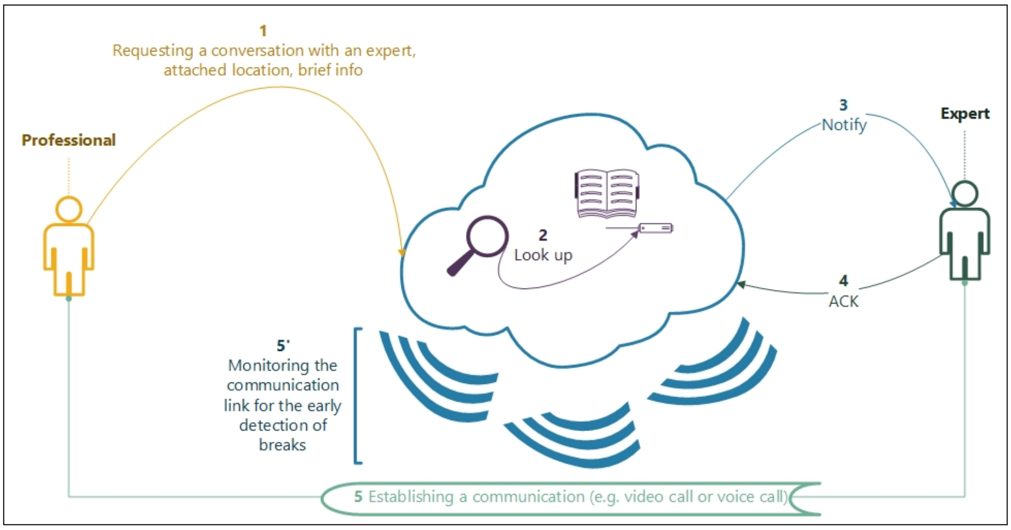

1
Requesting a conversation with an expert, attached location, brief info
Professional
3
Notify
Expert
2
Look up
4
ACK
5'
Monitoring the communication link for the early detection of breaks
5 Establishing a communication (e.g. video call or voice call)

Figure 4.5: Professionals Accessing the System to Discuss a Medical Case with an Expert

Preparation access

A professional might need to access the system to prepare a hospital to receive a patient who is experiencing an emergency if the professional is dealing with his/her medical case.

Furthermore, a professional can request specialized medical equipment based on the needs of the person who is in the emergency situation. Figure 4.6 shows how this access can be accomplished.

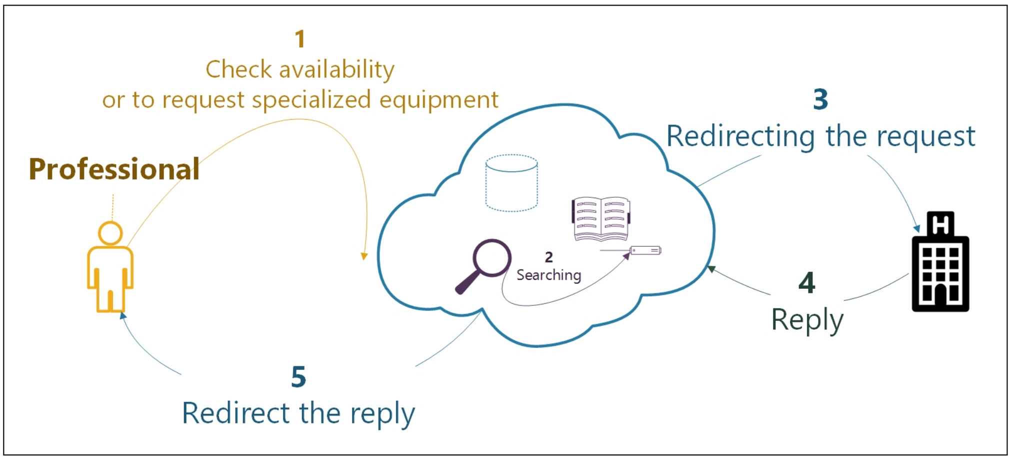


Figure 4.6: Professionals Accessing the System to Request Specialized Service from a Hospital

Health providers

Concerning access protocols to the system, this type of user might access the system to update patients' records by, for example, adding laboratory results or notifying the public of important information, such as sending an alert via the system to all users regarding the spread of a disease.

In the case of formal access, the user's details and relevant information have to be attached to the request to ensure the new records are added to the corresponding patient.

However, in the latter case, the payload of the notification is attached with the request as well as a specification of those to whom the notification will be sent (e.g., all professionals or all users). Figure 4.7 shows this access.

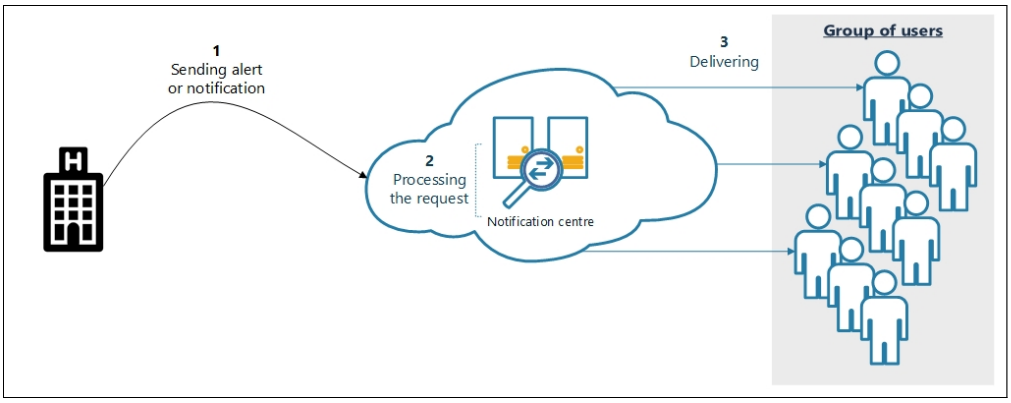


Figure 4.7: System Access Protocol for a Hospital to Send a Notification

4.2.5 Supporting services

There are a number of existing services/applications that could add unique features to such a system, such as SMS messaging or voice call ability. Another example could be the ability to analyse social media feeds.

It is clear that this type of service is not managed by the system or even a part of it but could support its main goal, which is to deliver healthcare services and support in the case of an emergency.

4.3 System architecture

Since the cloud is used to host all the proposed services, the system architecture is organized in respect to three main layers: infrastructure, generic services and application (as in Figure 4.8). From the name of each layer, it is easy to determine which components could be found there.

For instance, the infrastructure layer provides basic services, such as storage and computing. The middle layer offers everything needed by the application layer, such as databases, communication links and patient record management. Special applications are found in the application layer to allow users to advantage of the system.

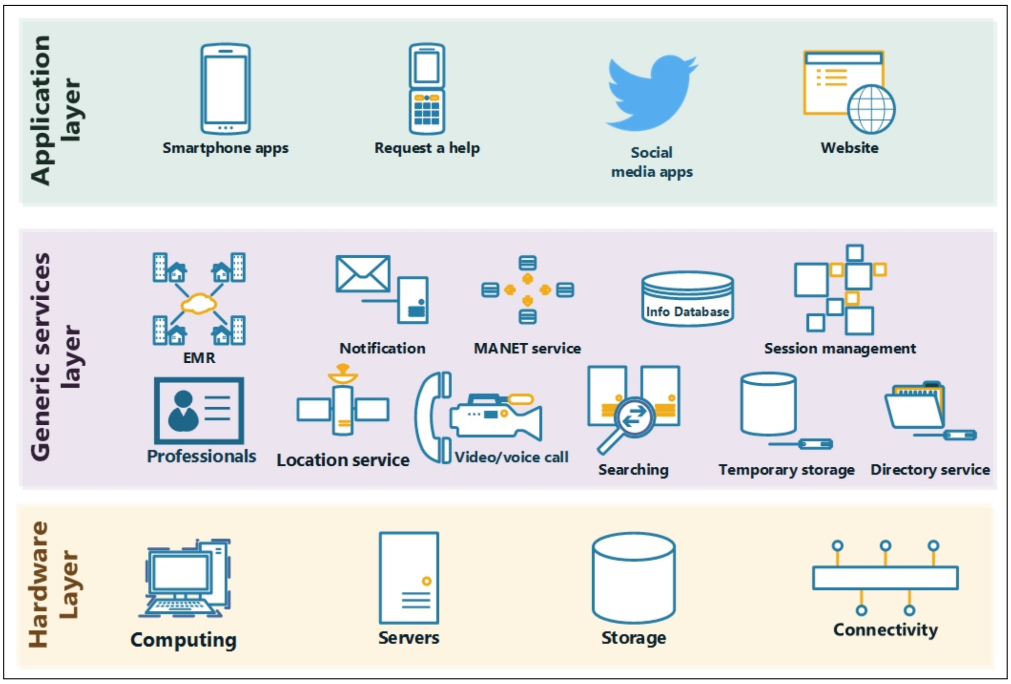


Figure 4.8: Layers of the Cloud

Infrastructure

The infrastructure layer is intended to enhance the connectivity aspects of the system by allowing mobile users to create/join self–organized peer–to–peer networks. This kind of network is then managed/monitored by the cloud and offers its users the ability to connect either to the cloud to seek care directly, or via a neighbour that has a better connection to the cloud. It also allows the members of such a network to share resources (or data) locally without the need to use a carrier/cellular network (3G/4G) with its associated costs.

Similarly, the cloud can reach users directly or via a neighbouring link. This method will increase system availability, as well as reducing or minimizing the risk of disconnection, which is one of the shortcomings of deploying mobile cloud computing technology.

The next chapter, chapter 5, discusses the implementation of the system infrastructure by introducing MANET–cloud model in detail and providing a set of experimental results.

Services

All the services are hosted in the cloud and can be accessed by mobile users; these services can operate together to deliver better healthcare services to users whenever needed.

Location, medical condition, and the connectivity status of users are tracked by the system in order to ensure the best possible outcomes. Figure 4.9 shows how users can access the cloud that hosts the services via mobile devices through a direct or neighbouring link, as stated previously.

An example of such a service could be sending a request to the cloud to retrieve medical advice that could help in delivering a healthcare service to a user who is facing an emergency situation on a motorway. Another service could be to allow users to find doctors or nurses who might only be a few steps away from them in order to discuss a medical concern.

A service could also link a monitoring device (such as a body sensor), whereby if an emergency case is detected, a request for help would be sent automatically without the need to involve the user in the operation.

In chapter 6, two proposed mobile cloud services are presented, together with the design and an evaluation of each service.

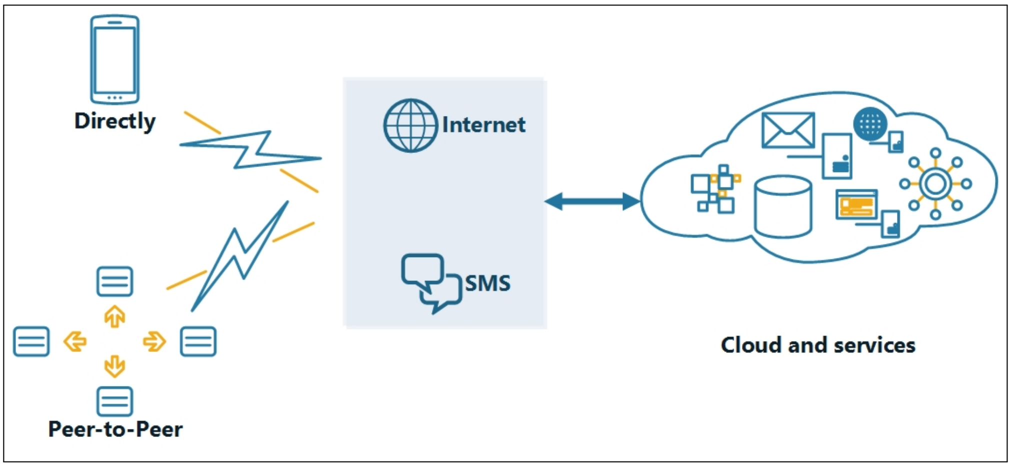


Figure 4.9: Diagram Showing How Users Can Access Cloud Services

4.4 Tools and Operating Systems

An Android operating system (OS) is chosen as the main OS for end users due to its flexibility and its open–source and well–documented API. Amazon provider is used to implement the cloud and its services (including storage needs) for the same reason. The cloud, which is hosted on the Amazon EC2, will be accessed via an Android–based app or web–based pages using a JSP or HTML format.

In practice, there is a wide choice of cloud platforms for evaluating and deploying cloud services. The most well-known cloud platform providers are IBM's Bluemix[1], the Windows Azure platform[2], and the Amazon EC2[3] that was selected for this research work. In general, choosing one of these platforms/providers is based on user preferences, as well as the deployment plan.

For instance, IBM's Bluemix can offer what is called Services as a Service (SaaS), whereby users can access several predefined services (such as IoT ability) and are able to modify these services or link them to its new cloud service, such as the Mongo database. Bluemix provides an easy user interface that requires little background in coding. However, Windows Azure and Amazon EC2 are very similar, as both can provide a Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). With regard to this research work and the preferences of the thesis author, Amazon EC2 was chosen for its well-documented API as well as its wider community.

Another way of implementing cloud services is by deploying a private cloud organized by the author of this thesis in a laboratory environment. Using this type of deployment can ensure a higher level of production to deal with the lack of privacy and security faced by the public cloud. However, a widely recognized limitation of using this approach is that a private cloud will require the purchase of hardware (a server) that would have added extra cost to the research, as well as requiring a maintenance plan that would involve another kind of cost. With Amazon EC2, platform resources (storage and a CPU) can be rented using a pay–as–you–use scheme, whereby the cost is calculated based only on the period the deployed services are used. Furthermore, subscribing to this platform meant that the author would not have to be concerned about a maintenance cost plan because it would be dealt with by the cloud provider.

[1] IBM Bluemix: https://www.ibm.com/cloud-computing/bluemix/

[2] Microsoft Azure: https://azure.microsoft.com/

[3] EC2 - Amazon Web Services: https://aws.amazon.com/ec2/

Finally, and more importantly, deploying the required cloud services on a well-known cloud platform, such as Amazon, provided the flexibility to increase and decrease the resources needed easily, an elasticity that is required for evaluation purposes such as assessing the scalability and availability of the proposed cloud services.

4.5 Chapter Summary

This chapter discussed the initial design and architecture of a system that is aimed at enhancing the delivery of healthcare information (or advice) and expand the point–of–care concept to anybody, anywhere. This means mobilizing care for people in emergency situations wherever these occur, as well as notifying health providers and establishing communication links (if needed) between all the parties concerned in order to deliver health services.

As part of the system design and architecture, a model was presented that will allow users to create and join mobile ad–hoc networks for the purpose of exchanging data locally and reaching the system services through neighbouring links if a direct link is unavailable. Possible deployment locations and performance requirements of this model were also discussed in this chapter. Finally, the chapter presented a number of services that could be deployed in the system and benefit users in the case of emergencies.

The next chapter, chapter 5, will present an important part of the proposed system: the MANET–cloud model. The following chapter, chapter 6, will then discuss the proposal for two services that are hosted on the cloud and accessible by mobile users: mobile cloud services.

Chapter 5

Managing a Mobile Ad-hoc Network in the Cloud to Support M–health Applications

5.1 Introduction

One benefit of creating a MANET [135] is that members with no direct access to the Internet can avail themselves of peers' connections and use public cloud services. Guaranteed access to cloud services is important in the case of emergency or disaster recovery situations.

As managing a MANET is a difficult task, this chapter introduces a new model that uses cloud services to manage the MANET–type networks that serve the basic infrastructure level of the purposed system. While preserving the ad–hoc nature of a mobile network, its management by the cloud provides reliability and robustness.

Furthermore, if two nodes of a MANET are engaged in a session as part of a client–server relationship [136] and one of them departs suddenly, their communication is aborted. The session is not active any more, work is lost and, consequently, the energy of the batteries has been wasted. Therefore, this chapter also introduces a framework, as part of the MANET–cloud infrastructure model, that allows the registration, saving, pausing and resuming of sessions between MANET member nodes so that both work in progress and energy are saved.

A checkpoint technique is introduced here to capture the progress of a session and allow it to be resumed in the event of an interruption.

The design and implementation of an infrastructure service to help the MANET–cloud middleware is presented in this chapter, as well as how this service was tested on Android–based devices and an Amazon cloud instance.

5.2 Problem Formulation

As stated in the previous chapter, this thesis is part of work carried out to help m–health systems to deliver healthcare and support to the public reliably and efficiently. In other words, providing robust and reliable connections between users (including patients and medical professionals) and the cloud in which the health services are hosted.

Therefore, the idea is to allow users to create/join a MANET to share content and information locally, as well as being able to reach cloud services in the case if no Internet access is available. On the other side, the cloud or the system administrator (e.g., an emergency department) can disseminate useful information in the case of an emergency, as well as reaching/notifying a missing user (e.g., someone who is missing during an Al–Hajj event) through one of his/her neighbouring network links.

However, MANET face some management issues, such as the high mobility of members and frequent changes in its topology (e.g., members joining or leaving suddenly). Therefore, the proposed MANET–cloud model is intended to:

- Address some of the challenges of MANET management and provide reliable and robust solutions to difficult operations, such as join/leave, split/merge.
- Track/save MANET members' behaviour (such as current location and connectivity strength or quality) to assist them in the case of an emergency.
- Collect information about each MANET (e.g., size or density) to allocate/deallocate resources smartly (e.g., storage and bandwidth).
- Propose a solution to deal with session breaks during execution and to help active sessions to be registered, saved and resumed. Doing so will avoid executing the whole session from the beginning, as is normally the case in MANET.

5.3 Service Supporting Infrastructure Design

As mentioned, MANET–cloud model is part of the proposed system presented in the previous chapter (chapter 4), in which a MANET is used with the mobile cloud to enhance the system infrastructure level. MANET face some limitations, such as the mobility of its members and a short lifetime.

Another important limitation is that mobile devices are scarce in resources, especially in terms of energy and storage. In contrast, the cloud has extensive computing resources that can be allocated on demand and energy is not an issue.

With regard to connectivity, communication between users and the cloud can be provided either by cellular networks (e.g., 3G/4G) or Wi–Fi, which allows the following assumption: at least one device in the network will be connected to the cloud at a time, all the time.

A MANET management service is created in the cloud and part of the MANET–cloud model has the role of managing all registered MANETs individually. When a MANET is created, it will register with the cloud and a new instance of the MANET management cloud service will then be created and allocated to it.

This service instance will monitor MANET membership and communication link status between individual MANET devices and the cloud. In this way, the cloud service can select the best communication link with each MANET or switch from a link with a device that is departing to the best quality link among the remaining devices.

One important consequence of this model is that in areas with a large density of mobile devices, the number of data communication links can be decreased, thereby reducing overcrowding of the radio spectrum.

Furthermore, the cloud allocates identifiers to MANETs and IPs to joining members. This centralized management of addresses avoids the expensive operation of checking for duplicates whenever a new device joins the MANET or when networks merge.

In addition, the MANET management cloud service can store MANET information on a constant basis.

5.3.1 MANET–cloud management function

In order to avail themselves of the service, each user who has an Internet connection and is able to reach the cloud is required to create an account in the cloud by providing a unique username and a password. Generally, a new cloud account can be created before using it in an emergency.

However, when the user sends a "creating an account" request to the cloud, this request will be checked against the cloud's database (DB). If the details are found in the DB, the cloud returns a suggested MANET_SSID to the user's mobile, as well as an allocated IP address.

This MANET_SSID is created based on user location and the capacity of each MANET in which the user is expected to provide a new SSID that must be unique and verified by the cloud. The IP address is allocated by a Dynamic Host Configuration Protocol (DHCP) [137] server running in the cloud to avoid duplicates.

When the user details are validated, a profile table is created in the mobile device DB, which contains account information such as username, password, MANET_IP and MANET_SSID. Afterwards, this user can access the cloud services. Figure 5.1 shows how this function works.

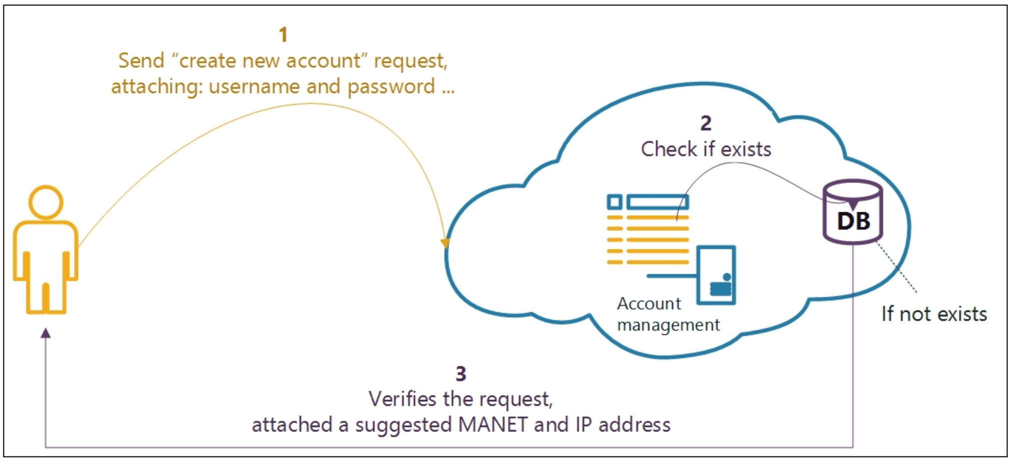

1
Send "create new account" request,
attaching: username and password ...
2
Check if exists
DB
Account
management
If not exists
3
Verifies the request,
attached a suggested MANET and IP address

Figure 5.1: Creating a Cloud Account by a User

5.3.1.1 Starting–up a MANET protocol

Once an account procedure has been conducted successfully and the user owns a valid cloud account with the suggested MANET_SSID and IP address, the user can start this new MANET.

When the MANET is started successfully, a "Hello" message is periodically broadcast to draw the attention of new members who have just joined the MANET.

The device also listens for any traffic in its radio range to detect any new members to add them to the neighbours table (this is also called a routing table).

However, users are allowed to change the MANET_SSID but this new MANET_SSID first has to be verified by the cloud, whereby a new IP address is allocated to this user as well. Then, the MANET can be started up as presented previously. Figure 5.2 shows how a user can create a new MANET and start it up.

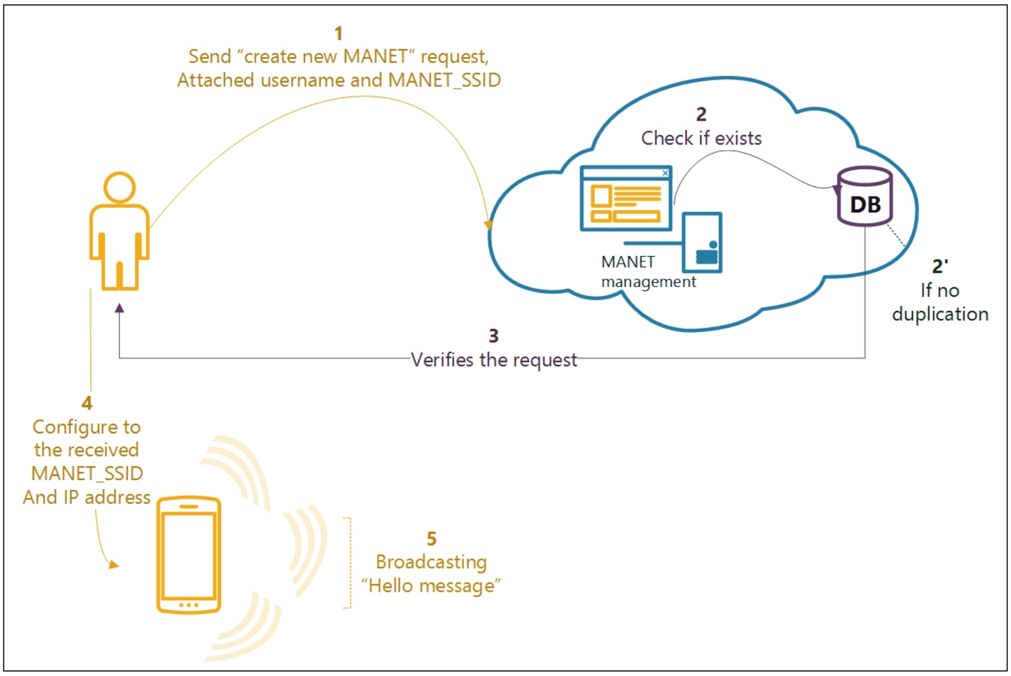


Figure 5.2: Starting up a New MANET

5.3.1.2 Joining an existing MANET protocol

The procedure for joining an existing network depends on the availability of the cloud. The following two subsections present how a user can join an existing MANET depending on the availability of the cloud.

The cloud is unavailable

If the cloud cannot be reached, the user's device can be configured to a global MANET and the IP address is set statically from the device's MAC address to avoid conflict (self–allocation).

A global MANET is an open MANET network in which some members have a direct connection to the cloud and some do not. The idea behind this network is to provide a form of connectivity when users cannot reach the cloud directly.

Members in this network can use neighbours' links to communicate with the cloud. Similarly, the cloud can communicate with those members who do not have an Internet connection through those who have an active link with the cloud. When a member acquires the ability to communicate with the cloud directly, one of the following scenarios will occur:

- The member *verifies* its information with the cloud and stays in the global network.
- The member *leaves* the global network and either creates a new MANET or joins an existing one.

The cloud is available

When a mobile device is able to reach the cloud, it will successfully login to its account. It can then issue a request to join an existing MANET. The cloud replies by including all the available MANETs in its vicinity that are found in the DB. After the requester chooses a network from the list, the cloud allocates resources, such as a central processing unit (CPU), memory, storage, and networking resources and an IP address to be used in the chosen network. Figure 5.3 summarizes this case.

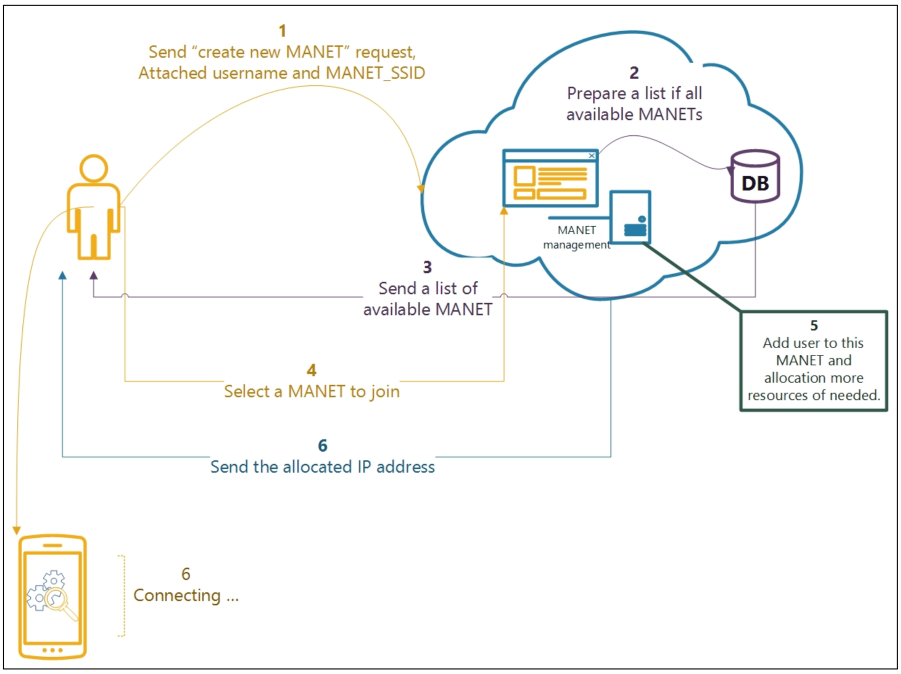


Figure 5.3: Joining an Existing MANET

5.3.1.3 Monitoring MANET activity

The cloud tracks each MANET usage and its members' behaviour to enable better performance. For example, the cloud can switch from one link to another to communicate with a particular member. By gathering information about network activity, the cloud can balance networks, such as by adding more members or redirecting them to another network to avoid overcrowding when other networks are scarce or inactive. Indeed, all information can be analysed to provide statistics for the service provider.

However, each member of a MANET can periodically send a message to the cloud for identification of its status and confirmation of the network to which it is connected. Alternatively, to save energy, a member will contact the cloud only when its status is about to change or has changed, such as departing/departed. On the other side, the cloud has to update the users' records when status messages are received, as well as checking inactive users by sending a "check status" message.

For the purpose of tracking each MANET efficiently and providing services to members who do not have an active link, the cloud chooses one of the MANET members as a cloud agent. The main role of this agent is to feed the cloud information concerning its connected MANET, such as MANET size and members joining/departing. The cloud can also reach members through this agent, as well as being able to broadcast messages to the whole network. Initially, the cloud picks the issuer of a new MANET as a cloud agent of the MANET. Then, if other members start to join the MANET, the cloud can change the MANET agent if there is another node that has a better link. However, if the agent has left the network, the cloud will start to check the status of all links, whereby the node with the best link with the cloud will have a higher chance of being selected. After choosing a MANET agent, the cloud starts to retrieve MANET information from that agent.

Furthermore, a MANET can have more than one cloud agent depending on the needs of this MANET (e.g., an overcrowded network with a high number of active users). This will avoid a bottleneck occurring in the cloud agent node due to high traffic through this node (either from the cloud to other members or from members to the cloud. Figure 5.4 shows some MANETs managed by the cloud with either one cloud agent or more.

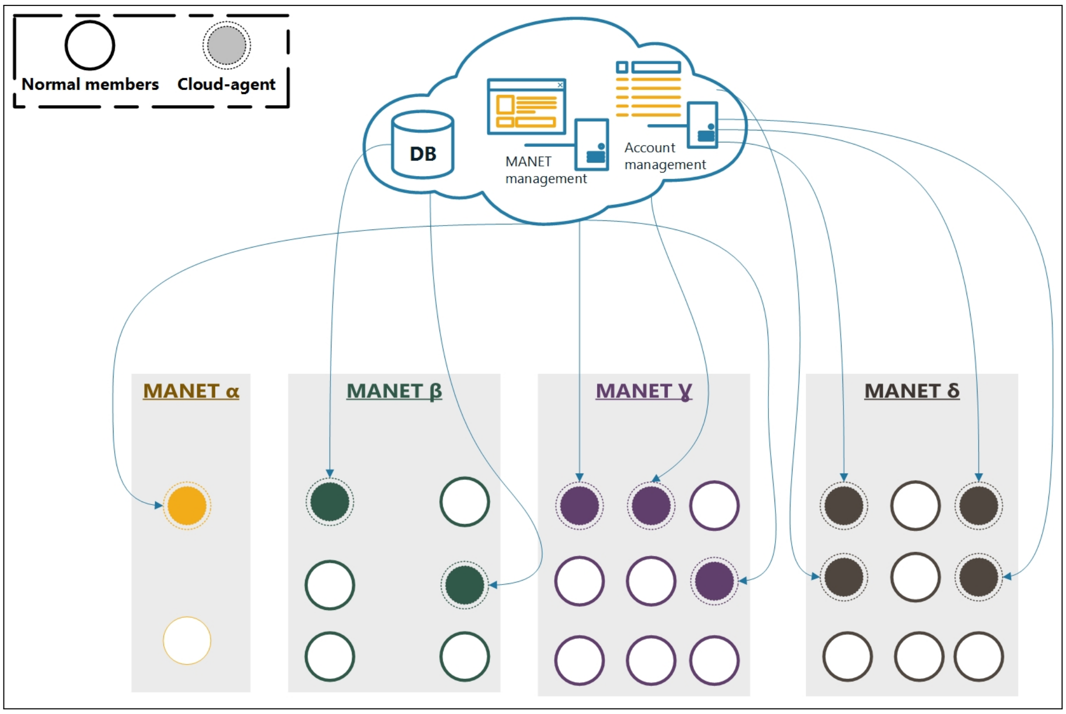


Figure 5.4: MANET and Cloud Agent Node(s)

5.3.1.4 MANET operations

To provide better performance for users, the cloud will intelligently perform actions based on statistics gathered from monitoring MANET behaviour such as a **split** or **merge**. The following subsections present these two operations in detail.

MANET split operation

A MANET split action involves separating an existing MANET into two (or more) networks. The cloud can split an existing network based on certain parameters, such as the MANET size, the purpose of creating/joining this network, and network activities. For instance, if a network is found to be congested and overcrowded, a split operation is practical.

Another example of implementing a split operation is to group members according to some criterion, e.g., role — doctors, nurses, firefighters and so on.

The cloud starts the split operation by listing users who will move to a new network. It then distributes the new MANET_SSID to those users for whom the IP addresses are kept as they have already been allocated from the cloud. When users receive this announcement from the cloud, they disconnect from the old network.

Figure 5.5 is an illustration of how the cloud splits a network into two separate ones, where the old SSID is kept are where the following pseudo code (Algorithm 1) presents this operation programmatically.

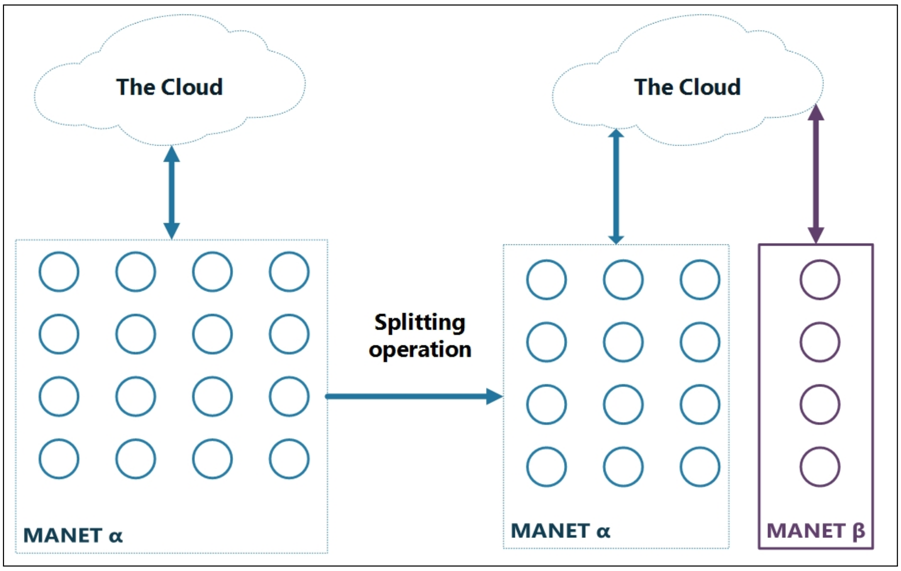


Figure 5.5: The Cloud Splitting a MANET into Two Separate

Algorithm 5.1: The Pseudo Code that Demonstrates a Split Operation in Both Side: The Cloud and Mobile Devices

```
Function The cloud side:
    Create a new MANET_SSID;
    List users who will move to the new MANET;
    Distribute the new MANET_SSID to users in this list;

Function Mobile side:
    Received "split request";
    if the new MANET_SSID != the connected MANT_SSID then
        Disconnect from the current MANET;
        if the new IP address != the current IP address then
            Configure to the new IP address;
        Connect to the new MANET;
        if Done then
            ACK the cloud
        else
            Notify the cloud attaching error log
    else
        Ignore the request
```

However, this operation can also be started by mobile nodes when the departure of one (or more) node(s) is detected. The cloud detects that the network is about to split in different ways, such as fading signals from some MANET members, not receiving anything from some members, or retrieving network operations via the cloud agent.

In practice, the cloud marks users' status to one of three categories: active, suspended and inactive. Active users send and receive messages using the cloud. In contrast, inactive users have left their MANET after notifying the cloud of their departure.

Between these two types, there are suspended users, where communication from/to those users suddenly stops without any previous notice.

Losing communication between the cloud and a user can be the result of various different circumstances, such as the mobile devices themselves (e.g., a dead battery), networking issues (e.g., no Internet access), or connecting to a MANET that is not registered in the cloud.

Therefore, the cloud has continually (every 10 minutes for example) to try to reach suspended users to update their status by sending a direct message to that member or retrieving its status from the cloud agent of its MANET.

One possibility is that a suspended user was connected to MANET α and it is found that it is now connected to a new MANET β, which is not registered in the cloud DB. The cloud then checks if there are other members of MANET α that are now members of MANET β.

If that is the case, the cloud considers that MANET α has split into two networks: one that has the same SSID and a second one that has a new SSID, e.g., β. Finally, the cloud starts monitoring the new MANET to capture its user behaviour as well as to allocate resources to this MANET.

Figure 5.6 provides examples of users' status where, as in the last example, splitting is detected by the cloud.

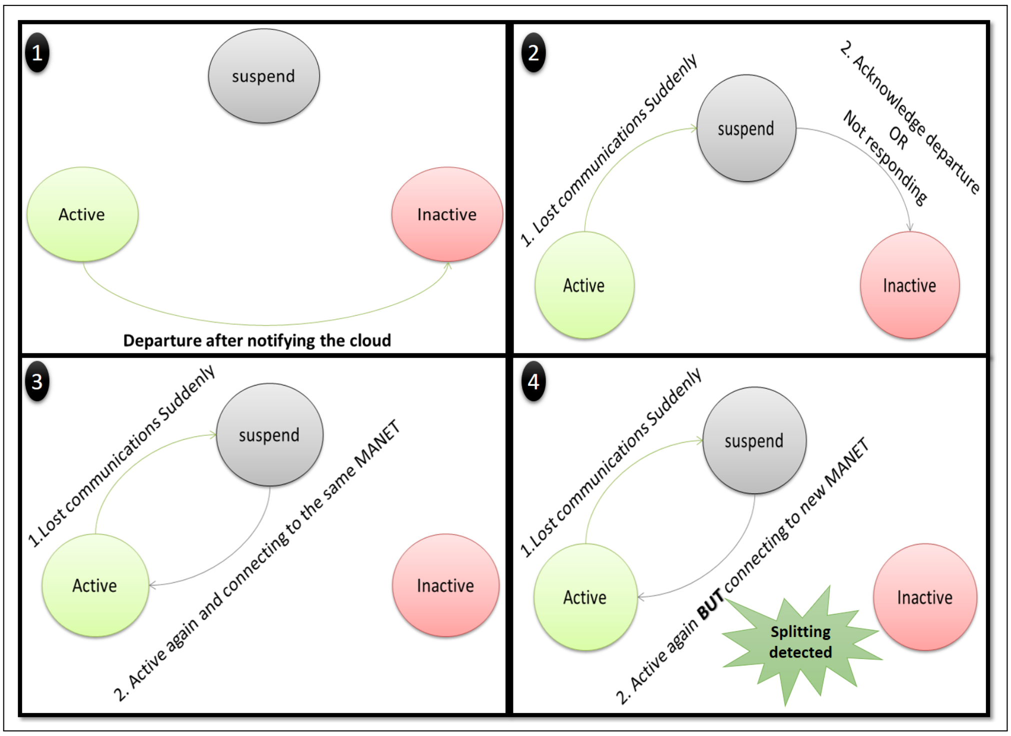

1
suspend
Active
Inactive
Departure after notifying the cloud
2
1. Lost communications Suddenly
suspend
2. Acknowledge departure
OR
Not responding
Active
Inactive
3
1.Lost communications Suddenly
suspend
2. Active again and connecting to the same MANET
Active
Inactive
4
1.Lost communications Suddenly
suspend
2. Active again BUT connecting to new MANET
Splitting detected
Active
Inactive

Figure 5.6: Users' Status

MANET merge operation

Essentially, managing a high number of MANETs will affect system performance. Therefore, the cloud first has to ensure that creating a MANET is not being used as an attack to harm the whole system. The cloud then has to reduce the number of MANETs to provide a reliable service to users.

One of the models for reducing MANET numbers is by merging two (or more) MANETs to a single MANET. The merge operation can be performed by the cloud when needed. If two small MANETs are located in the same area, merging these will be beneficial.

To ensure that merging is accomplished efficiently, the cloud keeps IP addresses of both networks' members. The SSID of one of these networks will be used for the new network. Choosing which SSID will be used depends on the size of each network. In other words, the SSID of the MANET that has the higher number of members will be used. The following pseudo code (Algorithm 2) presents this process:

Algorithm 5.2: The Pseudo Code that Demonstrates a Merge Operation in Both Side: The Cloud and Mobile Devices

```
Function The cloud side:
    Start merge operation;
    List MANETs;
    Set MANET_SSID = the highest MANET size;
    Distribute merge request to MANETs' members;
Function Mobile side:
    Received "merge request";
    if the new MANET_SSID != the connected MANT_SSID then
        Disconnect from the current MANET;
        if the new IP address != the current IP address then
            Configure to the new IP address;
        Connect to the new MANET;
        if Done then
            ACK the cloud
        else
            Notify the cloud attaching error log
    else
        Ignore the request
```

However, merging can occur between users without involving the cloud. If that is the case, the cloud has to have a mechanism for detecting this event.

For example, users on a train might create a new MANET (called α) to share data and then arrive in another city that has a number of MANETs, in which most of those users start to join a particular MANET (called β). The merge has to be detected by the cloud for a number of reasons, such as the allocation of extra resources.

In practice, the cloud watches members' activities (joining/departing) that are sent by the cloud agent mentioned previously. If the cloud finds that a number of members that are connected to another MANET have started to join the agent's network, the cloud assumes that a merging operation is in progress.

When merging is detected, the cloud releases resources allocated to the old network and allocates more resources to the MANET that is left.

In the case of the old network not being known to the cloud, a record of what has been done will be created in the cloud DB. Figure 5.7 shows an example of the merging operation.

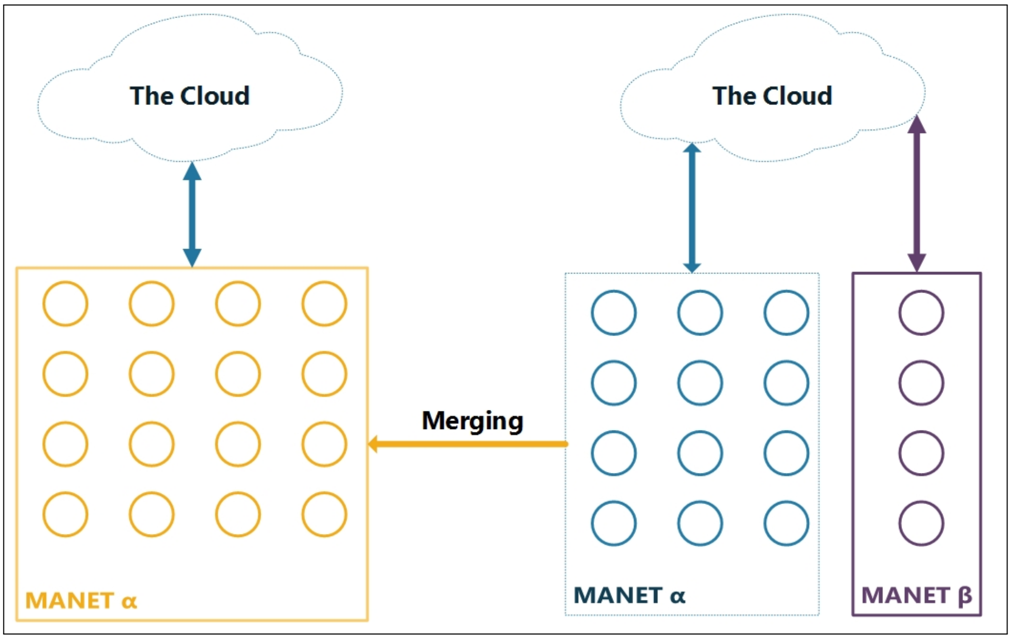


Figure 5.7: Merging Operation Performed by The Cloud

5.3.2 MANET session management

One of the most important benefits of introducing cloud services is saving an active session between MANET nodes, even if one of the two parties has left the network. These can be nodes in a client–server relationship and they can be engaged in active sessions. If, due to mobility, the communication links are broken, the sessions are affected. In a traditional MANET, if two nodes are communicating to do a job and one of them departs suddenly, the communication is aborted. The session is not active anymore and all work is lost.

However, when two nodes registered in the cloud have an active session and one of them is unexpectedly disconnected, the session can be saved. The break in the link is determined by the lack of acknowledgement from the departed node. The member of the MANET will start the session–saving procedure in the cloud, assuming the session is also saved on the receiver side. It will then return to being active when the sender sends a request to the cloud looking for the disconnected node to resume that job. The cloud has the role of acting as a bridge between these two nodes to keep the session active.

In this idea, a checkpointing technique is used to capture the progress of an active session. Therefore, each user creates a checkpoint that contains what has been performed so far. The most recent checkpoint is stored in the cloud when a disconnection occurs. As a result, when both users become active again, the session is resumed from this checkpoint instead of starting the session from scratch.

The following subsections discuss this idea in detail, starting with an overview of the session protocol and then going into execution, checkpointing, and resumption in greater depth. The section ends with some of the security mechanisms designed to ensure that a reasonable level of security is provided.

5.3.2.1 Protocol overview

A protocol execution is triggered when a user sends a registration request to the cloud to register a session. The registration request includes application details (ID, name, description, payload, etc.) and sender/receiver details (IDs, MANET, IPs, etc.). There are two reasons for this step:

1. The session will be linked to users' accounts in the cloud, which means users can browse all previously performed sessions at any time and use any mobile device once they provide valid credentials.

2. The storage needed to execute such a session will be offloaded to the cloud, which will also provide a way of resuming a suspended session when the users cannot communicate directly.

Once the request is received by the cloud, the cloud generates the following:

1. A registration ID: this is a unique number for identifying each session. The cloud is responsible for generating this ID and ensuring its uniqueness before sending it to the requester.

2. Security keys: these are two different keys—one for the sender and one for the receiver. The proposed system employs the username of each user and the session ID as a seed to feed the hash function to create a hash value. Once these unique hash values are created and sent to the corresponding user, they are also stored in the cloud DB. Therefore, when one of the users wants to resume a session, or even access its content, he/she has to provide this hash key to be able to start the process. The cloud checks the hash key provided to ensure that it matches the one pre–stored in the DB.

Furthermore, the cloud is responsible for saving the information from each session in a DB that is hosted in the cloud. The cloud also stores any necessary data or payload that is required to execute the session. An example could be a text or image that is involved in a session. Each session's information and data will be mapped to the session ID.

Finally, when the requester receives the session ID and the two security keys, the session can start, and the cloud waits for any update in the progress of the session. These steps are presented in Figure 5.8.

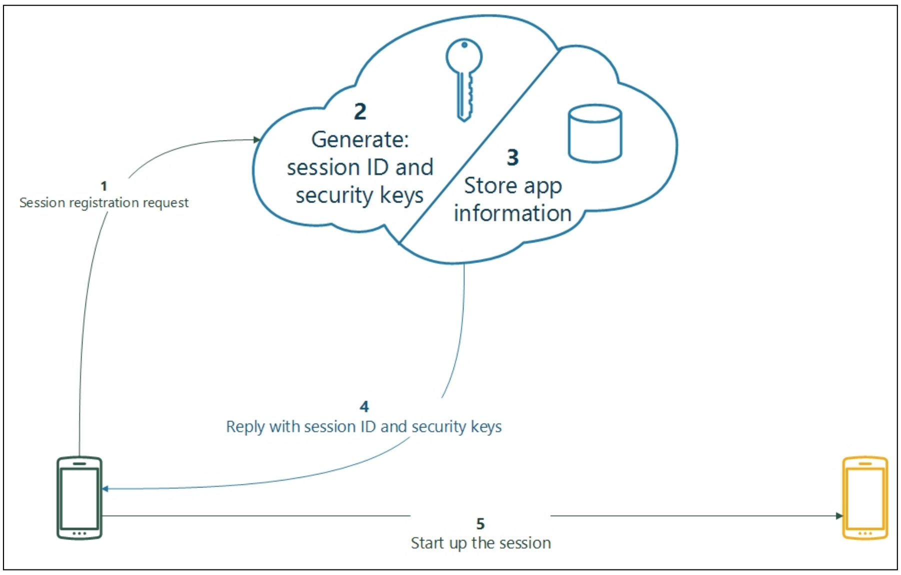


Figure 5.8: The Sequence of Operations Needed to Start a New Session

When a session is established, the sender will create a new checkpoint each time an acknowledgement (ACK) is received from the receiver. On the other side, the receiver will create a new checkpoint every time a new message is received. All the checkpoints are stored locally in a mobile device DB to minimize communication with the cloud.

Only the most recently stored checkpoint will be sent to the cloud if a disconnection occurs, unless an intensive checkpointing model is chosen. In the latter case, each time a checkpoint is created, it will be uploaded to the cloud (more details about checkpoint models are provided in the next section).

If the session is completed, the sender notifies the cloud about the completion, whereupon the cloud stores the session information in the log table in its DB, marks the session as "completed", and deletes all unnecessary data, such as payload and checkpoints.

Figure 5.9 shows an exchange of messages between the sender, the receiver, and the cloud to perform a session that is completed without any interruption.

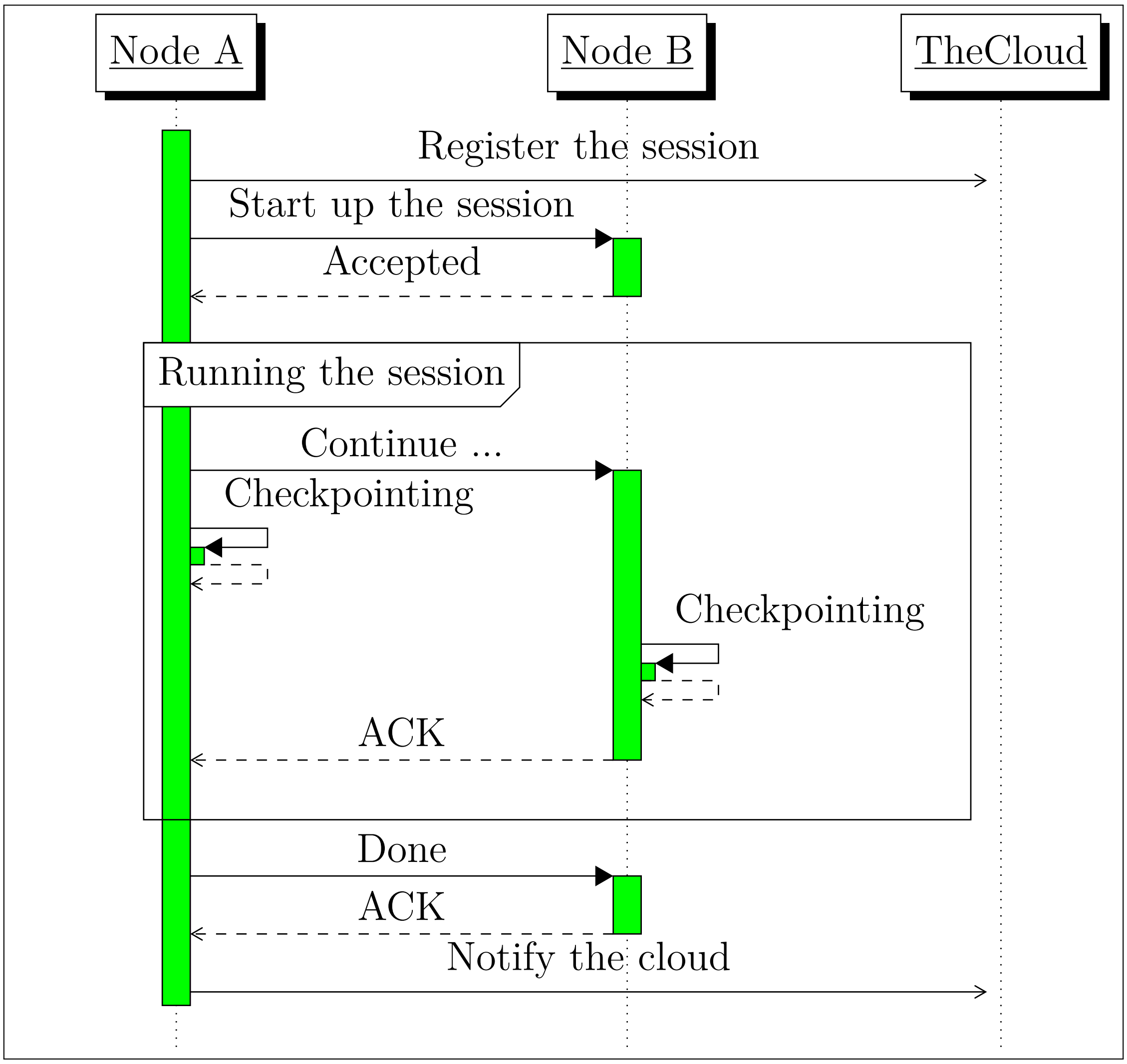


Figure 5.9: A Sequence Diagram Showing the Start and Completion of a Session without Interruption

However, if a session is aborted, the sender sends an "uncompleted session" request to the cloud, attaching the most recently created checkpoint to the request. It is assumed that the receiver also has the recent checkpoint saved locally.

The cloud updates the status of the session as "interrupted" in its database, then starts monitoring both users' connectivity and sends a reminder when both users are reconnected. Figure 5.10 provides an illustration of this operation.

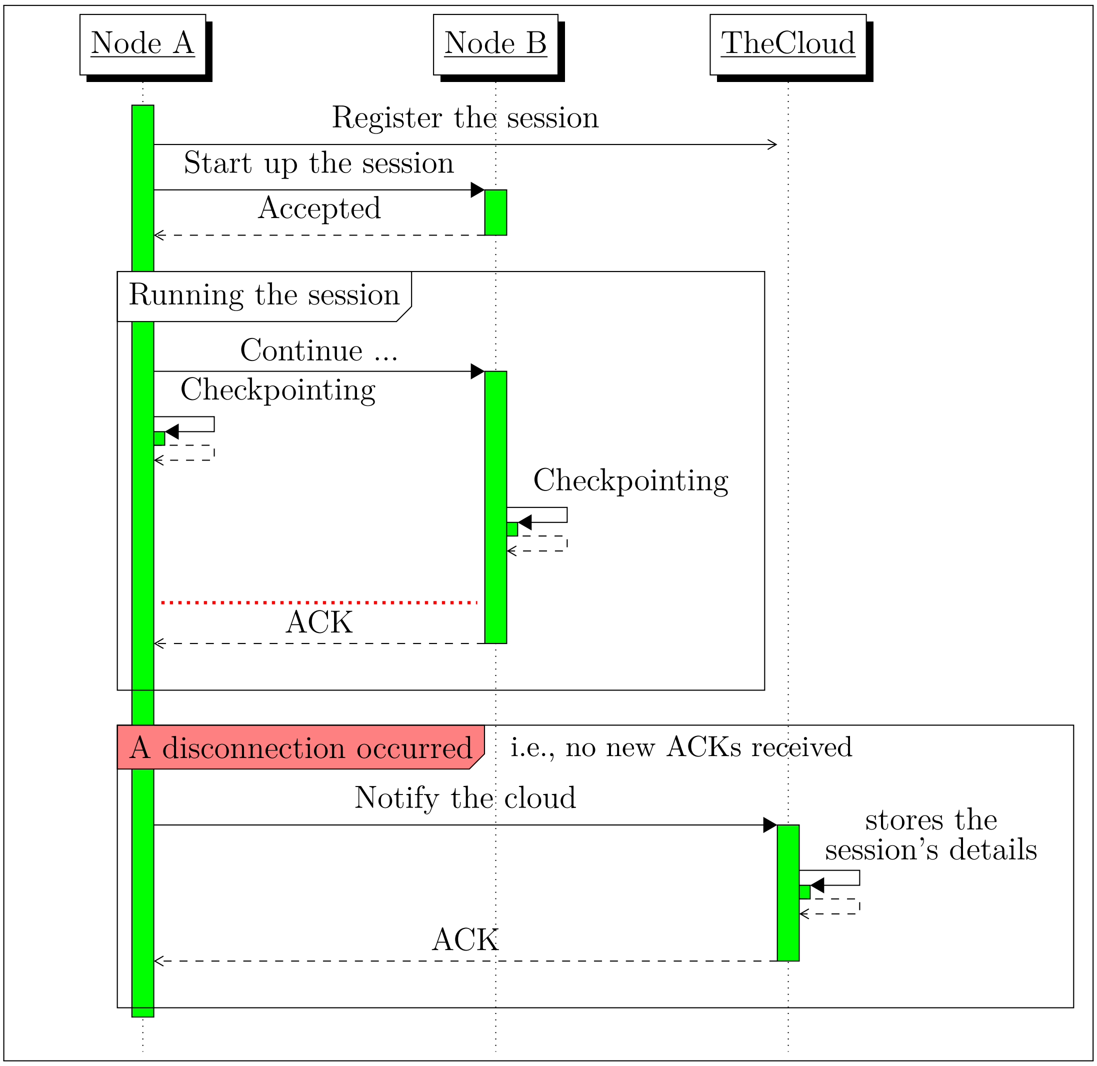


Figure 5.10: A Sequence Diagram of a Session that has been Interrupted

One of the most important benefits of introducing the cloud here is that, if a session is interrupted, the cloud allows the sender to search for the receiver.

If that receiver is not reachable to start the resumption process, a search is undertaken by the cloud by sending a “check status” message directly to the receiver or through a cloud agent, as reported previously.

Checkpoint framework

The main purpose of creating checkpoints is to determine the progress of a session. This means that by reading such a checkpoint, the session can be resumed without the need to start again from the beginning.

Therefore, each checkpoint has to include information such as application details and what has been done so far, as well as what is left to complete the session. A new checkpoint is created each time any interaction between the sender and the receiver occurs.

Checkpoints can be useful when a session is a long one and there is a strong possibility of it being interrupted. However, in some cases, checkpoints could add an unnecessary overhead to a session. Therefore, different checkpoint models are proposed here.

Intensive

This creates a checkpoint each time a new interaction occurs and, at the same time, stores it in a device database, as well as uploading it to the cloud. The benefit of using this model is that it will save mobile storage: users can delete checkpoints and retrieve them when a resumption process is started, even if this is done from different devices. However, this procedure adds extra cost in terms of time, energy, and bandwidth.

Normal

This creates a checkpoint locally and only if a session is interrupted will the most recent checkpoint be uploaded to the cloud.

Local

This creates a checkpoint locally each time a new interaction occurs. Checkpoints will not be uploaded to the cloud at all, even if the session is interrupted.

None

The checkpoint technique is not used at all.

Retrieve checkpoints

The latest, or every, checkpoint can be retrieved in the case that it has been deleted due to the limited storage space of the mobile device. Retrieving a checkpoint can be done locally between the sender and the receiver (as in Figure 5.11) or from the cloud (as in Figure 5.12). In the latter case, the cloud has to check if the user is part of the session or is eligible to retrieve a checkpoint from the sender.

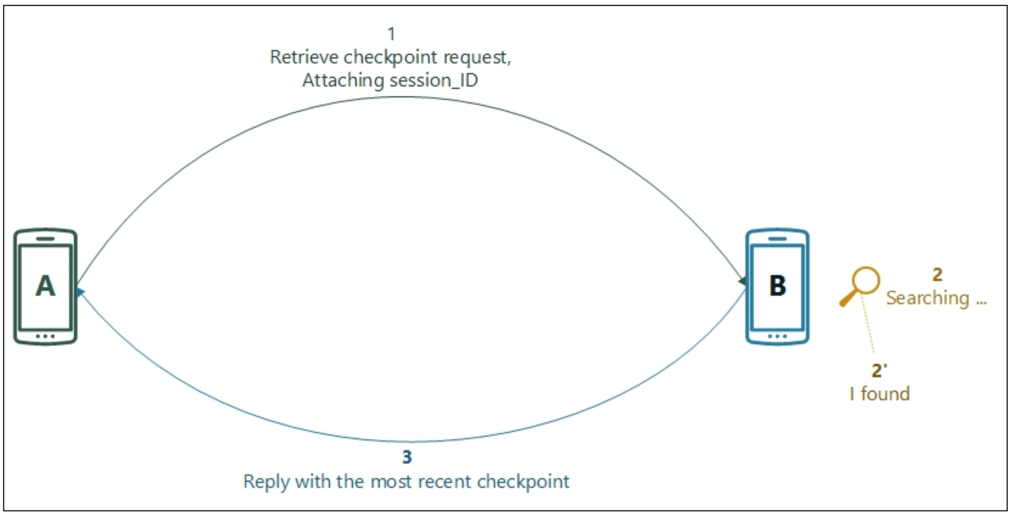

1
Retrieve checkpoint request,
Attaching session_ID
A
B
2
Searching ...
2'
I found
3
Reply with the most recent checkpoint

Figure 5.11: Retrieving Checkpoints Locally

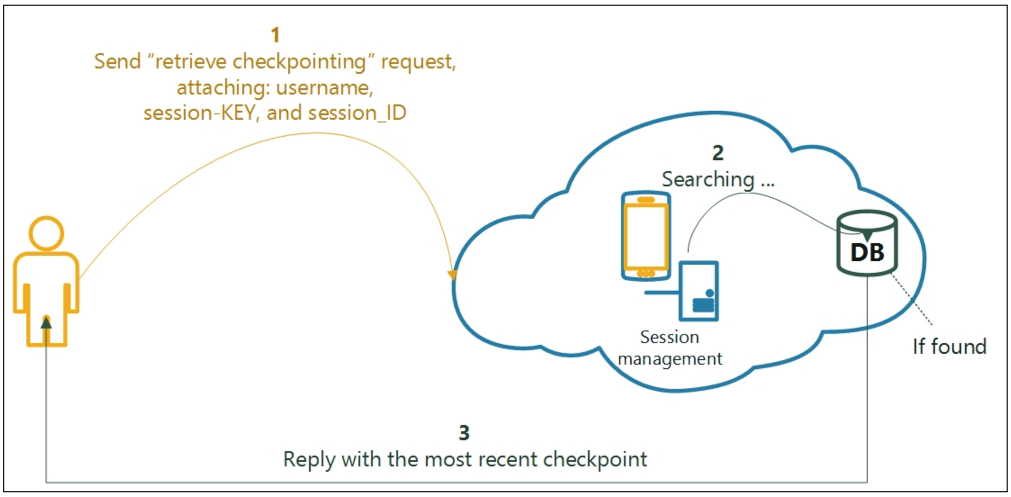

1
Send "retrieve checkpointing" request,
attaching: username,
session-KEY, and session_ID
2
Searching ...
DB
Session
management
If found
3
Reply with the most recent checkpoint

Figure 5.12: Retrieving Checkpoints from The Cloud

5.3.2.2 Resumption framework

When a session correspondent does not receive any new ACKs from the other party, it will be assumed that a disconnection has occurred. Resuming a session after an interruption will either be done by one of the users involved in the session or by the cloud. The following subsections present these two scenarios.

Resumption by the sender

The sender can resume a suspended session from the latest checkpoint when the correspondent reconnects. However, if the correspondent is not reachable because it is not connected to the same MANET, is not a one–hop neighbour of the sender, or is not connected any more (for example because of a dead battery), the sender can issue a request to the cloud to look for the receiver to finish the session. The result of the look–up will be one of the following (see also Figure 5.13):

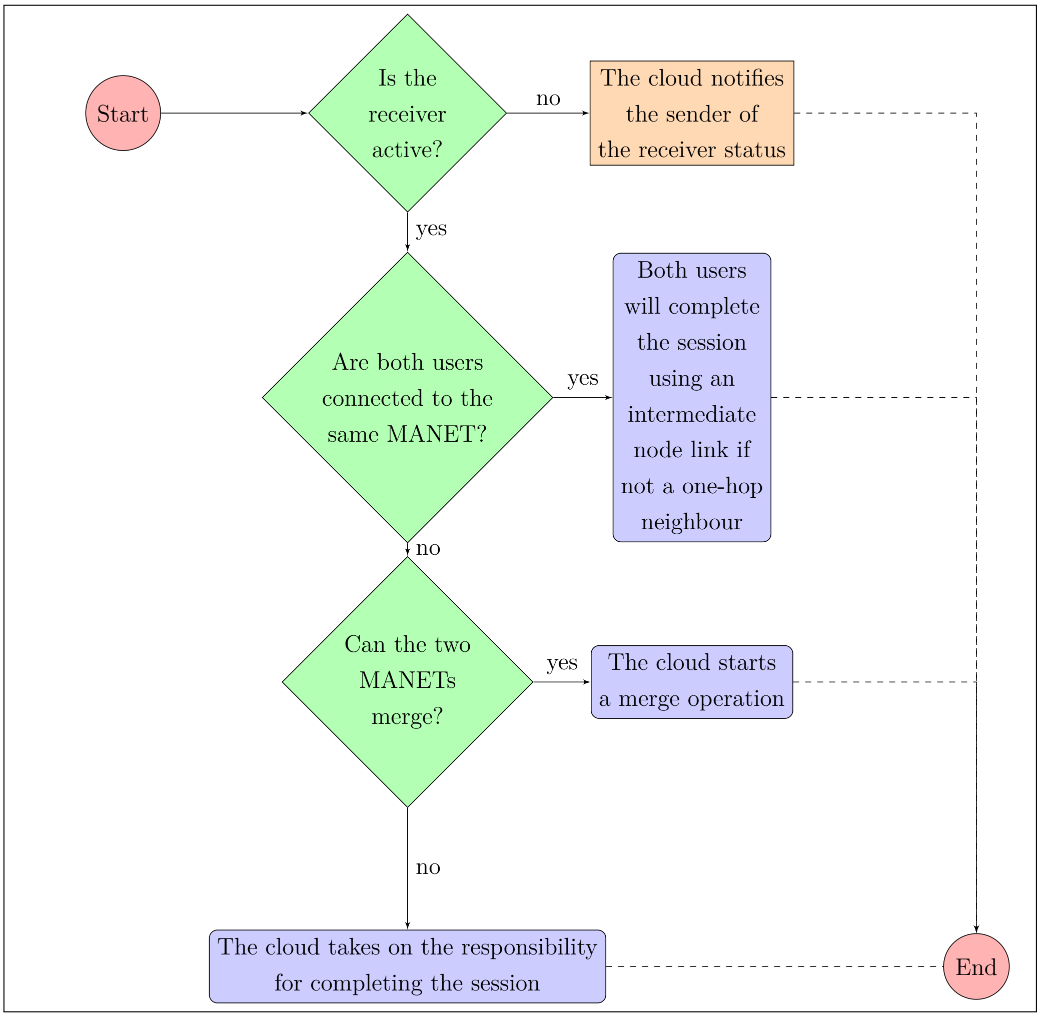

Start
Is the receiver active?
no
The cloud notifies the sender of the receiver status
yes
Are both users connected to the same MANET?
yes
Both users will complete the session using an intermediate node link if not a one-hop neighbour
no
Can the two MANETs merge?
yes
The cloud starts a merge operation
no
The cloud takes on the responsibility for completing the session
End

Figure 5.13: The Cloud Replies to a "Resuming Session" Request

- The receiver is not connected to the cloud and its status is inactive: the cloud notifies the sender of the receiver status.
- The receiver is connected to a different MANET: the cloud will merge either the two MANETs, as presented previously, or takes on the responsibility for completing the session, which means the cloud acts as a bridge between the two.
- The receiver is connected to the same MANET but is not a one–hop neighbour of the sender: both users will complete the session using an intermediate node link.

Resumption by the cloud

If a session status is marked in the cloud as "paused", the cloud will monitor the connectivity of the two users involved in the session. When both users are reconnected, a notification message is sent to the user who has started the session. The action the cloud takes will depend on the reply from that user (as in Figure 5.14).

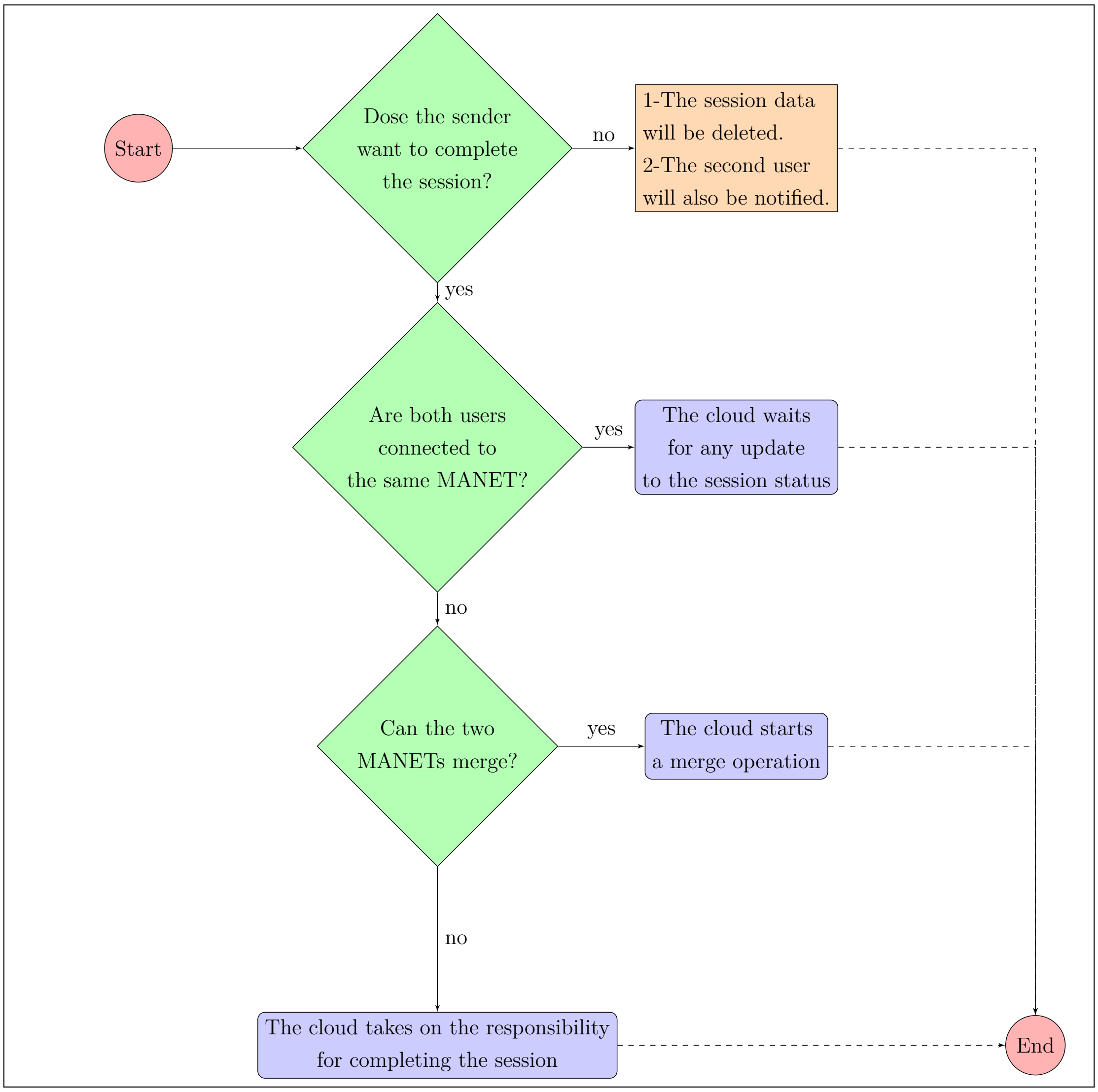

Start
Dose the sender want to complete the session?
no
1-The session data will be deleted.
2-The second user will also be notified.
yes
Are both users connected to the same MANET?
yes
The cloud waits for any update to the session status
no
Can the two MANETs merge?
yes
The cloud starts a merge operation
no
The cloud takes on the responsibility for completing the session
End

Figure 5.14: Flowchart Demonstrating the Possible Replies from the Issuer of a Session

- If the user is interested in completing the session, the cloud will then check to which MANET each of the users is connected.
  - If both users are connected to the same MANET, the cloud waits for any update to the session status.

  - However, if the users are connected to different MANETs, the cloud takes on the notification finishing the session.

- In contrast, if the user is not interested in completing the session, the session data and details will be deleted. The second user will also be notified that the session has been ignored on the sender's side.

5.3.2.3 Security mechanisms

MANET networks suffer from specific issues because of their particular characteristics, such as a lack of a centralized access point, constrained battery life, weaker overall security, dynamic topology, slower data transfer rates, and lower bandwidth. These unique characteristics increase the likelihood of attacks. An overview of common attacks on and threats to MANETs has already been provided [138].

However, the following three aspects need to be considered:

1. **Ownership:** it is important to ensure that users who are participating in such a session are registered in the cloud and their details are known. This will reduce the risk of receiving requests from unknown users that may harm or slow down the whole system.
2. **Eligible access:** the system needs to ensure that only users who are part of a session can access its content and are able to resume the session in the case of interruption.
3. **Content and delivery:** the system needs to ensure that session content is protected and delivered through a secure connection. Some solutions have been proposed; for example, content can be split into different parts and then distributed randomly through multiple paths to its destination. This will, however, increase the cost of performing the session.

Therefore, the following solutions are proposed:

1. Only users who have an account in the cloud can register and resume a session. Nevertheless, the sender has the authority to resume a session on another member's behalf. This will address the ownership aspect.
2. Once a session is registered in the cloud, two keys will be distributed. Each key has a unique hash value that is generated by the cloud. The first key is for the sender and the second for the receiver.

Hence, a user who is part of a session has to provide a key as well as a session ID to resume that session. These keys can be distributed using one of several well–known protocols [139]. As a result, the cloud will match the requester's ID, the key, and the session ID to ensure that, upon resumption, no rogue devices can access a session of which they were not part.

3. Regarding the session content, all data will be encrypted on the sender's side and decrypted on the receiver's. This ensures a high level of protection of session content. All communication will also be conducted through secure channels.

5.4 Implementation

This section presents the implementation of the previously mentioned model, providing more details about the sort of applications that were used and the types of data that were collected to evaluate this model, as well as the forms of communication that have been considered.

5.4.1 Cloud instance configurations and interactions

A Java Server Pages (JSP) page, hosted on the Amazon Elastic Compute Cloud (EC2) [140], was developed to deal with mobile users' requests. For instance, if a user sends a session registration request to the cloud, the request, including the generation of security keys, will be dealt with on this page.

Communication between mobile users and the cloud DB, which is an instance of the Amazon Relational Database Service (RDS) [141], is also dealt with through this page. Data, such as files or images, are stored using the Amazon Simple Storage Service (Amazon S3) [142].

Administrators can manage Amazon services in different ways, such as by accessing an AWS console. Alternatively, another Android app was developed (on a Samsung Tab 2) for administrator use for sending notifications, checking users' status and the retrieval of statistics about system usage.

5.4.2 Android app framework and interactions

An Android app was developed and installed on Android–based mobile devices (three Google Nexus Ones, two Samsung Galaxy IIIs, and one HTC One) to provide the MANET activities (neighbour discovery and routing protocol) and cloud operations (login, logout, and tracking membership). Some devices provide super–user access to allow Wi–Fi chip access and modification. Moreover, the mobile devices' databases were used to store users' information and active session details, as well as data checkpoints periodically during an active session. More details of the Android app can be found in Appendix B.

5.4.3 Mobile ad–hoc networking

Two ways of performing ad–hoc networking in Android mobile devices are used: first, the MANET app is used as a library to start and stop the MANET network [143]; second, a Wi–Fi configuration file is modified, as previously explained. The former method is used here for the sake of simplicity.

5.4.4 Testing applications

Three sorts of application were used to test the model validity:

Sending messages application

Users can send a message to neighbours either directly or using multi–hop. Users can also send messages to the cloud directly or using another peer's link. The time needed for a message to reach its destination is then calculated.

Sharing file application

Users can send a file to a peer either directly or via routing. The cloud can also be involved in sending files between two nodes that were connected to the same network but which then departed. The file can, for example, be a utilities map of a city that is shared by a rescue team in a disaster area (e.g., after a hurricane or earthquake). The MANET is set up and registered in the cloud before deploying it in the disaster area.

Transaction application

Another type of application provided here is for performing transactions such as executing queries on a database or downloading a file from a remote server using neighbouring links. A transaction was implemented to execute a number of database queries on the neighbour's side, as well as for retrieving textual information from a cloud database. This type of application is provided especially for evaluating the session management framework.

5.4.5 Communication types

The applications detailed above were run using different communication types and are presented below.

5.4.5.1 Communication between MANET members directly

When a destination is chosen, a Wi–Fi link is established using the Transmission Control Protocol (TCP) to ensure delivery. On the receiver side, a notification is triggered to notify the receiver about receiving new data. If the sent data contain a file or an image, it will be stored in the device's Secure Digital (SD) card.

5.4.5.2 Communication within MANET (multi–hop)

To achieve multi–hop communication, the sender is forced to route the request to one of its neighbours found in the neighbours list to reach the receiver — the routing table referred to earlier.

5.4.5.3 Communication to the cloud (directly)

- If a sharing file application is used, a multi–part http request is built containing file details to be uploaded to the cloud directly using a Wi–Fi or 3G/4G link. When the file arrives at the cloud, it will be stored in S3 space.
- If the messaging application is chosen, the message's text is sent as a direct http request to the cloud. When the cloud receives the message, an acknowledgement is sent to the sender.

- If the transaction application is used, the DB query is sent directly to the cloud and executed in its database. The cloud will return the results to the user when the query is successfully executed in the database.

5.4.5.4 Communication to the cloud via neighbours' links

If a user cannot reach the cloud directly, a neighbour's link can be used. Files/messages are sent directly to one of the neighbours found in the neighbour table, where the chosen neighbour acts as an intermediate node between the sender and the cloud to transfer the request.

This intermediate node will also send an acknowledgement to the sender when data are delivered to the cloud successfully. In this way, the sender will be able to compute the time resulting in sending these data.

In a real–world scenario, a request routed from a sender to the cloud may travel over more than one intermediate node. Therefore, the app forces the request to be redirected via neighbours to achieve one–hop, two–hop, three–hop, and four–hop communication to test the ability to serve requests from a sender to the cloud and vice versa.

5.4.5.5 Communication to a neighbour via the cloud

It is the nature of a MANET that, if a member leaves the network, messages can no longer be received. To benefit from introducing the cloud to this system, messages that are sent to a departing node can be redirected to the cloud to ensure delivery.

5.4.5.6 Receiving notifications from the cloud

Clients connect to a cloud to seek services. The cloud answers requests after storing them in the database. Sometimes, however, the cloud service needs to notify users by broadcasting an alert.

The cloud can push information to a group of devices without involving them in the operation. There are different ways to do this in practice, such as sending GET requests periodically to the cloud to pull messages.

However, this system uses another solution that is provided by Amazon called the Amazon Simple Notification Service (Amazon SNS) [144]. In the SNS, developers can send notifications (messages, email, or SMS) to endpoint users using one of the existing platforms [145] [146]. Since this implementation is deployed on Android–based devices, Google Cloud Messaging (GCM) for an Android platform is used [145].

When a user logs in successfully, the app registers on the GCM server and then sends an ID to the cloud. In the cloud, IDs are stored in the database, where an administrator can write a message and send it to users, either to a particular user or to all registered users. On the user side, notifications will be presented in the notification bar. By clicking on the notification bar, the content of the notification is expanded.

5.4.6 Test scenarios

Two main experimental scenarios were performed, as follows.

5.4.6.1 MANET general activity scenarios

- Communications between devices inside ad–hoc networks:
    - directly
    - multi–hops
- Communications between the cloud and mobile devices.

5.4.6.2 MANET session scenarios

- Session is completed without interruptions.
- Interruptions occur: session is interrupted before receiving the last chunk/part of the session's payload.
- Resumptions
    - By the sender:
        - the session is resumed from the beginning.

        - ∗ the session is resumed from the most recent checkpoint that was sent before the session was interrupted.
    - By a one–hop neighbour.
    - By the cloud, where the cloud will only send a notification to the sender when both parties become reconnected.

5.5 Experiment Results and Interpretation

This section presents the performance results of the MANET–cloud model implementation, including its infrastructural services. Each mobile device was placed away from the others but within radio range. Three main experiments were designed and run.

1. MANET *activity* experiment, including communications between devices inside ad–hoc networks, including direct and multi–hop communications and communications between the cloud and mobile devices.
2. MANET *sessions management* experiment, including cost calculations and discussions.
3. The third experiment was concerned with *security enhancements* when accessing the mobile cloud service in the cloud.

Each result represented the average of experiments having been conducted 10 times.

5.5.1 Experiment 1: Wi–Fi direct

This section describes an experiment that enabled users to create a Wi–Fi ad–hoc network using Wi–Fi direct technology.

5.5.1.1 Experimental setup

For the sake of simplicity, a desktop computer was set up as a cloud running a Java platform and holding a database. A JSP was coded to be used as an interface for the server's services.

Attached to this server was a MySQL database to hold users and MANET information, such as usernames, IPs, neighbours' information and so on.

5.5.1.2 Evaluation and experimental results

After installing the app on Android–based devices, two scenarios were performed:

1. Sending 10 http messages directly to the server.
2. Sending the same messages to the server over one of the neighbours' links.

In the second scenario, the app enables peer discovery. This means that any mobile device that has a Wi–Fi direct service enabled will be added to a list. If one of the peers in the list is chosen, a Wi–Fi communication link is set up using the Wi–Fi direct technique. When a successful connection is achieved, the requester sends the request to the server over the connected peer waiting for a response. In both scenarios, the delay time before the acknowledgment is received by the sender was calculated. Table 5.1 shows the round-trip times for sending several requests directly and through neighbours to the server.

Table 5.1: Experimental Results of Comparing the Average RTT for Sending 10 Requests to the Server Directly with Sending 10 Requests Through a Neighbour's Link Using "Wi–Fi Direct"

Request Type	Avg. RTT in Milliseconds
Directly	142
Through a peer	162

It is clear from the table above that sending a request to the server over a peer link will result in a slightly longer delay — 20 milliseconds, which is less than 15% — compared with the delay resulting from sending a request directly to the server.

However, the delay can be reduced or increased depending on the quality of the link between both users, as well as the quality of the link from the chosen peer (intermediate node) and the server. In other words, the better the peer's link, the smaller the difference between sending a request directly and through a one–hop neighbour.

5.5.1.3 Limitations

Using Wi–Fi direct to create an ad–hoc network is feasible and can allow users to reach the end server through neighbouring links. This technology is offered in most recent mobile devices and can be implemented in a straightforward manner. Link security and low–level connectivity jobs have been looked after, whereby a developer needs only to interact with this service in a high–level fashion, such as by starting peer discovery and sending a request.

However, using Wi–Fi direct requires many interactions from the users, such as accepting requests and allowing a connection with a peer, which break the pattern in using a neighbouring link (multi–hop) to reach the final destination (e.g., the cloud that hosts the mobile services). Moreover, there is a high possibility of a bottleneck occurring at the group owner node because all requests have to pass through this node (as in the Bluetooth master–slave framework).

Most importantly, using Wi–Fi direct makes it difficult to be involved in the management work, such as the IP allocation and routing protocol, which was the main goal for assigning the management task to the cloud.

5.5.1.4 Conclusions

As seen above, mobile devices can connect to the server successfully either using a direct link or using a mobile ad–hoc network (through an intermediate node). Each device has the ability to create an ad–hoc network to connect the server. On the other side, the server can receive requests successfully and store users' data in the database.

5.5.2 Experiment 2: MANET activity

5.5.2.1 Sharing file between MANET members

A small text file (1 KB) was transferred 10 times over TCP from the sender to the receiver. The round-trip time (RTT) was calculated at the sender side by comparing the system current time when the request was sent with the system current time when the acknowledgement was received.

Table 5.2 shows that the time needed to share a file directly between two nodes took around 2 seconds, which was the average RTT for sending 10 (1 KB sized) files from one device to another inside the same MANET network. It took almost double that time when the file was sent via another neighbour. This result is explained by the fact that a copy of the file is stored in the intermediate node. Interestingly, sending files over the cloud took almost the same amount of time as sending files by one-hop communication. The length of time will be affected by some factors, such as the availability of the destination node and the quality of the connection between the cloud and the mobile devices.

Table 5.2: Experimental Result of the Sharing Files Service

	Communications type	Avg. (RTT) in Seconds	Standard Deviation
To the cloud	Directly using Wi–Fi	0.447	0.43
	Directly using 3G	1.7111	0.47
	Via a neighbour's link	4.3647	0.43
To a peer	Directly	2.0508	0.01
	One–hop	4.2933	0.34

5.5.2.2 Sending messages between MANET members

A string containing "hello" was sent 10 times using TCP from source to destination. The delay in receiving this message at the destination was calculated on the sender side. Table 5.3 presents the results of this service using different communication types.

Table 5.3: Experimental Result of Sharing Messages Service

	Communications type	**Avg. delay (Milliseconds)**
To the cloud	Directly using Wi-Fi	260.3
	Directly using 3G	2825.2
	Via neighbour link (one–hop)	2701.6
	Via neighbour link (two–hop)	2776
	Via neighbour link (three–hop)	2883.2
	Via neighbour link (four–hop)	2921.1
To peer	Over the cloud	3864.4
	Directly	34.9
	One–hop	63.6

Table 5.3 indicates that the time needed to send a message directly to the cloud using a 3G link was almost the same as the time taken to send a message to the cloud via neighbours. Furthermore, the delay might be less if another neighbour has a better link. There are two main reasons for this result: firstly, the delay time added by communicating inside the MANET is very small compared to the delay that results from communicating with the cloud; secondly, the quality of links varies, which means some nodes have better links than others. This means that communicating with the cloud via a neighbour that has a better quality link will reduce the overall delay time. Furthermore, the delay time can be less than when sending the same request over a low–quality link.

However, sending messages between two peers via the cloud takes around 4 seconds, because using the aforementioned notification system (SNS with GCM) adds an extra delay to the request, whereby the request travels from the sender to the cloud and then to the GCM platform before reaching the destination node. The acknowledgement will also travel all the way back to notify the sender.

Table 5.4: Experimental Results of Comparing the 99% CI Values of Sending 10 Messages Directly to The Cloud and Sending the Same Messages via a Neighbour's Link

Request Type	**Confidence Bounds**	**Values** in Milliseconds
Directly	Lower Confidence Limit	2212.0
	Mean	2825.2
	Upper Confidence Limit	3438.4
Through a peer	Lower Confidence Limit	2271.5
	Mean	2701.6
	Upper Confidence Limit	3131.7

For better analysis of the data, the upper and lower bounds of the confidence interval (CI) are collected here. Using this statistical method allows a determination of whether the mean of the collected data falls within this range. Comparing two sets of data using their CI range also increases the accuracy level, in comparison with using only the mean value. Using this method can also help to predict where unknown data that might be collected after running further similar experiments are likely to fall. For further reading on CI, see references [147] and [148].

According to Table 5.4, the results revealed 99% confidence that the mean delay resulting from sending messages to the cloud directly and the delay that results from using a neighbour's link are very similar. Furthermore, the delay could be lower if that neighbour has a better link to the cloud.

5.5.3 Experiment 3: MANET session

Time and energy were calculated in both the case of an interruption using no checkpoint and when the checkpoint models were used. The aim was to compare, in terms of energy and delay costs, the no checkpointing case against checkpointing cases in which a session is resumed from the most recent checkpoint. This would show the overhead cost of resuming a session, as well as the cost when no checkpoint was used.

There are also other factors that can be reported here, such as the amount of storage needed and network throughput cost. For the former, the proposed checkpoint technique stores only a minimal amount of information about the session and, as a result, the memory usage is very low and will not affect the application performance.

A session might have some payload or media, such as an image or text file, but these data are already stored in the mobile devices and there is no need to store them again, in which case only their URL or path will be recorded/stored by the checkpoint.

The checkpoint technique could also reduce mobile storage by uploading a session's payload to the cloud and deleting it from the mobile device, as in the case of an intensive checkpointing framework. Network throughput is potentially an interesting factor and further discussion is provided in Section 5.5.3.5.

This experiment used a power profile called Trepn [149] to capture energy consumption. A Trepn Profiler is a diagnostic tool that allows developers to profile the performance and energy consumption of Android applications. The experiments were run on four devices: three Samsung Galaxy SIII smartphones and one HTC One smart device. The devices were located within their radio frequency range.

5.5.3.1 Checkpoint models

The main purpose of the experiment was to determine the running cost of different checkpoint models. All the aforementioned checkpoint models (intensive, local, normal and none) were tested, and the experiment was run five times for each model.

Figure 5.15 shows that using an intensive checkpoint has the highest cost both in terms of time and power because each time a new checkpoint is created, it is uploaded to the cloud at the same time. The normal model was less costly but still more costly than the local and no checkpoint models. However, using a local checkpoint resulted in a cost comparable with that of not using a checkpoint.

For example, completing a file–sharing session using intensive checkpoints resulted in around a 70% increase in cost in terms of power versus completing the session without creating checkpoints. Here, repeating the session from the beginning would be better than using the intensive model, particularly if the possibility of a break occurring was very low. Using intensive and normal modes add an unnecessary overhead to the system if the session is short or its completion is not essential.

However, not using these types of checkpointing would be counterproductive if the session were sensitive, requiring a long computational process, such as joining multi–queries in the case of a DB transaction, or the link was not stable and there was a higher chance that a break could occur.

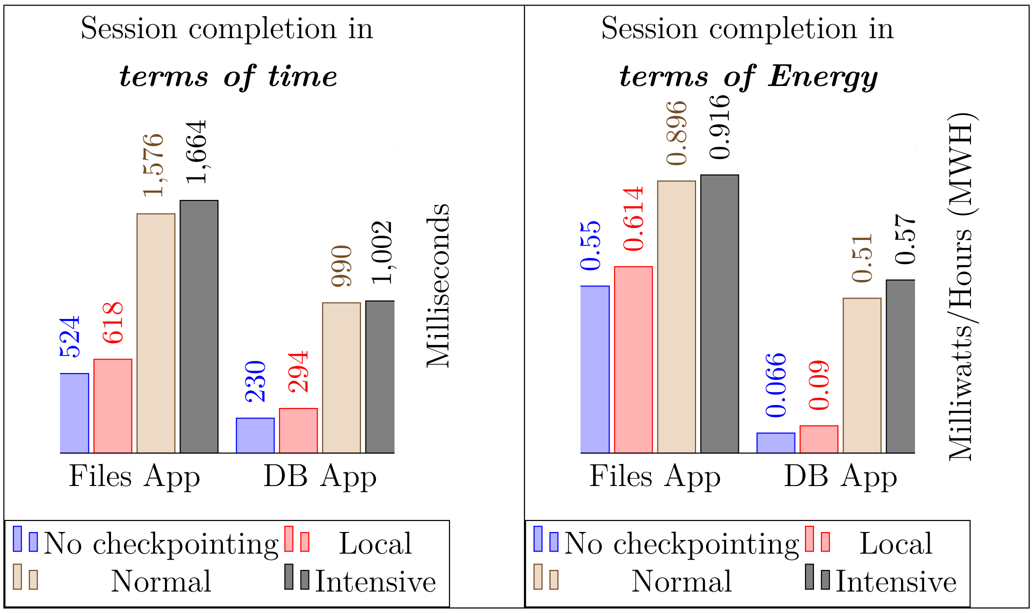


Figure 5.15: Checkpoint Models' Costs in Terms of Energy and Time

However, values collected from mobile devices are sometimes dependent on factors such as the operation context (such as other apps running concurrently and battery level, leading to scaling methods). Therefore, Figure 5.16 and Figure 5.17 detail the 95% confidence interval (CI) for the results of completing a session (file sharing and DB transactions) both in terms of power and time, giving the medial number of each data set. The figures also show the upper and lower band values of the CI.

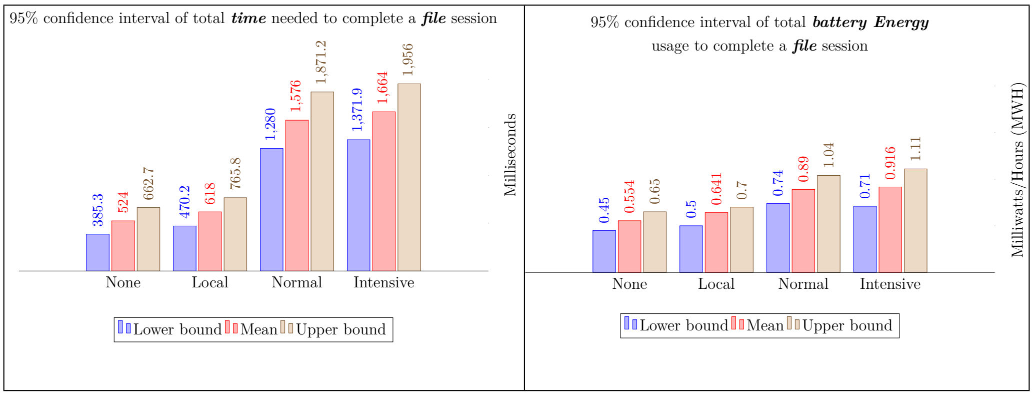


Figure 5.16: CI of Completing a File Session in Terms of Energy and Time

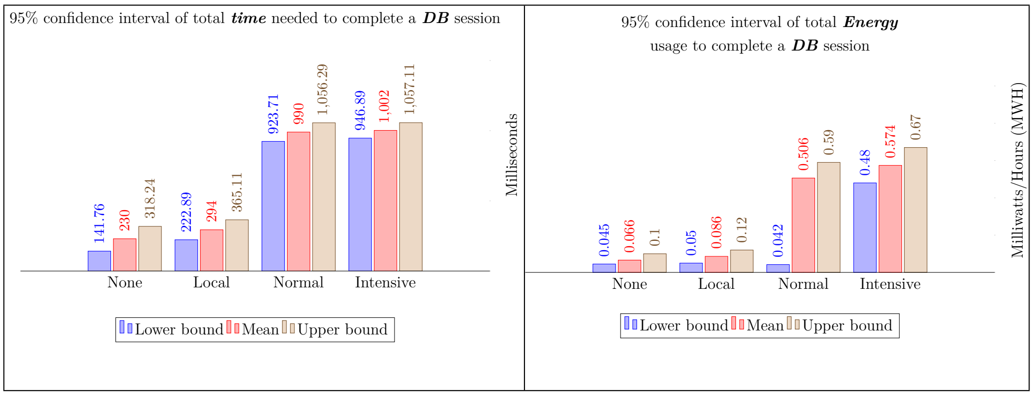


Figure 5.17: CI of Completing a Database session in Terms of Energy and Time

5.5.3.2 The cost of using checkpoints

The checkpoint cost C_c was calculated using the following equation:

$$C_c = C_p - C_n \tag{5.1}$$

Where C_n is the cost of completing a session without interruption and C_p is the cost of completing a session using a checkpoint technique. This includes the total overhead.

Sharing file results

First, the shared file is split into several equal parts. Afterwards, when the session is registered and acknowledged by the corresponding user, the sender starts sending parts of the file. Each time a part is received, the next part is sent and a checkpoint is created locally in the DBs of both mobile devices. This checkpoint is only uploaded to the cloud if an intensive checkpoint model is chosen.

When all parts have been sent and received successfully, the cloud will be notified about the termination of the session. The receiver merges all the file parts to obtain the original file.

Looking at the left–hand side of Figure 5.15, it is clear that using local checkpoints results in a slightly higher cost in terms of time and energy compared with no checkpoint: 13–18% and 6–14%, respectively, higher than performing the session without creating any checkpoints. However, this extra cost adds strength to the app; it guarantees that progress is saved and what has already been performed will not have to be repeated.

Executing database transactions results

Three different database queries — create, insert, and union — were sent from one device to another within the same MANET. The right–hand side of Figure 5.15 shows that creating checkpoints locally will result in a slightly higher cost compared with the cost resulting from executing the session normally.

Using the normal and intensive models will result in a higher cost compared with both the no checkpointing and local models. These models are recommended in the case of an unstable link between two nodes, or if the session has a large amount of data.

5.5.3.3 Session resumption overheads

Locally (inside MANETs)

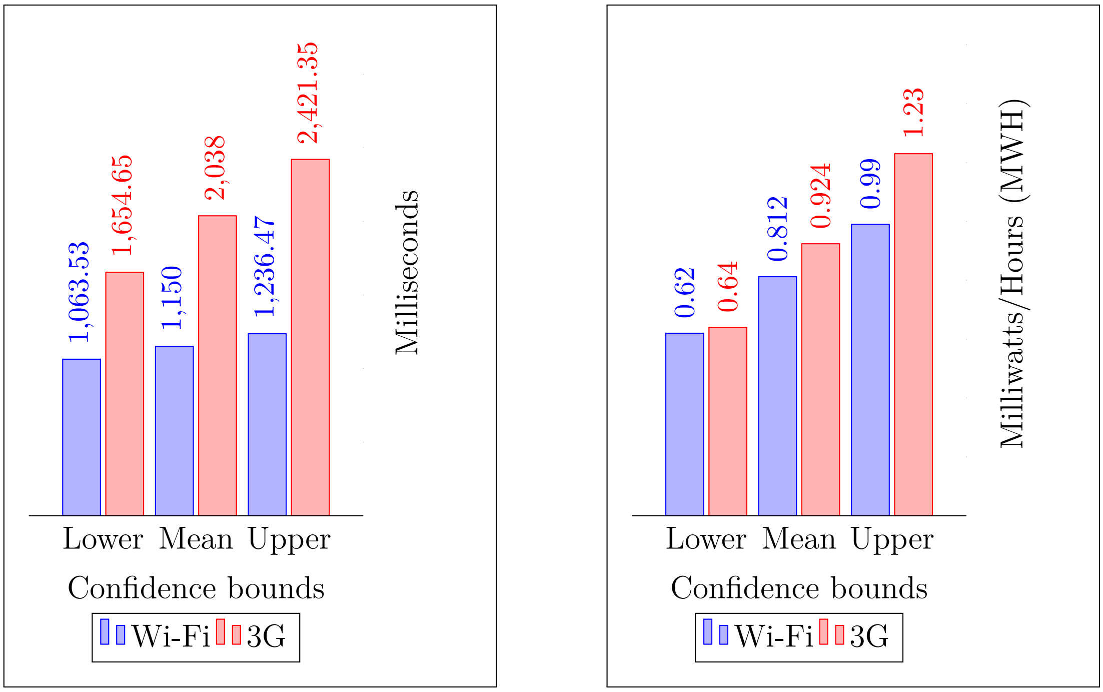
This experiment reported the overheads associated with resuming a session. When a session is interrupted and a checkpoint technique has been used, the session can be resumed from the latest checkpoint.

The type of application that was used in this session had no impact on this process. However, before resuming a session, the following actions were performed:

- Retrieving the session details, including the most recent checkpoint ID.
- Retrieving the most recent checkpoint to identify what had been sent and received before the session break.
- Checking if the target user who was involved in the session is active and reachable.

According to the experimental results from running the experiment 10 times, the average delay time resulting from preparing a paused session for resumption was around 30 milliseconds. This amount of time is reasonable and ensures that the session will not restart from scratch.

Over the cloud



(a) Overheads in Terms of Time

(b) Overheads in Terms of Energy

Figure 5.18: 95% confidence intervals of overheads of resuming 5 paused sessions by the cloud in terms of time and energy

Resuming a suspended session can be done by the cloud, as mentioned earlier. It was found that sending a resume request to the cloud took around 2 seconds and consumed around 1 milliwatt per hour.

These results were higher than for the resumption of overheads locally. As devices can only run in one mode, either ad–hoc or normal, a cellular interface is required to reach the cloud. However, if users can reach the cloud using a Wi–Fi connection while connecting to an active MANET, this will result in around half of that delay with less energy usage. Figure 5.18 presents the overheads incurred when 5 paused sessions were resumed using the cloud services, with 95% confidence intervals.

By a peer

This experiment showed the cost of resuming an interrupted session via a neighbour who is one hop away from both users (sender and receiver). The cost of resuming a session via a peer C_{rop} was calculated using Equation 5.2:

$$C_{rop} = C_{ip} + C_{cpp} \tag{5.2}$$

Where C_{ip} is the cost resulting from an interrupted session that used checkpoints and C_{cpp} is the cost of completing a session from the most recent checkpoint through a peer. The OLSR routing protocol was used, thus an intermediate peer was chosen from the routing table.

Experimental results from resuming 10 paused sessions showed that the difference between the average cost of resuming an interrupted session through another neighbour (758 milliseconds) and the average cost of resuming a session without using a checkpoint technique (794 milliseconds) is very small (less than 40 milliseconds). Furthermore, the results showed that the cost of resuming a session via a peer one hop away from the sender and the receiver consumed around 30% more energy than that consumed by repeating the session from the beginning when the break in the link occurred.

However, since the receiver might no longer be reachable, the extra cost that this entails should be taken into account when considering the peer resumption method as a possible solution. Furthermore, each time a sender fails to reach the receiver, the total cost will be increased. As such, resuming a session from the beginning may result in higher costs than resuming a session through another peer.

5.5.3.4 Energy consumption

For the file–sharing application, the total amount of battery energy used to resume a session using a checkpoint technique was compared with the energy used to resume a session without checkpoints. The cost of resumption was calculated using Equation 5.3:

If C_{cn} is the cost of completing the session after a break occurs and C_{in} is the cost resulting from interrupting the session before the last part is received, the cost of resuming the session,C_{rn}, will be:

$$C_{rn} = C_{cn} + C_{in} \tag{5.3}$$

If no checkpoint is used, the session will start from scratch, which means the value of C_{cn} will be equal to the cost resulting from performing the session without incident.

In comparison, the cost of resuming a session when checkpoints are used, C_{rp} , will be the sum of C_{ip} , which is the cost resulting from an interrupted session that used checkpoints and C_{cp} , which is the cost of completing a session from the most recent checkpoint (see Equation 5.4).

$$C_{rp} = C_{ip} + C_{cp} \tag{5.4}$$

Figure 5.19 compares the cost resulting from resuming a session when a checkpoint technique was used with one when it was not. The left–hand side of this chart shows that using a checkpoint technique can save up to 37% more energy than an interrupted session that did not create checkpoints.

For the database application, the cost resulting from resuming a transaction session was calculated using Equations 5.3 and 5.4. The right–hand side of Figure 5.19 indicates that using our scheme could save up to 36% of the cost resulting from a session that is interrupted and in which no checkpoints were created.

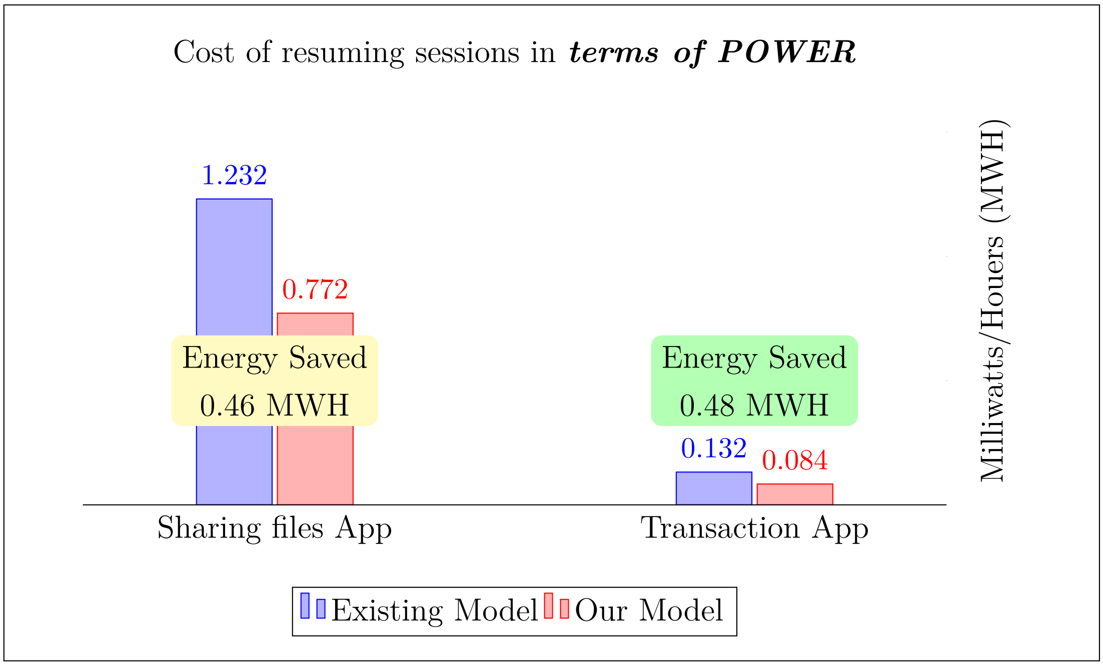


Figure 5.19: Cost of Resuming Sessions in Terms of Energy

5.5.3.5 Network cost

There are two main types of network cost. The first is network throughput cost, which will depend on the decision an application makes about whether it should send a task to the cloud for execution or whether the task is better executed using the resources of the mobile device. This is one of the recognized research directions of the mobile cloud computing domain and could also be an offloading issue.

It is a complex decision for an application to decide whether to back off and wait to connect to the cloud another time in order to improve network throughput or send a request. See Barbera et al. for further reading on mobile cloud offloading check [150].

The second type of network cost is the actual cost or price that results from reaching the cloud, because checkpointing might cause a large transfer of data over a 3G/4G mobile network, implying potential additional financial costs for the data transfer.

Therefore, users can choose to operate checkpointing in either Wi–Fi or MANET mode (e.g., via a neighbouring link) in order to avoid extra costs resulting from a mobile network. As a result, checkpoints created by uploading will be postponed if no Wi–Fi is detected.

5.5.4 Experiment 4: Security enhancement

5.5.4.1 Discussion

Once each user had provided valid credentials, all of their activities needed to be linked to their account in the cloud, regardless of which mobile devices were used. For this reason, the way users access their account needed to be enhanced in order to ensure a high level of protection.

Therefore, in this section, the authentication process presented previously was enhanced to deliver a higher level of protection, as well as to allow users to access their account in the cloud more confidently. In practice, there are many ways in which authentication is proposed for mobile device access. These proposals are classified into three main methods, depending on their fundamental mechanisms [151], as follows:

Something you know: such as using a personal identification number or password, which is what was used in the initial design. This approach is considered to be a standard level of protection and is the most common authentication mechanism.

Something you have: such as a token or subscriber identity module (SIM), although this method suffers from the limitations that the means of authentication can be lost, stolen, or misplaced.

Something you are: such as biometric characteristics, which are defined as being uniquely individual, non–transferable to others, impossible to forget or lose, difficult to reproduce or falsify, usable with or without the knowledge/consent of the individual, and difficult to change or hide.

A two–way authentication method using SMS messages was chosen, which is a type of biometric authentication method. In effect, when a user provides valid credentials (e.g., username and password), the cloud sends an SMS message that contains a unique code that expires after a certain amount of time (1 min, for example) to that user via a pre–stored phone number. If the user replies with a correct and valid code, the user is then allowed to access the cloud services.

Figure 5.20 illustrates the enhancement of authentication to access the cloud services. The implementation of this method is presented in the following section.

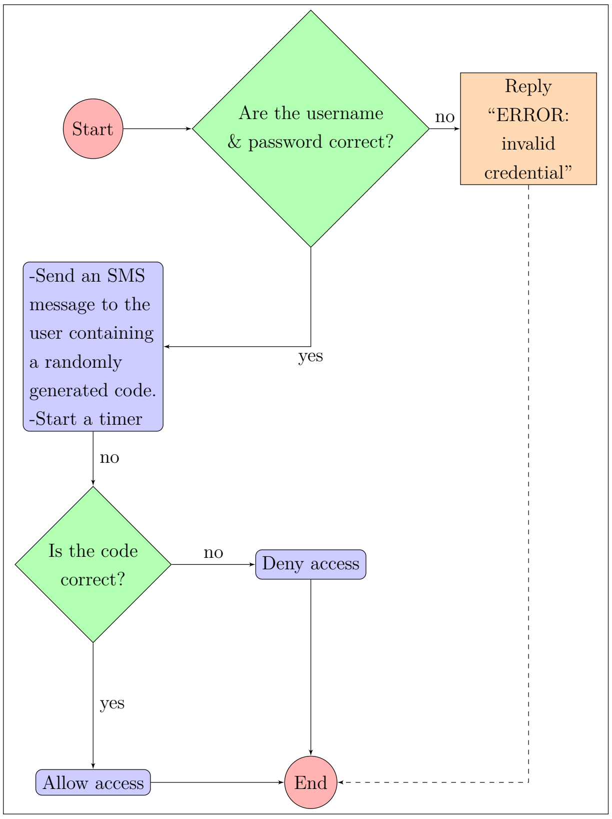

Start
Are the username & password correct?
no
Reply "ERROR: invalid credential"
yes
-Send an SMS message to the user containing a randomly generated code.
-Start a timer
no
Is the code correct?
no
Deny access
yes
Allow access
End

Figure 5.20: Flowchart of the Enhancement to the Login Process

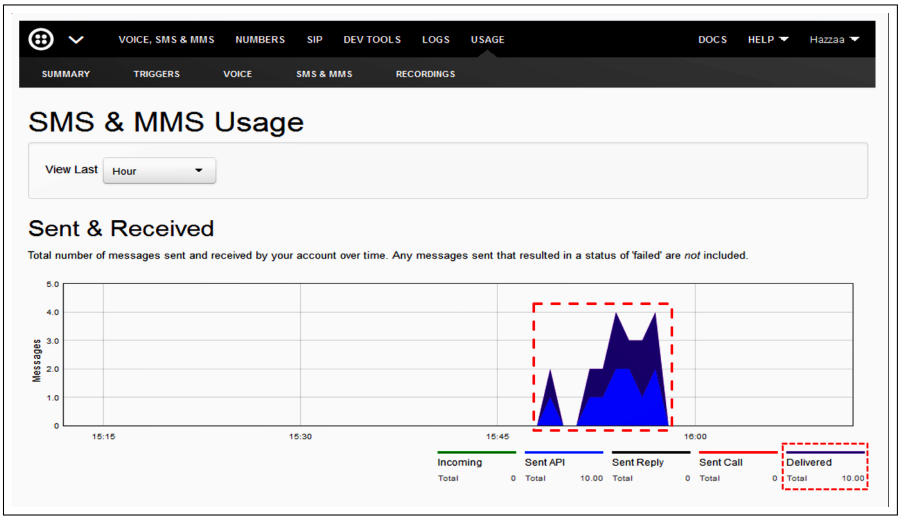
5.5.4.2 SMS authentication

The Amazon cloud only offers an SMS gateway to US phone numbers, so this experiment was implemented using the Twilio platform [152], which is a web service API that allows the sending and receiving of SMS messages between users and the cloud.

In other words, when the cloud receives a login request from a user with a valid username and password, the cloud sends a one–time unique generated code via the Twilio platform as an SMS message. If the user receives this message, it will re–type the code to send it to the cloud. The cloud finally validates the code to either allow or deny access to this user.

To test the idea, 10 attempts at the login process were made using the aforementioned authentication method. The results were then collected in terms of the time delays with respect to waiting for an SMS message to arrive after providing the correct username and password. Figure 5.21 is taken from the Twilio API website and shows the number of sent and delivered messages.

Figure 5.21 illustrates that all the SMS messages sent were delivered to the end user. Each message had a unique code that is saved in the cloud database under the user's profile for 1 minute. A full login process was performed using SMS authentication and it was found that the idea was feasible and that the average delay time was 18–19 seconds, which is within an acceptable range.



SMS & MMS Usage

View Last: Hour

Sent & Received

Total number of messages sent and received by your account over time. Any messages sent that resulted in a status of 'failed' are *not* included.

Incoming	Sent API	Sent Reply	Sent Call	Delivered
Total 0	Total 10.00	Total 0	Total 0	Total 10.00

Figure 5.21: Screenshot from the SMS Gateway (Twilio) Presenting the Number of Sent and Delivered Messages

5.6 Evaluation

In this section, an integrating perspective of the system is presented and discussed in respect to some performance requirements, such as scalability. The following subsections present these requirements and discuss the above design to assess how it can fit and how it can be extended.

5.6.1 Availability and reliability

The main goal of introducing a MANET–cloud model is to enhance the availability of the whole system. In other words, if a user in an emergency is looking for help but does not have a direct connection to the cloud, this user can benefit from the proposed model by sending a request to the cloud through a neighbouring link. Moreover, if this user has a poor–quality connection to the cloud whereby one of his/her neighbours might have a better–quality connection, using the latter user's link to communicate with the cloud can minimize the delay that would otherwise result from serving the requesting user by the cloud.

According to experimental results, the mean delay resulting from sending messages to the cloud directly and the delay resulting from using a neighbour's link are very similar. Furthermore, the delay could be lower if that neighbour has a better link to the cloud.

Another factor this model is trying to enhance is the reliability of the system. This can be achieved by saving the progress of a session that is started between two MANET members and a break in the link occurs. In this situation, the user who started the session can save the session to resume it when both users reconnect.

5.6.2 Scalability and efficiency

After designing and implementing the MANET–cloud model, a question can be posed: What if the cloud needs to manage a high number of MANETs (e.g., thousands of MANETs)? In that case, the extent of the cloud resources (e.g., storage) allocated to manage each MANET has to be considered, as well as an estimation of how the cloud can manage a higher number based on this calculation.

Assume that the value of the resources for managing a basic MANET that consists of two nodes is V and the type of resource is S. Therefore, the amount of storage needed to manage a MANET network that has two mobile nodes is Vs. Then assume that the storage needed to accomplish an operation to join a user to a MANET that is already managed by the cloud is Sj.

Hence, estimating the storage needed to manage a MANET that has 1,000 members can be achieved using Equation 5.5 :

$$\text{Total Storage} = Vs + (1000 \times Sj) \tag{5.5}$$

However, an estimation can be considered to determine how much managing 1,000 basic MANET networks can consume using Equation 5.6:

$$\text{Total Storage} = Vs \times 1000 \tag{5.6}$$

According to the above experiments, the average storage needed to create basic MANET network that has two members is 744 K.

Here, managing 1000 MANETs by the cloud will consumed around 744000 kilobyte (around 744 megabyte) of the cloud resources.

Therefore, from this calculation and estimation, the system administrator can consider the amount of resources needed as well as how efficient this model is from the point of view of saving or at least minimizing cloud resources.

5.7 Chapter Summary

This chapter has presented a new infrastructure service for the middleware system proposed in the previous chapter which aimed at allowing users to create and join mobile ad–hoc networks that are managed by the cloud. The design and implementation of the system was discussed. The most important benefits of this solution are its impact on MANET operations such as split, merge, join and leave. The model is reliable and robust, as split, for example, does not require the reallocation of IP and SSID addresses, as they are managed by the cloud. All local communication between mobile devices for split management becomes unnecessary and, therefore, saves mobile resources. The same applies to merge, join and leave operations.

The experimental results showed that the proposed model is feasible and produces excellent results under laboratory conditions. The experimental results indicate that the delay involved in sending messages to the cloud directly and the delay resulting from using a neighbour's link are very similar. Furthermore, the delay could be lower if that neighbour has a better link to the cloud.

With regard to MANET sessions, the loss of work and resources in a MANET when sessions are broken is a serious problem. This chapter demonstrated, also empirically, that saving session details in the cloud will provide robustness to the network, as a session can be resumed as soon as the disconnected node(s) reconnect. Furthermore, if the corresponding nodes are connected to different MANETs, the session can still be completed by the cloud, as the cloud works as a bridge to finish the session.

The use of the cloud service does not affect the ad–hoc nature of a MANET. On the contrary, it supports the session completion in the context of users' mobility and mobile devices' scarce resources. According to the experimental results, the energy consumed by resuming a session locally is around one–third lower than when a session is interrupted and no checkpoint is used.

Using a checkpoint technique will add 6—14% extra cost (in terms of energy) to the session but will ensure the session will not restart from scratch. In addition, an interrupted session can be resumed if its users cannot communicate directly. Even though this will consume more power, it can be considered a feasible solution. The results could be further improved if a session takes a long time due to large data transfers.

In summary, managing a mobile ad–hoc network of Android–based devices in the cloud is achievable and offers features such as reliability and robustness. Furthermore, a sessions' management model was presented and tested which can be used as a guide for best practice regarding session management in a MANET. Applications history or current network parameters can be used as input into the decision component to decide before a session starts whether checkpoints are worth using.

The following chapter presents two cloud services that are aimed at helping people on the move who encounter an emergency. These two services make the best use of the MANET–cloud model and the framework presented here to deal with breaks in the connectivity of active sessions in order to enhance the reliability of these mobile cloud services.

Chapter 6

Emergencies Help Facilitated by the Mobile Cloud

6.1 Introduction

As part of the proposed middleware system, this chapter introduces two mobile cloud services that provide healthcare to people who are in an emergency situation but out of reach of home or office, for example in a crowded area such as an Al–Hajj event. These services are hosted in the cloud and can be accessed via mobile devices in which the MANET–cloud model is also used. In other words, these two services are set on top of the MANET–cloud model in order to make the best use of features such as being able to use MANET networks when the regular network, such as a cellular network, is not available.

Here, users can create/join a MANET network to seek help. On the other side, rescuers can use this type of network to look for missing people or victims in the case of an emergency.

The scenarios previously presented in this research, such as Al–Hajj or driving on a highway, were taken into account in the design of these two services to ensure that they can provide the types of help that are needed in these situations. Figure 6.1 shows how these two services interact with the MANET–cloud model.

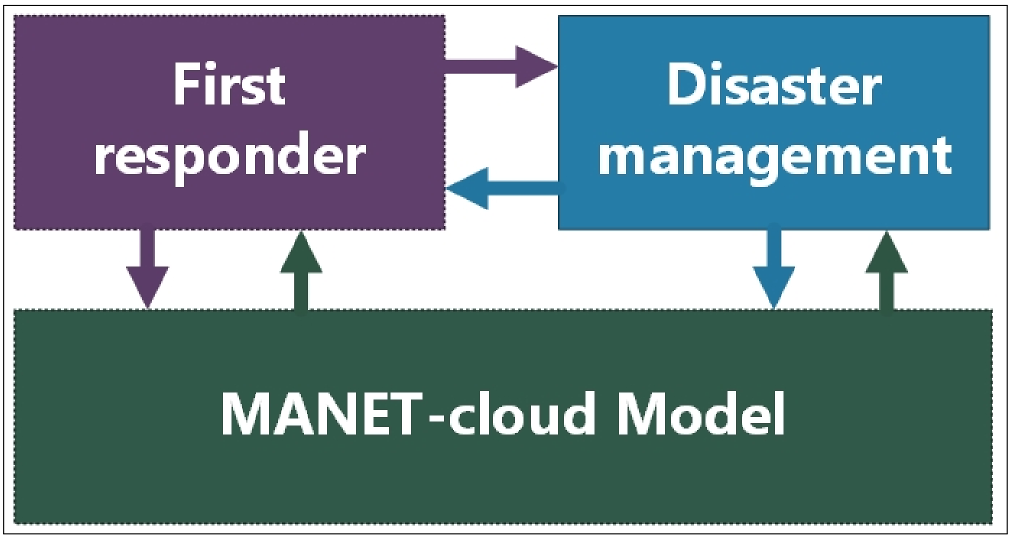


Figure 6.1: Services and Infrastructure Layers of the Proposed Middleware System

The first service consists of directories of medical practitioners, such as doctors and nurses, and medical organizations, such as emergency departments and medical centres, all of whom volunteer to provide a first responder service. This service is based on availability and location, and aims at helping people in need as quickly as possible.

The second service processes live Twitter streams, in order to detect emergency calls and notify end users. End users can be mobile, such as someone who wants to check the status of an event in his/her town, for example in the case of sudden flooding.

Alternatively, an emergency department could look for information on social media to assist in taking decisions or actions regarding an event that has occurred. Furthermore, users can use social media either to reach the cloud to seek help or provide valuable information to others regarding an event.

The following two sections present each service separately, starting with the main goal of each service, then providing detailed information about the design and implementation of this service, followed by the method used to test each service. Finally, the experimental results are discussed and provided as the evolutionary part of this service.

6.2 Service 1: First Responder

Emergency cases require the swiftest possible response from an appropriate medical service if they are not to become life–threatening. In the medical emergency field, response times to emergency cases are a major concern and receive a high degree of attention.

Many of the systems proposed in the literature are intended either to replace an existing emergency system with a fully automated one, or build on unreliable or less efficient frameworks that are based on some sort of social media application, such as redirecting emergency requests to Facebook friends.

This section introduces the first mobile cloud service that works side by side with an existing emergency system and is aimed at reducing the time spent waiting for emergency help to arrive, as well as making the best use of medical professionals who may be in close proximity to the medical case.

6.2.1 Objectives

The main objective was to design a mobile cloud service that creates a directory of trusted and qualified medical professionals, as well as tracking their location and availability. As a result, the service will:

- Allow users to search the directory to facilitate a swift response to the requester's medical needs and reduce the total waiting time.
- Make the best use of medical professionals registered in the system who may be in a particular location, such as in a crowded area, and could assist the traditional emergency services (e.g., ambulance) by arriving on the scene more quickly and providing medical help until the emergency personnel arrive.

6.2.2 Design and implementation

This section presents the service design and implementation. A high–level overview of the service model is presented in Figure 6.2 and consists of three main components:

1. **The cloud**, which plays the most important role by hosting services.

2. **The users/potential patients** who might experience emergencies.
3. **Medical professionals**, such as doctors and nurses.

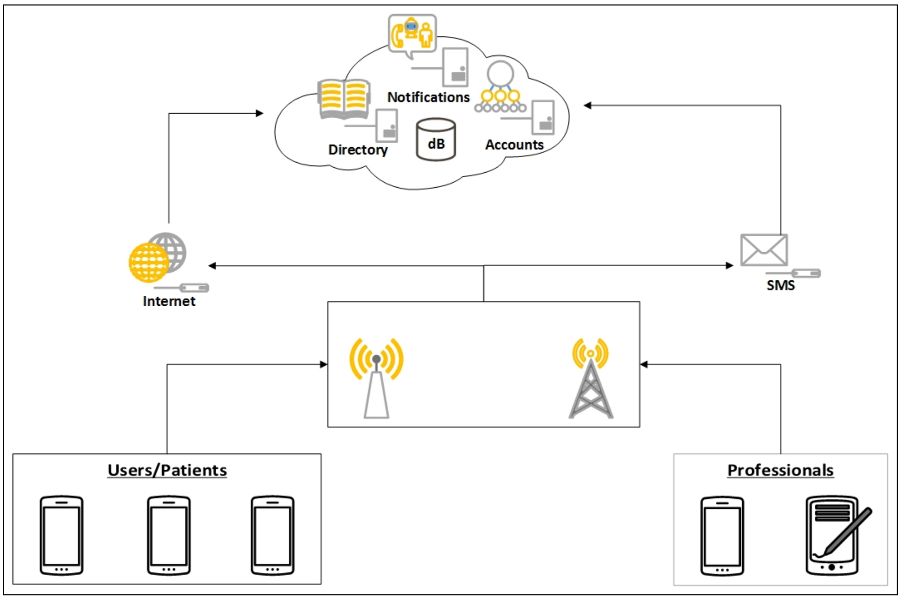


Figure 6.2: High–Level Overview of the Proposed Model

The initial step is the registration of doctors, nurses and medical organizations with the cloud service. Only after that has happened is the service ready for users to send requests for a professional to deal with the medical emergency or concern that he/she has.

The central element of the cloud service is a directory that lists all the available volunteer medical professionals and allows users to alert those professionals who match their medical needs. Those who are able to provide this mode of medical care have to register their details, including personal information (name, gender, etc.), employment (ID, occupation, etc.) and contact (email and telephone number) details in the directory through their medical organization before they begin providing a service (more details are provided in the section below on security) and entries are organized according to their role (doctor, nurse, etc.).

In addition to the details that are provided in the registration operation, this service will track each professional's availability in terms of whether he/she is connected, disconnected, busy or free and current location. For reasons of fairness, the cloud will ensure that all professionals are treated equally based on the total number of cases assigned. These details will play a major role in the 'look–up' operation, which is presented in the following section.

6.2.2.1 Look–up operation

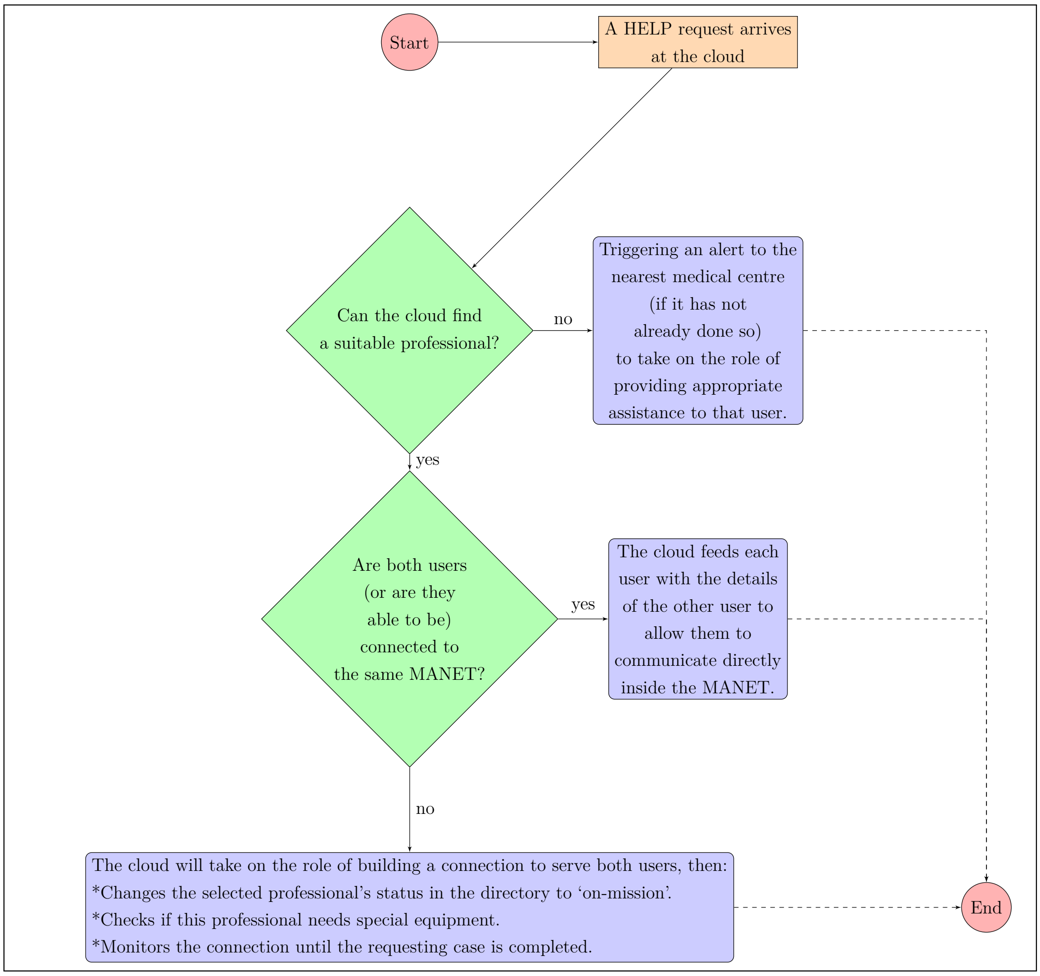

Start
A HELP request arrives at the cloud
Can the cloud find a suitable professional?
no
Triggering an alert to the nearest medical centre (if it has not already done so) to take on the role of providing appropriate assistance to that user.
yes
Are both users (or are they able to be) connected to the same MANET?
yes
The cloud feeds each user with the details of the other user to allow them to communicate directly inside the MANET.
no
The cloud will take on the role of building a connection to serve both users, then:
*Changes the selected professional's status in the directory to 'on-mission'.
*Checks if this professional needs special equipment.
*Monitors the connection until the requesting case is completed.
End

Figure 6.3: Responses from the Cloud to a Help Request

This operation is executed when a user sends a HELP request to the cloud to look for a medical professional to provide assistance. The request is redirected to the directory service to select the professional who most closely matches the needs of the patient making the request. Four attributes are considered: status, availability, medical speciality and current location.

Thus, only a professional who is both actively receiving jobs from the cloud and is available, meaning that he/she is not treating another patient and is able to deal with the emergency case received, will be considered. Finally, he/she will be selected according to his/her proximity to the location of the emergency case.

However, emergency services, such as ambulances, are also called when the look–up procedure starts. Figure 6.3 demonstrates the responses from the cloud to a HELP request.

Depending on the medical situation of the requesting user, another professional might be required to attend the emergency location; for example, a doctor with another speciality. A request is then sent to the cloud to notify the other professional. This kind of service is offered only to professionals who are on call. A communication link is then set up between the two professionals. Directions to the emergency location are also provided if required.

6.2.2.2 One–to–one communication protocol

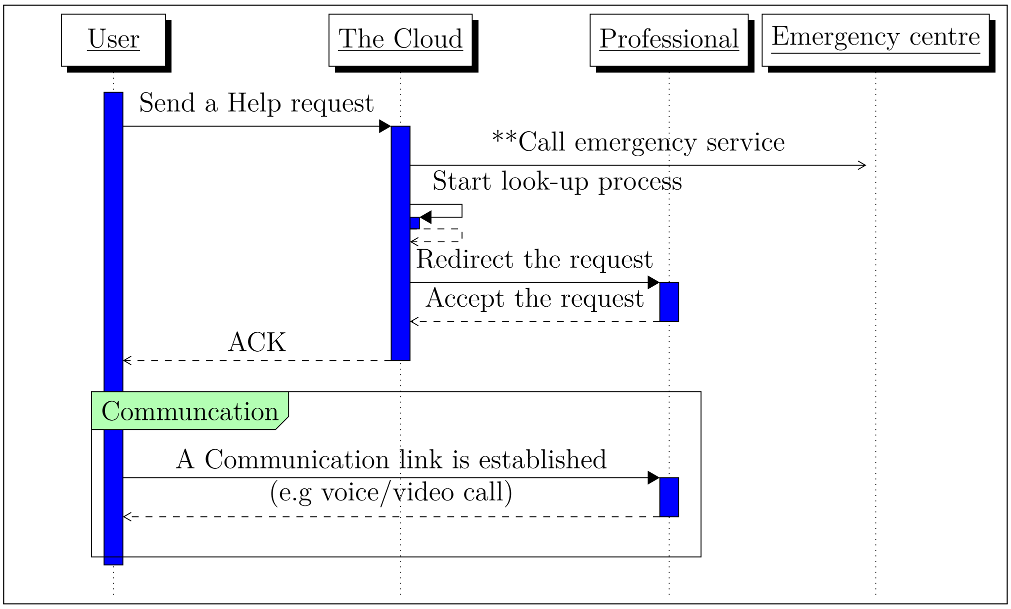


Figure 6.4: Sequence Diagram for Establishing a Link Between a Requesting User and a Selected Professional

Once a professional is selected to deal with the requesting case, a communication link is established between the two users. The cloud is responsible for monitoring this link, as well as for providing any support to the professional such as notifying other practitioners. Figure 6.4 shows a sequence diagram for establishing a link between a requesting user and a professional.

6.2.2.3 Help requests prototype

A user can use a mobile phone to send a HELP request to the cloud to look for a professional to discuss a medical concern, with the ability to submit real–time information about this medical concern such as answering questions that will help with diagnosis.

However, a user might sometimes not be able, or not have time, to provide details of an emergency case but will at least be able to use a mobile device to request help. Similarly, someone could help another person who is in an emergency situation by requesting medical help on that person's behalf. Furthermore, a user might not be able to access conventional medical services at all, with nobody available to help him/her request appropriate help. Therefore, three ways of asking for medical help are provided by the cloud depending on the status of the emergency case. Furthermore, all these forms of help can be sent either over the Internet or via SMS because it might be the case that users do not have an Internet connection but have the ability to use a cellular network to make calls and send SMSs.

1. Completing a form

Any user who has an active account in the cloud can send a HELP request to look up a professional to discuss a medical case or to help another person who needs medical assistance (e.g., an injured child). This request has to include information explaining the user's medical situation, such as a keyword (e.g., "heart", "breathing" or "fainted"), current location, and connectivity status to help the cloud select the most suitable professional from the directory service.

2. Clicking a button

Another way of requesting help is by clicking a button that will trigger an alert to the cloud to help the user as soon as possible, taking into account that the requester is at high risk and thereby according this request a higher priority. As a result, the cloud will redirect this request to an available professional and emergency department to seek a swift and appropriate outcome.

However, a user who intends to request this sort of help has to have pre–defined any allergies or special medical requirements. To ensure best practice, a simple EMR is provided in the cloud that is linked to each user account. Each record includes a collection of electronic health information about each individual patient, such as medical history, medication, and allergies.

The cloud also allows professionals to view/edit these EMRs if required.

3. Detecting a medical case

The third way of seeking medical help is through wearable medical devices. Alternatively, detecting emergency cases can be done using out–of–hospital schemes. Both types have been given a high level of attention in the literature.

First, a user can link a wearable device to his/her account in the cloud, which results in the cloud starting to monitor this device to detect an emergency situation without further input from the user. Then, if an emergency case is smartly detected (e.g., abnormal heart activity), a HELP request is automatically sent from this wearable device through a mobile device (or the Internet if the device has the ability to connect to the Internet directly) to the cloud to facilitate a look–up process and, at the same time, sends the user's information to an emergency department.

6.2.2.4 Security mechanisms

This service does not deal directly with the privacy and security issues involved in implementing cloud computing in the health field. However, some enhancements are provided to the proposed service, as shown in the following subsections, to ensure a reasonable level of security.

Trustworthiness

To guarantee that the service is trustworthy and that only medically qualified people are added to the directory, the cloud sends each professional's details (e.g., name, registration ID and occupation) to the health authority (e.g., the Ministry of Health in Saudi Arabia) to verify that the person in the process of registration is qualified to provide healthcare services to the public.

If the medical professional is validated, a new account is created and a message containing the result of this operation is sent to that professional. In addition, the details are added to the directory of professionals.

Privacy

Dealing with health data in cloud–based systems (e.g., storing, sharing and accessing) is one of the open issues in the integration of cloud computing into the healthcare sector [153]. A number of ideas have been proposed with the aim of providing high–level privacy for medical data, as well as ensuring the high–level protection of these data, such as using a private cloud [154] to provide a high standard of protection. Others have used a hybrid cloud type [155], in which data are stored in a public or private cloud depending on how sensitive the data are.

In other words, the more sensitive the data, the more likely it is they will be processed and stored in a private cloud. With this in mind, the following actions are carried out to protect users' privacy:

1. **Access**: only users who have an active account in the cloud can access cloud services, including users' EMRs.
2. **Anonymity**: users and professionals are referred to using their IDs instead of their names.
3. **Sharing**: the cloud issues a new unique ID for each HELP request, then links this ID with the requesting user's ID. This means that users' data can only be accessed through an active HELP request that is already assigned by the cloud to the particular professional who has resulted from the look–up process.
4. **Storage**: users' data that are stored in the cloud will be encrypted using one of the well–known encryption protocols, such as the Advanced Encryption Standard (AES) [156].

Preventing malicious use of the service

As mentioned previously, the first form of help requires the user to complete a form with brief medical information to help the cloud find a suitable professional.

However, the cloud needs to ensure that a HELP request comes from a real user and is not being used as a method of attack (e.g., to slow down activities in the cloud).

Therefore, the user is required to type a code that is generated randomly after completing the form, whereby the user has only three attempts to enter the correct code and thereby proceed with the request.

6.2.3 Experiment design

To determine the feasibility of the proposed service, the Android app presented on the previous chapter was extended and installed on three Android–based devices: one HTC and two Samsung S3 smartphones.

The database and all the services were hosted by an Amazon instance [140]. In addition, an SMS gateway was deployed using the Twilio [152] web service API to allow SMS messaging.

The app provides two main interfaces, one for each type of user (professionals and members of the public), including registration and login screens. Appendix B provides more details about this app and its usage.

6.2.3.1 Forms of help

As mentioned previously, three ways of seeking help were designed, depending on the requesting user's medical status.

First, users can send a HELP request to the cloud by completing a form using the Android app. Second, a blank activity was created that only has a 'HELP' button. Once this button is clicked, a request is sent to the cloud that includes the user's ID and location.

In both cases, a look–up operation is started, as well as sending the user's information to the nearest emergency centre. If a suitable professional is found, the request is redirected to the professional selected, providing the location of that user, and the professional is allowed access to the requesting user's EMR. The cloud then waits for any update made by the professional to the user's status.

Coming to the third form of help, according to some researchers [157], the normal heart rate among adults is approximately 70 beats per minute (bpm) at rest, while one medical research group [158] gives the normal resting heart rate for adults as ranging from 60 to 100 bpm.

These rates are not the same for everyone, as bpm depends on a number of factors, such as weight, age, and general health.

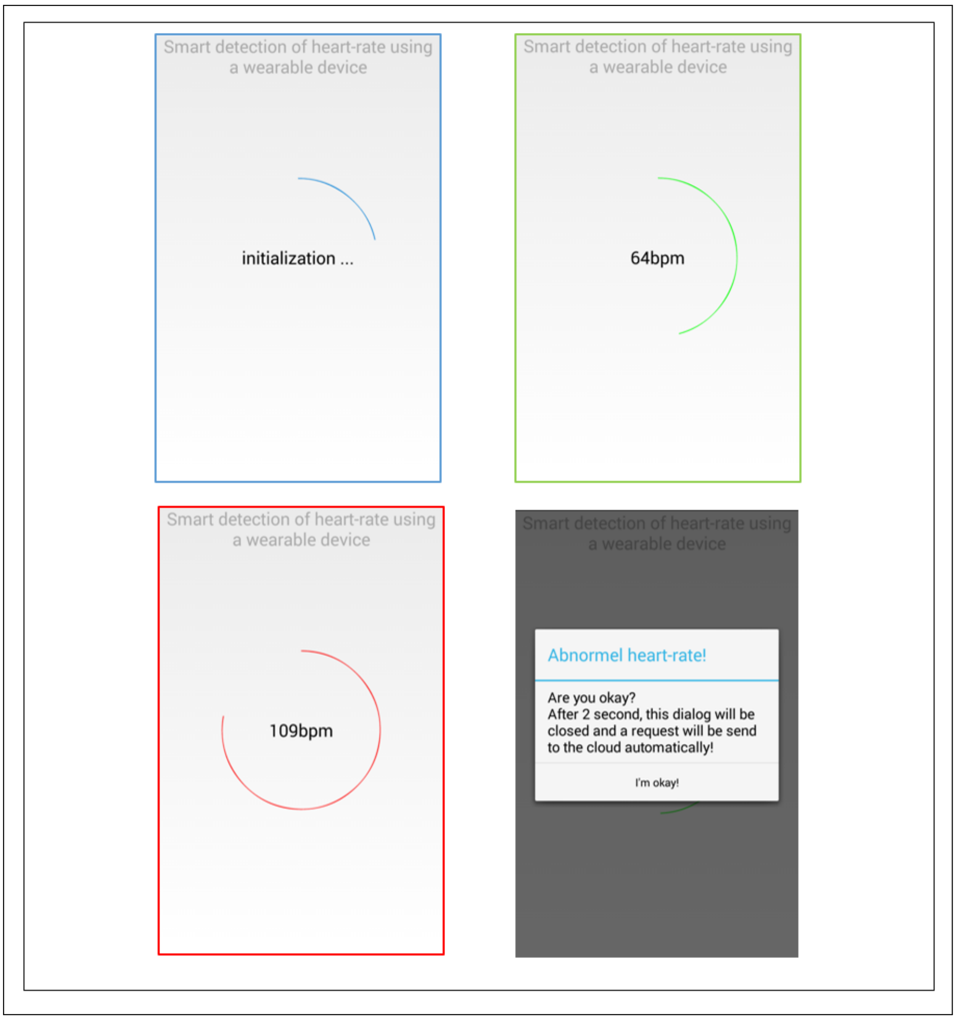
Therefore, this service has implemented a setting whereby when a heart rate is outside the 60–100 bpm range, an emergency case is detected.

A web–based app using a Tizen Wearable SDK [159] was developed and installed in a Samsung Gear S smartwatch [160] to monitor the user's heartbeat rate and send this to a paired mobile device. Transferring heartbeat data was achieved using the Samsung Accessory Protocol (SAP) [161].

To avoid false detection, a countdown (of 5 seconds) starts running with the option of cancelling the alert if an emergency case is not in progress.

If no action is taken by the user, an alarm is triggered to draw any people who might be in the proximity and, at the same time, a HELP request is sent to the cloud marked as 'a high–risk case'.

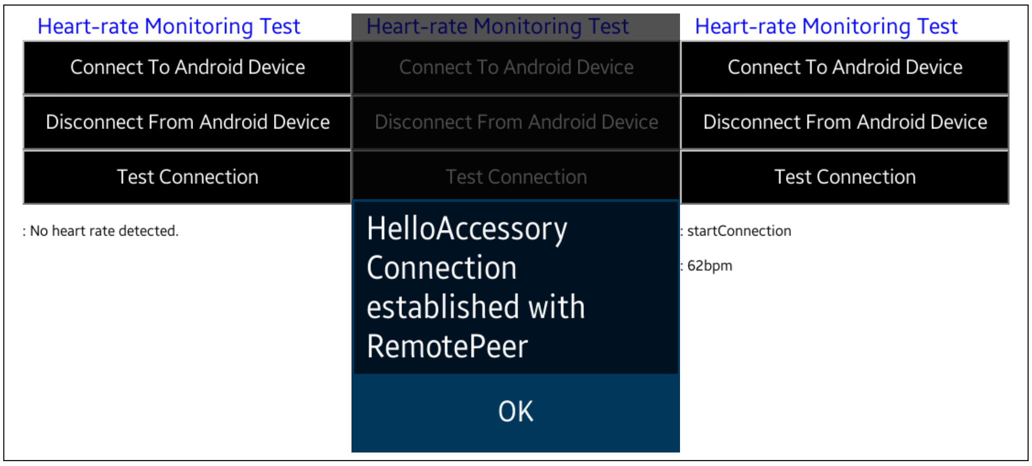
The cloud then notifies an emergency department, as well as starting a look–up operation. Figure 6.5 presents screenshots from the Android app of the smart detection of an emergency case. Figure 6.6 shows screenshots from a Gear S app.


Smart detection of heart-rate using a wearable device
initialization ...
Smart detection of heart-rate using a wearable device
64bpm
Smart detection of heart-rate using a wearable device
109bpm
Smart detection of heart-rate using a wearable device
I'm okay!

Abnormel heart-rate!

Are you okay?
After 2 second, this dialog will be closed and a request will be send to the cloud automatically!

Figure 6.5: Heart–Rate Monitor and Detection Using a Gear S Smartwatch and Android App


Heart-rate Monitoring Test
Connect To Android Device
Disconnect From Android Device
Test Connection
Heart-rate Monitoring Test
Connect To Android Device
Disconnect From Android Device
Test Connection
OK
Heart-rate Monitoring Test
Connect To Android Device
Disconnect From Android Device
Test Connection

: No heart rate detected.

HelloAccessory Connection established with RemotePeer

: startConnection

: 62bpm

Figure 6.6: Screenshots from a Gear S Smartwatch Showing How to Start a Sap Connection and Feed Heart Rate Data

6.2.3.2 Chat application

In the Android app, a chat application feature is offered with the help of the cloud service. This feature allows the requesting user and the selected professional to share text, photographs, and files after the look–up operation in the cloud has finished. Two scenarios are considered here, depending on the status of the connection of both users. If the two users are connected to the same MANET and can communicate directly, the chat application starts using their MANET IP addresses. However, if the two users cannot communicate directly, the chat application makes use of a notification–based format with the help of GCM [146], as in Figure 6.7. Therefore, when a user sends a message to the selected professional, it will be sent from that user to the cloud and then to the GCM platform responsible for delivering the message to the professional.

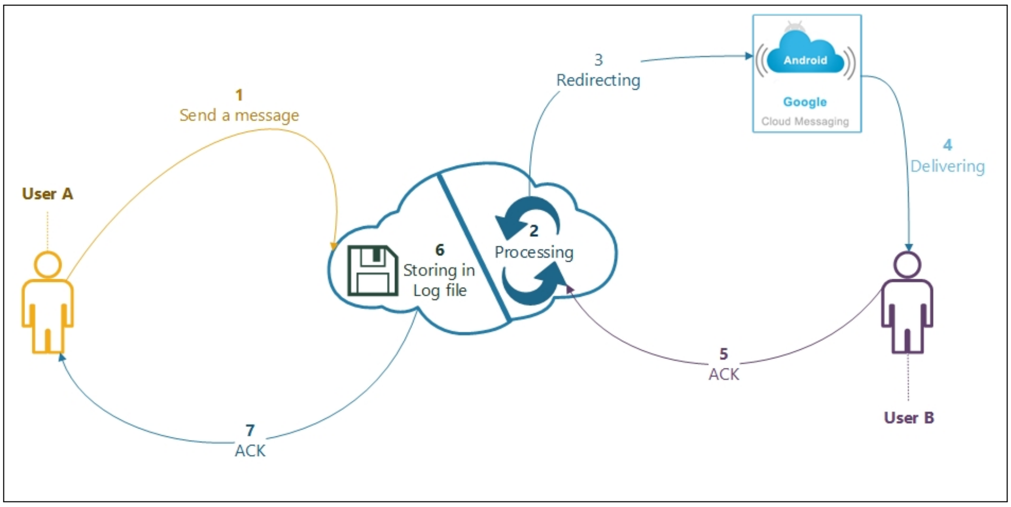


Figure 6.7: Chat application with the Help of The Cloud and GCM

Similarly, the response will travel all the way back to reach the sender over the cloud and the GCM. Therefore, after receiving a HELP request and a professional is selected at the end of the look–up operation, both users can then start a chat session that is served by the cloud and GCM.

6.2.3.3 Seeking medical help via SMS

Mobile devices can use SMS as a supporting communication method to enhance communication with other users/systems [162]. For example, one possibility is updating current location (using GPS coordinates) via SMS if the Internet cannot be reached/accessed.

Technically, sending and receiving SMS in the Amazon cloud platform is provided only to US mobile numbers. Therefore, the Twilio API was used to allow HELP requests to be received via SMS, as well as to reach those (e.g., professionals or users) who do not have an Internet connection but can be reached by SMS.

Put simply, the app constructs the HELP request in the background as an SMS payload and sends it to the Twilio platform, which will then be responsible for redirecting the request to the cloud. An acknowledgement will be sent to notify the user that the cloud has received the message successfully.

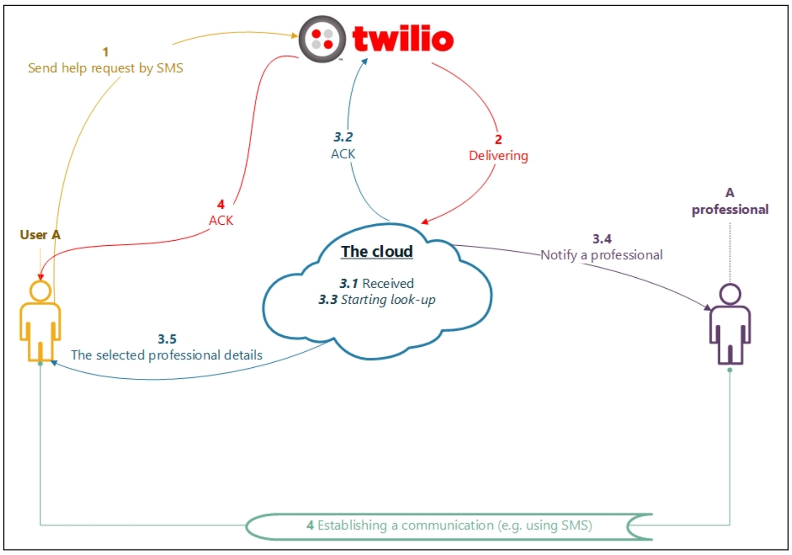


Figure 6.8: Requesting Medical Help by SMS Messaging

Then, a look–up operation will be started to select the most suitable professional. When one is found, a request is sent to that professional that includes the sender's contact information in order to start an SMS conversation. The cloud also feeds the selected professional's details to the requesting user. Figure 6.8 shows these steps.

6.2.4 Evaluation

The experiments were carried out 10 times to determine the average time consumed in setting up the connection between the person in need and the first responder. Table 6.1 shows the RTTs for a HELP request using different communication methods. The average time needed to set up a connection for help requests between a user and one of the professionals listed in the cloud directory was between 4 and 25 seconds, depending on the communication method used.

During that time, the request is directed to a professional with the expertise appropriate to the medical situation, as well as being located within reach of the requester's location.

In addition, the form of help had no effect on the amount of time that was taken because the delay was captured from when the request was sent from the mobile devices until the acknowledgement (ACK) was received from the cloud stating that a professional had been selected and was engaging with the medical case.

Table 6.1: Cost of Setting Up Connection

Communication method		**RTT (Seconds)**
Wi-Fi	Home broadband	4.077
	4G portable broadband	5.231
Cellular Network	3 G	8.432
	SMS	24.775

6.2.4.1 Medical response time

The time needed to respond to medical emergency cases is an important factor in healthcare fields when evaluating the quality of emergency medical services [163]. Response time is defined as the period between receiving an emergency call and the arrival of the rescue team (e.g., an ambulance crew) at the emergency location [164] [165]. This time has been standardized as follows: "Response times of four minutes for BLS [basic life support] first response and eight minutes for paramedics have become an international standard for urban EMS [emergency medical services] systems" (p.45) [166]. However, exceeding this time would not result in lower patient survival rates in all cases, according to real–world experiments presented elsewhere, and it is suggested that realistic response time standards should be developed that take into account the needs of each case [167].

However, the first responder service presented here achieved a response from a medical professional in a matter of seconds, as well as delivering details of an emergency case to the emergency services. This suggests that the amount of extra time involved will not affect the overall response time for this type of emergency case. This amount of time can increase, however, if the selected professional does not respond as soon as the request is received.

To avoid any delay that might be caused by the human factor, the cloud gives the selected professional 1—3 minutes to interact with the received request and, if there is no response, the request will be assigned to another professional. However, in some situations, this delay might not occur; for example, in the Al–Hajj scenario, there are usually a number of medical volunteers who are readily available to serve pilgrims.

6.2.4.2 Network issues

Another factor that might have an impact on the length of time involved is the quality of the communication links. As mentioned previously, the ability to use SMS messages as an alternative communication method is discussed. Here, the cloud or users can communicate via SMS messages to avoid having to rely on low–quality Internet connections.

Furthermore, in the case of a crowded environment or an inability to reach the cloud directly, a user could benefit from the MANET–cloud model by creating/joining a MANET network that is managed by the cloud, as reported in the previous chapter. This means that a user can create or join a MANET network to submit details of his/her medical needs. The other member of this MANET can be either a medical professional who can provide help to the user or simply an ordinary user who acts as a bridge to allow communication between the user who needs assistance and the cloud.

6.2.4.3 Further analysis

To compare this proposed idea with existing types of systems, such as calling an emergency help centre (e.g., 999) by phone, the steps that are involved in the emergency operator processing a help request and the duration of each step have to be understood. These steps have been defined as follows (see [167] for more details):

Regulation

The interval between the time the centre receives the call and the time the nearest available rescue team is notified. This includes the call handler's pre–analysis time, the medical evaluation time, and the time taken to pass the request to the original hospital in the case of less urgent emergency calls..

Preparation

The interval between the time the call handler notifies the care team and the time the care team leaves for the rescue.

On–site

The time interval between the care team arriving at the scene and the time it leaves the scene.

Diagnostic or therapeutic radiography (DTR)

The time interval between the care team arriving at the DTR medical service and the time it leaves this service.

Drop–off

The time interval between the care team arriving at the destination hospital and the time it leaves the hospital.

As the proposed mobile cloud service does not conduct any real pre–analysis or perform most of the actions included in the regulation step above, its results cannot be compared with the amount of time taken during this stage. However, the proposed service allows users to search for a medical professional, which results in the building of communication between the user who requested aid and the professional selected, as well as redirecting the emergency case to an available professional after pushing any available information to that professional to enable him/her to reach the requester. Therefore, it should be possible to compare the results presented in Table 6.2 with the time taken in the preparation step. The paper [167] provides the average real regulation and preparation time of 6,658 received calls in an emergency call centre in France for a period of 11 months from 1st October 2010 to 31st August 2011 (Table 6.2).

Table 6.2: Average Emergency Processing Times (Minutes)

Type of call	**Priority level**	**Avg. regulation** t	**Avg. preparation** t
Primary	*1*	6.5	3.3
	2	12.8	3.8
Secondary	*1*	22.8	4.9
	2	53.8	6

In the above table, the lowest average for preparation time was 3.3 minutes, which is far higher than the preparation time of the service proposed here of 25 seconds, taking into account that the latter service sends the received request to the emergency department before taking any other action, such as the look–up operation. This is intended to deal with the issue of “no professional is found” and ensure the request is redirected to an emergency department.

Furthermore, if it is assumed that the selected professional is in the same location as an ambulance or rescue team, reducing the preparation time needed will lead to a reduction in the total waiting/response time.

As mentioned previously, this service is aimed at finding a medical professional who is available and, more importantly, located near the scene of the emergency. This means that there will be a high probability of finding a professional who can interact with the emergency case immediately and is located close enough to the emergency location to reduce delays resulting from travelling to the emergency scene. As a result, the total response time will be improved and will, it is hoped, lead to a better outcome.

6.3 Service 2: Disaster Management

The previous section presented a mobile cloud service that is aimed at providing medical help to people in the case of an emergency. However, all emergency events require a fast response and decisions based on first–hand information. Therefore, this section presents the second mobile cloud service, which makes the best use of social media applications, such as Twitter, in emergency and risk management.

Here, risk and emergency teams can receive data in a matter of seconds that can inform their decisions when an emergency has affected areas under their management. The proposed service allows users to provide on–the–ground information regarding such an event, as well as early notification to people who are in the vicinity of an emergency situation. The service matches users’ requests to a set of pre–defined labels that will help rescuers to understand the situation more clearly. The service was implemented and tested with Android devices and a cloud–computing instance hosted on an Amazon platform. A cloud–based tool is also provided for risk and emergency management teams to interact with users’ requests. The experimental results show that the system enhanced the early detection of emergencies.

6.3.1 Objectives

One of the main goals of this service is to identify an emergency event that has occurred in different areas of a city and that might cause damage to the main infrastructure, cause people to panic, or result in the loss of life. The service will, in addition, carry out further actions, including sending notifications advising users what to do in the case of an emergency situation and building interactive maps. Furthermore, the service includes a historic database that could be used if the same type of emergency event occurs again, in order to improve risk and emergency management.

The main benefit of this service is that it enables smart actions to be carried out based on results from the data analysis process, which means that this service can raise an alert with an emergency department if a high–risk event is discovered or provide a real–time and interactive map that can be made public and which shows both the affected areas and safe places. The cloud can also use data collected from social media to warn people who are located close to an emergency.

6.3.2 Service design

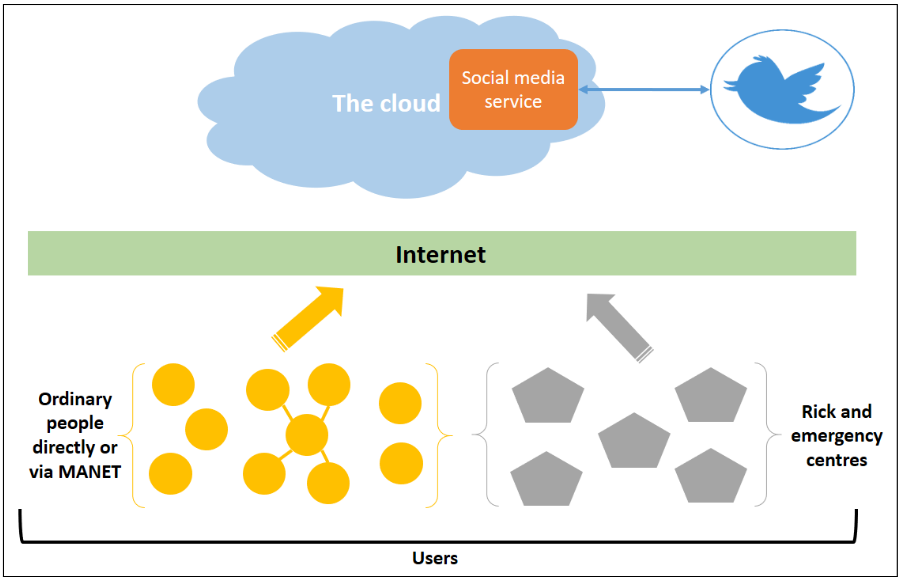

The cloud
Social media service
Internet
Ordinary people directly or via MANET
Rick and emergency centres
Users

Figure 6.9: Social Media Service Design Overview

The new service is hosted in the cloud and connects to a social media application (Twitter was selected in this thesis) to retrieve real–time information regarding emergency events.

This service will also set up real–time communication between members of the general public and emergency management centres, whereby help can be sought via social media and rescue teams or emergency services staff can send useful information to the public. Figure 6.9 shows the way in which this service is connected to a Twitter app for searching and retrieving purposes. The figure also shows that users can interact with the cloud over the Internet, either directly or via neighbouring links. Risk and emergency centres can also benefit from using this service, for example, by monitoring or tracking an event.

The architecture and components of this service are shown in Figure 6.10. The first component is listening to live Twitter streams. As a result, all tweets will be received/collected here. The next step is storing the received streams in the database before starting the analysis process. This will help the cloud to ensure the origin of the data after the results of the analysis process have been provided. In other words, emergency centres can extract the raw data of an event to explore tweets and multimedia that have been collected. However, to avoid storing a vast amount of data, the cloud will delete all unnecessary data collected after the risk of an event has passed or after a certain period of time (one month, for example). Another action the cloud can take to reduce database usage is by keeping tweet IDs in a table that is linked to an event so that these can be retrieved whenever needed.

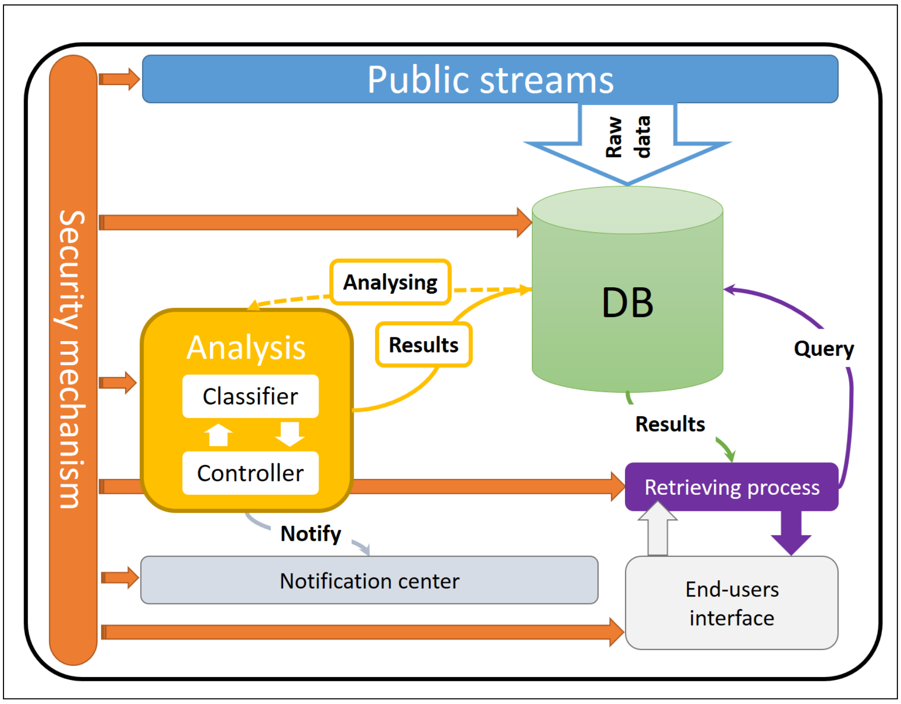


Figure 6.10: Social Media Service Architecture

Once the raw data are stored in the database, they will be sent to the analysis component, which consists of a controller and a classifier responsible for classifying tweets into meaningful categories. The analysis process includes three main stages, as shown in Figure 6.11 and detailed below:

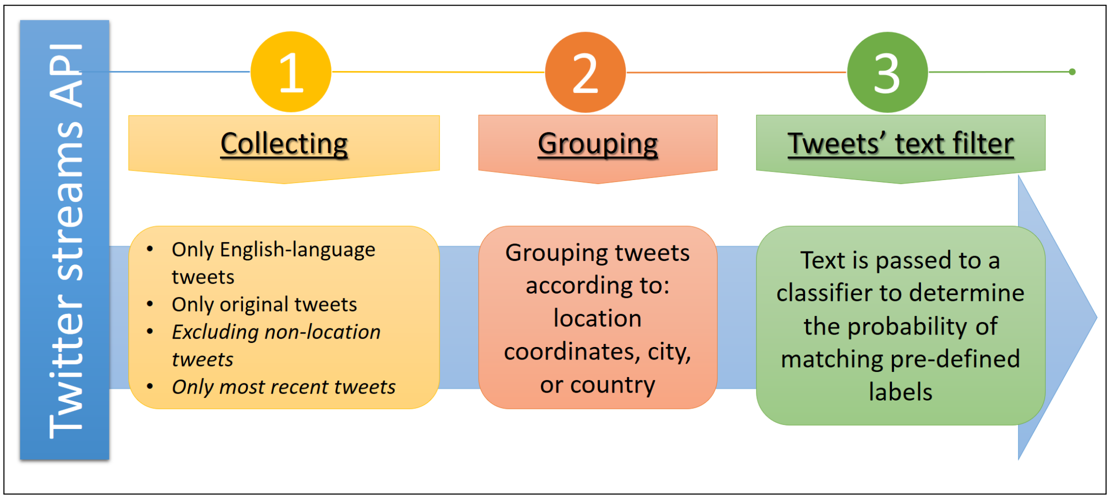

Twitter streams API
1
Collecting
2
Grouping
3
Tweets' text filter
Only English-language tweets
Only original tweets
Excluding non-location tweets
Only most recent tweets
Grouping tweets according to: location coordinates, city, or country
Text is passed to a classifier to determine the probability of matching pre-defined labels

Figure 6.11: Analysis Stages

1. **Collecting stage:** only tweets that contain English–language text are collected here and, to avoid duplication, only original tweets are included. This means that no retweets or in–reply tweets will be collected. Tweets without location coordinates will not be passed to the next stage either.

2. **Grouping stage:** tweets are grouped based on their location (e.g., coordinates, city, or country). This will help in detecting a risk event. For example, if a high number of tweets come from almost the same location, the system can highlight this information and send a notification to the management centre stating the possibility of a large event occurring.

3. **Classification stage:** each tweet is passed to the classifier to determine the probability of matching a set of pre–defined labels, such as “fire”, “abuse”, etc. More information about the classification stage is provided in the implementation section.

The results of the analysis are organized in the database by event, attaching any associated multimedia such as photographs or videos. These results can be used in two ways:

Firstly, in the retrieving process, which starts once a user, whether a member of the public or an official, looks for information about an event and provides some keywords for that event.

The query is sent to the database, which looks for data that present the best match. Once the results are found, they will be pushed back to the requester. The results that are returned can be photographs, tweets or texts, depending on the requester's specifications, as well as relevant results from previous events found in the database.

However, the level of the returned data will depend on the role of the requester. For example, a requester wanting to check the latest status of an event will receive only sufficient information to serve his/her needs. In contrast, more specific data with certain recommendations will be served if the requester is an officer in an emergency centre or a rescuer who is at the location of the event.

Secondly, to notify an emergency department or rescue team when an emergency event is detected, the cloud can use social media and the processed data to notify people who are either registered to receive risk event updates or those who are in the vicinity of a detected emergency event. This kind of notification will help to reduce the impact of an emergency event and the possibility that it will spread and affect more people.

One example is that it could be used to notify drivers that they should avoid roads that are flooded. Pushing these types of notification can be done by collecting the current location of users who are already registered in the cloud. Once they cross into areas that are marked as dangerous zones in the cloud database, an alert is sent to them. To avoid distracting people while they are driving, the cloud can send a short voice alert to their mobile device, such as "Hazard ahead!" or "Road closed!" (see Figure 6.12 for more information).

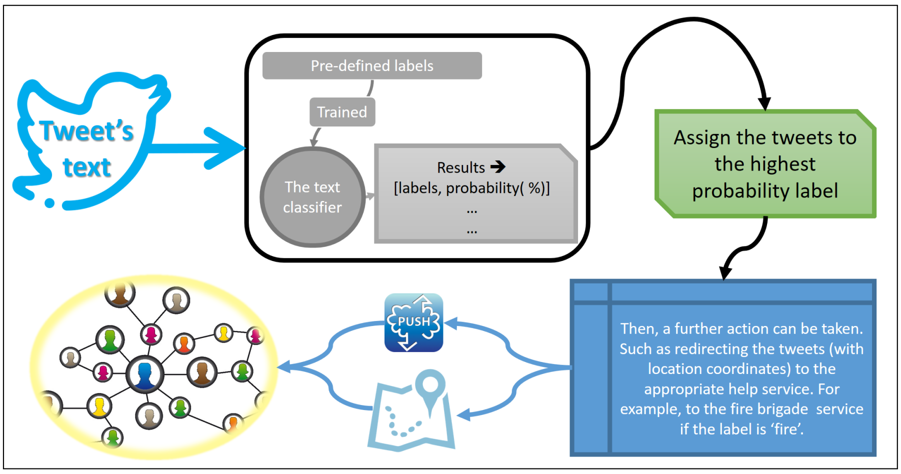

Tweet's text
Pre-defined labels
Trained
The text classifier
Results ➔ [labels, probability(%)] … …
Assign the tweets to the highest probability label
Then, a further action can be taken. Such as redirecting the tweets (with location coordinates) to the appropriate help service. For example, to the fire brigade service if the label is 'fire'.
PUSH

Figure 6.12: Tweet Classification Process

The final component of this service is security. The following three main actions are taken to ensure a high level of protection when accessing/using this service, as well as a high degree of privacy regarding storing and retrieving data.

1. Only users who have an active account in the cloud can access and use this service. This will reduce the risk of the misuse of this service; for example, by sending a large number of requests to slow the system or misusing the analysed data by carrying out an attack in a place that was considered safe or where there are a large number of people. The cloud allows users to use their credentials in the Twitter app to access cloud services. This will allow as large a number of trusted users as possible to benefit from the cloud services (more details are provided in the implementation section).

2. The cloud will, at all times, refer to tweets using their IDs to ensure that ownership of data is held by the originators. The cloud will redirect a request to the Twitter app if it needs to retrieve actual tweets.

3. All data stored in the cloud database will be encrypted. Users can only retrieve the results of the analysis of these data, even if the retrieval request comes from a registered emergency centre looking for extra information about an emergency event.

6.3.3 Implementation

Moving to service implementation, two perspectives are discussed here:

- How the system is implemented to allow end users, both members of the public and risk and emergency teams, to interact with the system and benefit from its features.
- How the social media service is deployed in the cloud and how the cloud will handle users' requests and carry out appropriate actions based on these requests.

The following subsections present the five main parts of the implementation of this service, taking the two perspectives into account.

6.3.3.1 Labelling and classification of live streams

One important feature of using social media is the analysis of live streams in order to produce meaningful results. For example, a company might analyse tweets to check the success of its products and to understand how the products fit within the market. Another use could be to deliver recommendations to users that may attract them based on their activities (e.g., tweets and followers), such as recommending a place to visit while travelling or an account to follow. Most existing tools or systems analyse the text of tweets to check how positive or negative the user is being about a particular product or idea [168] [169] [170]. This is conducted using natural machine learning (NML) features to analyse the text [171].

Here, six main labels were defined: “emergency”, “fire”, “abuse”, “healthcare”, “crowd”, and “disaster”. Each of these labels contains a set of words that explain each label. For example, “flood” and “storm” are included under “disaster”, while “bleeding” and “accident” come under “emergency”. This is called a bag–o-f-words (BoW) approach [172].

The classifier was then trained to these labels before starting the classification process. A Naïve Bayes protocol [173] was used because of its simplicity and propriety. As a result, when a new tweet is received, the service will:

1. *Pre–process* the text of the tweet by extracting the Uniform Resource Identifiers (URLs) and mentions of users’ names and delete stop words, such as “the” or “and”, and punctuation [174].

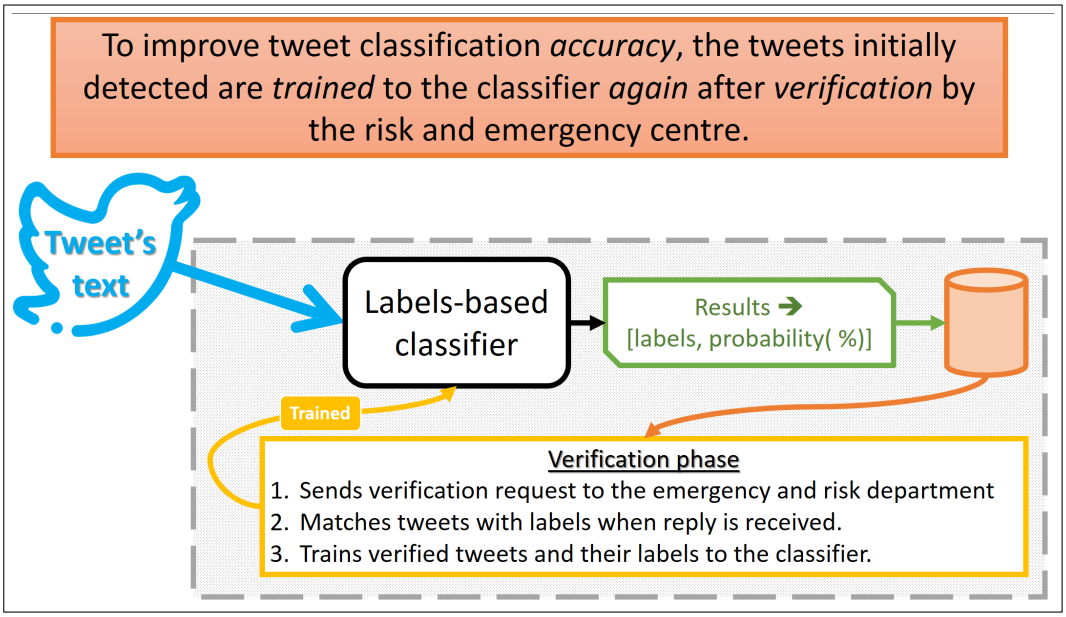
2. *Calculate* the number of occurrences of each word from the pre–defined list of labels in the text of the tweet.

The results will then be attached to the tweet to be stored in the database. The labels include words and terms commonly used in an event. However, the results gained using the classifier will not be as efficient as they could be.

Thus, what the researcher calls a two–tier trained classifier protocol (as in Figure 6.13) was added, which means that when the classifier assigns a new tweet to a certain label, the result is sent to an emergency and risk department for confirmation.

Once an event is confirmed, the text of the tweet will be sent again to the classifier to enhance the production level/probability of the featured classification.

In this case, tweets are classified using labels to better understand the request, as well as to be able to carry out further actions, such as calling the fire brigade in the case of a fire. Moreover, the system can provide more data about a location that has a particular label. For example, if there is a tweet that is labelled as denoting a flood in a certain area, and this event had previously been reported by a user in the past and labelled as a flood by the system, the management centre can then consider that there is an issue in this location and that the possibility of a flood having occurred is high. Another example is if a tweet is labelled by the system as indicating abuse and older tweets were labelled in the same way for the same area. The management centre can then consider that there is an issue in that vicinity that needs further investigation.


To improve tweet classification *accuracy*, the tweets initially detected are *trained* to the classifier *again* after *verification* by the risk and emergency centre.

Verification phase
1. Sends verification request to the emergency and risk department
2. Matches tweets with labels when reply is received.
3. Trains verified tweets and their labels to the classifier.

Figure 6.13: Two–Tier Trained Classification Design

6.3.3.2 Identifying an emergency event

There are two main ways in which an occurrence can be defined as a high–risk or emergency event that needs to be managed to reduce its impact and avoid damage or loss of life. The first is when a city emergency and risk centre issues an alert to the cloud and defines an event as having large–scale risk. Here the cloud starts monitoring and tracking the propagation of this event, as well as collecting relevant data from social media, particularly Twitter, as chosen above. The second method is when the cloud receives requests from public users about an event when they are looking for help in dealing with the situation.

Thus, if the cloud continues to receive requests from a high number of different users in almost the same location with similar descriptions, an alert will be triggered and sent to the risk and emergency management centre closest to the event and the cloud will consider this as a high–risk situation. Once the alert triggered by the cloud is confirmed by the risk and emergency management centre, the cloud starts monitoring/tracking this event, as well as collecting relevant data and then analysing them to serve members of the public who may be affected by this event or submit recommendations to centres responsible for managing risk and emergency events.

6.3.3.3 Historic database

Once an event is defined as an emergency, the cloud starts to collect data in order to build an information history. The data collected will include what action has been taken, what types and amount of data have been exchanged/used (tweets, images, videos, etc.), the areas affected and the level of damage on the ground. However, sending enquiries to this database will result in more statistics or metadata, rather than a full chunk of data, that need to be analysed.

The main purpose of this database is to increase the level of efficiency in managing similar or repeated risk events, which will result in less damage and reduce the effects of events that occur frequently (such as winter storms). For example, if a city is affected by a flood that causes significant damage to property or results in death, and the cloud has analysed historic data for this flood that it had stored in its database, these data will be used if a similar flood occurs. This will enable the emergency management team to take defensive actions in areas that have a high risk of flooding or track river levels that had risen in a previous flood and caused the main damage.

6.3.3.4 Interactive map

One of the important features of the proposed service is the ability to carry out smart actions that will help in an emergency event. One example consists of sending real–time notifications to users about an event. Another type of smart action is that, in the event of an emergency, the cloud can build a real–time map to show the affected areas so that they can be avoided by the public and focused on by the emergency management centres.

In addition, safe areas can be shown so that resources and medical volunteers can be directed to where they are needed. People who require medical help can also reach these safe areas and receive medical support. Entries to this map will be made after analysing the data collected by the cloud and the coordinates/locations that are sent from the risk and emergency management centres.

6.3.3.5 Building a communications platform

Another important feature to be gained from introducing a social media network to cloud services is the ability to achieve two–way communication between the cloud and end users, whereby users can seek help from the cloud through Twitter, the social media chosen for this research, as well as the cloud being able to reach as many users as possible through Twitter. To make it easier for the cloud to receive requests and submit information to the public, the system can benefit from using the hashtags featured in Twitter.

The following hashtag was set up: #TheCloud. The cloud monitors this hashtag 24/7 to receive requests from users and to send information to users through this hashtag. As a result, each request sent to the cloud has to include the #TheCloud hashtag and users have to monitor this hashtag to receive the latest updates from the cloud regarding an event. In addition, the cloud will listen to live Twitter streams to report an emergency or risk event to the appropriate centre, such as a city council, the police or fire brigade.

6.3.4 Experiment design

The experiment was designed with two main environments in mind, depending on the type of user who accesses the mobile cloud social media service:

End–user environment, which includes the design of how people generally can access the service and gain from its features.

Rescue teams' environment, which allows risk and emergency teams to interact with the cloud services, such as by verifying the classification of tweets, receiving notifications, or looking at an interactive map.

Furthermore, information about how the cloud is configured to listen to a Twitter stream, as well as how the service analyses and stores the data, is provided in the following subsections.

6.3.4.1 The end–user environment

Once the service has been configured to listen to a public stream, as shown in the service design section, users can access an official Twitter app or website to tweet their HELP request after ensuring the location service has been enabled.

However, to make it easier for users to interact with the service without having to pay attention to the system requirements, as well as benefiting from the previous solutions in dealing with session interruptions and connection issues, the previous Android app was extended to allow users to undertake a number of actions with the help of the twitter4j [174] library, as shown in the following sections. Appendix B presents screenshots of how these actions can be used in the Android app.

Login using a Twitter account

To extend the user base of the system and introduce another way to access the cloud services, users can use Twitter account credentials to access these services.

Sending a help request by tweet

Users can send a help request to the cloud using the same method that Twitter uses to send tweets. In other words, the user has to provide a brief description of what he/she feels or what he/she is facing and then, in the background, the app re–organizes the request as a tweet.

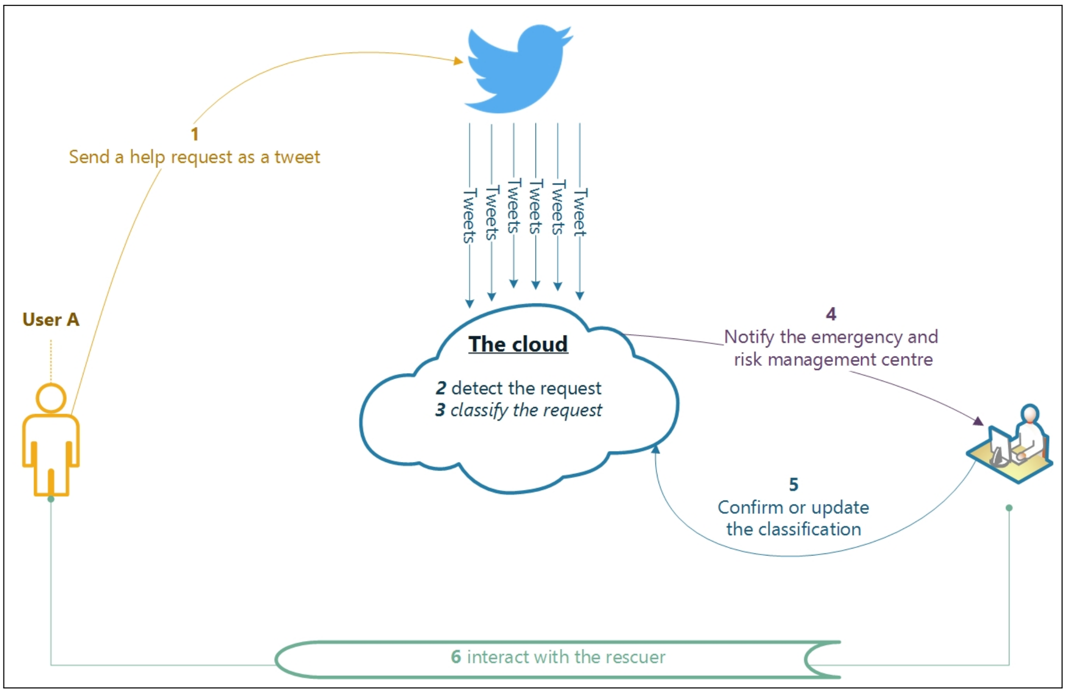


Figure 6.14: Requesting Help by Tweet

This means adding the #TheCloud hashtag, highlighting keywords, and ensuring that the body of the text is below the limit (140 characters). This preparation will ensure that all help requests match the cloud service requirements, such as the GPS of the mobile device being on and enabled. The text of this request will appear on the requester's Twitter timeline. Figure 6.14 shows these steps.

Reporting an emergency

Another benefit of integrating social media, as mentioned above, is the ability to collect valuable on–the–ground information, including texts and images or even short videos. Users can send what they see on the ground through the Android app as if they were using a Twitter app. The main difference in this case is that the request will match cloud requirements, such as providing a current location, as well as in respect of the Twitter app.

Watching updates of nearby emergency events

Users can use the app to check what emergency events have been detected for their current location and browse the latest updates from the cloud, which are the results of the data retrieval process in the social media service.

On the same screen, the app also shows a map that has the most recent updates on the ground.

6.3.4.2 The rescuers' environment

The system needs to interact with risk and emergency centres, both with staff who are in the office and those who are on the ground during an event. Interactions are divided into two aspects: notification and confirmation. For the first, the cloud needs to send notification about a certain event that has, for example, been reported by a user who tweeted a help hashtag to the cloud.

The second aspect is important for ensuring the level of efficiency of the system, which means that when the cloud has detected that a large–scale event is occurring, based on a high number of tweets received from the same location, this information has to be verified by a member of the risk and emergency department to avoid raising a false alarm. Thus, two web pages was designed: one page lists the results of all the classified tweets and asks a staff member to confirm detection or even suggest a new label if one is needed; the second page contains an interactive map that shows live requests containing information.

For evaluation purposes, an account was created for a risk and emergency centre to do what such centres do in the event of an emergency, such as posting updates and replying to users.

6.3.4.3 Consuming and analysing a Twitter stream

A new Node.Js–based service was created [175] in the Amazon cloud platform [176] to listen to live Twitter streams using the Twitter streaming API [177]. However, because it is not possible to listen to public streams without a filter, the #TheCloud hashtag was set as a filter when implementing the API streaming. Once a new tweet is received, the service starts. It first sends this tweet to the classifier, which calculates to which label it belongs, then redirects it to the most appropriate help centre based on the classifier prediction.

6.3.4.4 Storage medium

One database is used here, which is an instance of the Amazon Relational Database Service (RDS) [141], to serve all the storage needs, including the historic database. Various data from different components need to be stored in the database, such as:

- Tweets received from live streams.
- The results of analysing the tweets.
- Pre–defined labels with their bag of words.
- Tweets that are initially classified by the cloud and then confirmed by the emergency and risk department.

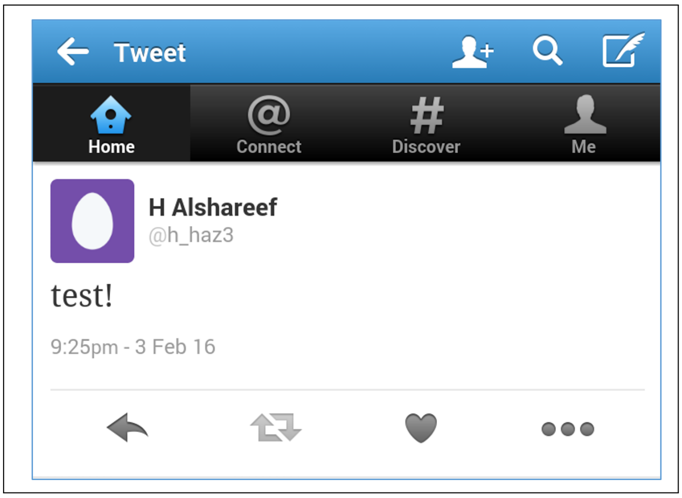
6.3.5 Evaluation

This part of the research is interested in ensuring that users can interact with the service with only a reasonable delay, as well as receiving support facilitated by the cloud as quickly as possible. In addition, it was important to ensure that the classifier could organize the received streams into meaningful categories or label the type of help required in submitting a request to the appropriate emergency centre.

Two experiments were run, on system feasibility and classifier performance, and are outlined in the following subsections.

6.3.5.1 System feasibility

The service was tested with tweets from an active Twitter account, whereby these tweets arrived successfully in the cloud, stored in its database, classified and the results updated in the cloud database. Finally, the tweets were shown on the account time line (as in Figure 6.15) and the rescuers' panel (as in Figure 6.16). In addition, all the information contained in the tweets, such as name, location, initial classification result, was captured and forwarded, which means that communication can then start between the rescuer and the owners of the tweets.

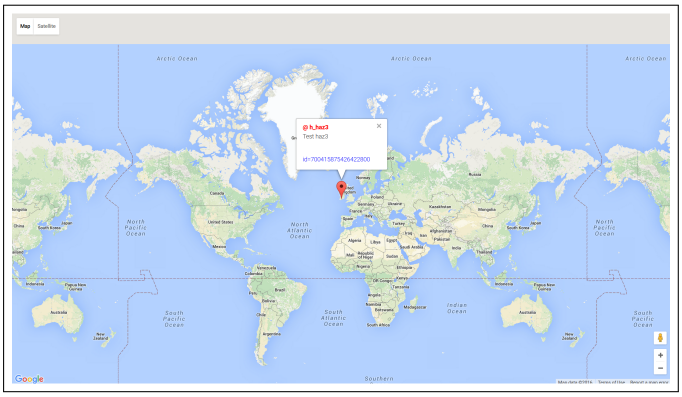


H Alshareef
@h_haz3

test!

9:25pm - 3 Feb 16

Figure 6.15: Twitter Account Time Line Showing the Test Tweet



@ h_haz3
Test haz3
id=700415875426422800

Figure 6.16: Interactive Map Showing Tweets Received and Added to The Map

6.3.5.2 Classifier performance

The aim of this experiment was to evaluate the performance of the classifier, particularly when it is trained with tweets/texts that have been confirmed by an emergency and risk management department. First, a dataset that contains three different text messages that correspond to three types of emergency ("flood", "fire" and "personal health") was defined.

It was then sent to the classifier to calculate the likelihood of matching the pre–defined labels mentioned previously — this corresponds to a simple model. Then, the risk and emergency account that was created confirmed the initial detection and trained the classifier using this dataset.

After that, the same dataset was sent to the classifier. It was found that there was an improvement in the classification results, as shown in Figure 6.17.

A second dataset that had different texts but corresponded to the same types of emergencies was defined. Then, the probability of this dataset using the classifier that was trained only with the pre–defined labels and classified again after being trained with the first dataset was calculated.

The results in Figure 6.18 show that the proposed model not only matched the text with a correct label, but had a higher probability compared with the simple model.

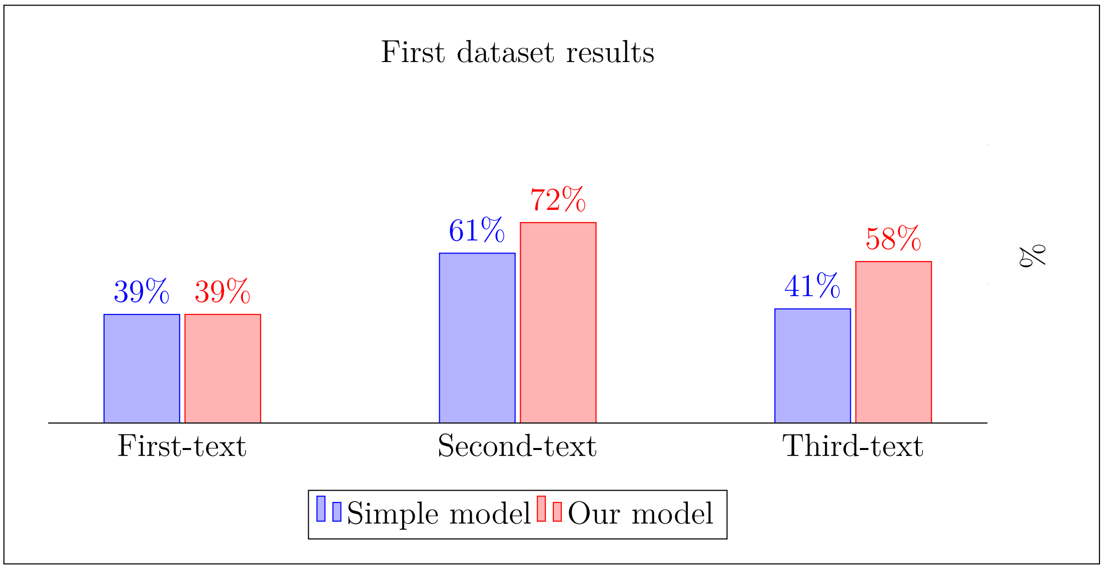


Figure 6.17: Comparing the Performance of Classifying Three Texts Using a Classifier That was Trained Using Pre–Defined Labels with a Classifier That Was Trained with Pre–Defined Labels and The Same Text as The Three Examples

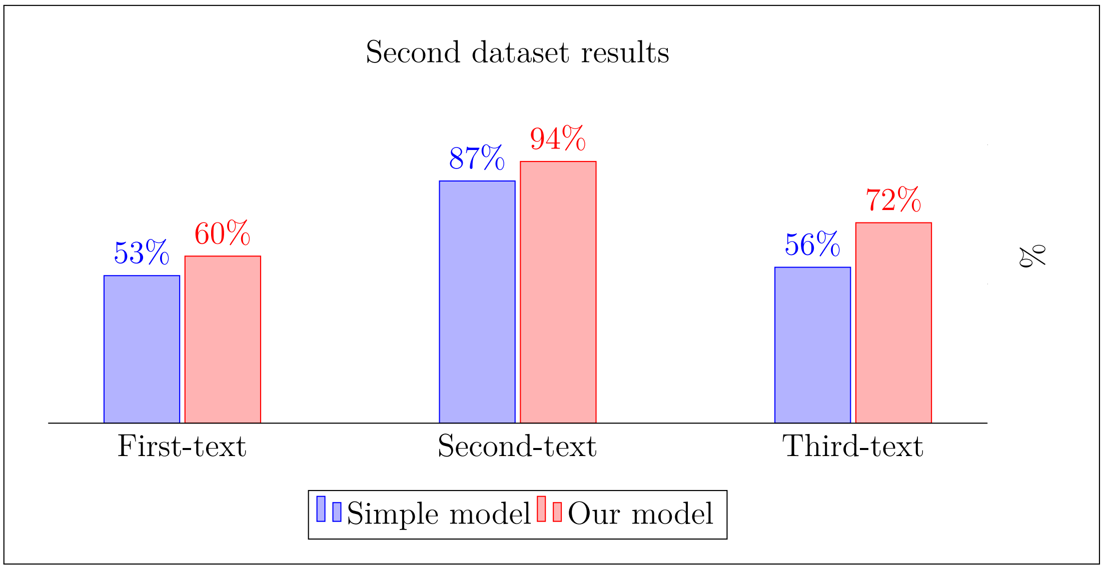


Figure 6.18: Comparing The Performance of Classifying Three Different Texts Using a Classifier That Was Trained Using Pre–Defined Labels with a Classifier That Was Trained with Pre–Defined Labels and The Text From The First Dataset

Theoretically, this result is plausible because, if the population of a label or category is increased, the likelihood of a new text matching this label will also increase.

6.4 Chapter Summary

This chapter presented two mobile cloud services that are aimed at helping people in emergencies and can benefit from the previously mentioned MANET–cloud model in dealing with issues such as limited access to the Internet, reaching the cloud, and allowing local and fast communication between other people in the same location. Both services are hosted in the cloud and can be accessed via mobile devices.

The main difference between the two proposed services is that the first stressed responding swiftly to a person who is experiencing an emergency and requesting medical help. This service offers people who are experiencing medical emergencies while on the move the possibility to 'look up' doctors or nurses who are located in their proximity and who can respond more quickly than the emergency services.

Three main forms of HELP request were designed to serve as wide a range of emergencies as possible and to enhance system usability, including smart detection using a wearable device.

Furthermore, details were established of a communication link that is managed by the cloud service, with the option to start a chat session to exchange text, photographs, and files. The SMS messaging ability is also provided as an alternative method for reaching the cloud services and seeking medical help in emergencies.

The service benefits from using cloud features, such as high performance and availability, and extended connectivity through the MANET–cloud model previously mentioned in this thesis.

The aforementioned service was designed and deployed on real mobile devices and real cloud instances as a validation test. Security enhancement was also added to the system by implementing a verification scheme to ensure that requesters are real people who are looking for medical help.

The set–up time was less than 25 seconds using SMS and less than 5 seconds using the Internet, which meets the initial time requirement of the system, as well as ensuring no extra delay is added to the total medical response time.

On the contrary, it will make the best use of a medical professional who may be located just a few steps away from the medical case.

Furthermore, comparing this amount of time with the time needed to prepare a care team with an ambulance using a traditional model (e.g., calling 999), it was found that the mobile service could significantly reduce delays in responses by selecting a professional who is able to interact immediately with the emergency case.

If the look–up process selects a professional who is located close to the requester's location (e.g., a few steps away), this will also decrease the time needed to reach the emergency location, which will result in an increased chance of survival.

Interestingly, the time results for this service are considered a requirement for proposing a system called the "OCarePlatform" [178], which aims at supporting independent living.

With regard to the second service, as is known, analysing social media streams to deliver better help and management in emergencies has recently attracted a high degree of attention.

The second mobile cloud service proposed takes advantage of analysing a live Twitter stream to detect any possible emergencies, as well as using social media networks as a communication platform to enhance system connectivity and disseminate help services/data.

The service was tested in a real–world scenario by sending tweets to the cloud to assess how the service performed. It was found that the proposed solution is feasible and can achieve its objective.

In addition, an experiment was run to test the performance of the classifier. It was found that the solution could improve the likelihood of detecting new tweets/texts when previous classification tweets/text are confirmed and trained.

Chapter 7

The Reference Architecture

7.1 Introduction

As an important part of this research work, this chapter presents and discusses in detail the reference architecture of the proposed middleware, then reviews the requirements for the proposed middleware that were defined in chapter 2. This chapter also provides an evaluation based on the research work in the thesis. The experimental results in chapters 5 and 6 are taken up to determine if the proposed system (including the middleware that uses a MANET–type network) meets these requirements.

7.2 The Reference Architecture

After discussing a number of scenarios, and how help can be requested from the proposed middleware system if an emergency occurs in one of these scenarios, this section presents the reference architecture and all the services that form the middleware system.

Figure 7.1 presents a diagram of the reference architecture and the following subsections provide additional details about the main components (e.g., services) of this diagram.

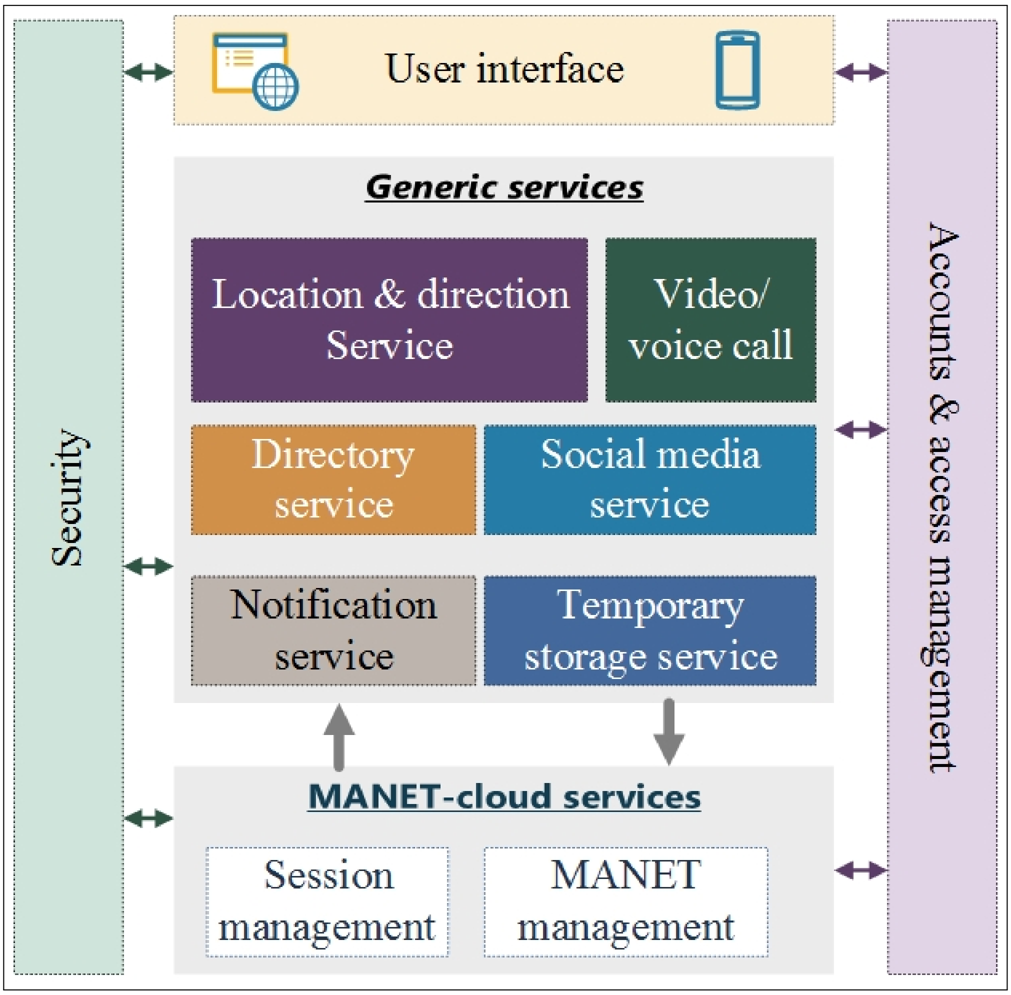


Figure 7.1: Reference Architecture

7.2.1 Directory service

Any directory system needs to store/organize data to make them ready for future use. One way of organizing data/entries is to add them to a directory to allow quick access to them.

The following three main directories were created in the proposed middleware system.

1. Information directory, filled in advance with instructions for most common emergency cases that might occur in people's day–to–day lives (such as dealing with injuries or burns). These instructions are provided by medical experts to ensure the correct way to deal with each case is given. In addition, information is delivered in different multimedia forms, such as text, video and audio.

2. Professional directory, contains all registered and qualified medical professionals to ensure that all of them are treated equally and their time is saved, as well as getting the best from them. This service is mainly intended to save/update professionals' personal details, including ID, name, contact details, occupation, and current location. This service can be used when a user faces an emergency situation and is accessing the system to look for a professional to discuss his/her concern.

3. Hospital directory, in which all the medical centres, from large hospitals to small GP offices, that are found in small residential areas are included. Each centre should provide its own information, such as name, address, kind of services offered, and availability. These data are kept to be used by the system in the case of an enquiry, such as a user wanting to search for a medical centre in a city that he/she is planning to visit.

7.2.2 Location and direction service

Real–time location awareness is one of the powerful features that any of these types of systems will try to offer. This feature will affect system performance as it requires continuous synchronizing with users' current locations but, on the other hand, it will lead to improvement in all aspects of the system. For example, when a user is searching for doctors near the site of an accident, it would not be realistic to redirect the request to a doctor who had submitted location information many hours earlier or ask the user to provide current location each time he/she seeks a service. Therefore, this is one of the most important services offered because it will be involved in most requests, such as searching for hospitals or doctors, requesting an ambulance, and tracking professionals. The system acquires the locations of all parties (hospitals, professionals, and users) and saves them for future requests and then updates the location regularly as needed, particularly in the case of tracking one of the users, such as a doctor who is on the road or an ambulance on the move.

The other part of this service is providing directions, as the system can provide directions so that the requester (if required) can reach locations such as hospitals or the site of an accident. The system will decide whether to update the location before submitting it or send the most recent location found, depending on when the location was gained and if the requested location was expected to move often.

7.2.3 Temporary storage

Sometimes, the system needs to save information temporarily to do a certain job. When the job is done, the system either deletes this piece of information or saves it in one of the databases.

For example, if the system receives a request from someone who claims he/she is a doctor, the system needs to check whether that person belongs to the Health Department in order to allow/block this person in providing care for the public.

7.2.4 Notification service

When a user requests a service that needs to involve another person's response, the system will notify that person about the requesting service to draw his/her attention. This kind of service is needed to draw another user's attention and redirect jobs to that user.

Simply put, when the system receives a request, it will check who should be assigned the role of dealing with this request; this could be a professional, an ambulance department, etc. Furthermore, the system may need to notify a particular user (or group of users) regarding an event or distribute useful information based on the location of the notified user(s).

Generally, two notification methods were implemented in this thesis: firstly, SMS–based notifications with the help of a Twilio platform that acts as an SMS gateway to the system; and secondly, notifications using a GCM platform because the proposed system uses Android–based mobile devices as users' space.

Using the GCM platform for notification will avoid opening another socket in mobile devices to listen to a new notification from the system, which will negatively affect the battery life. Furthermore, notifications from GCM will reach users even if they are using another application.

However, the GCM notification platform faces some instability. For example, a notification might not be delivered to end users for various reasons, such as the user being offline.

Modifying the GCM platform is not possible. Therefore, the proposed system implemented an extra part on the GCM platform that would improve the stability of the system and ensure notifications are received by users.

The implementation of this extra part was divided into two parts:

1. Before sending the notification to the GCM: (i) create a unique ID for each notification and add it to the notification payload; (ii) set 'time to live' (the default value of this field is 4 weeks [179]) for the notification to 1—3 minutes (depending on the importance of the notification; for example, it will be set to 1 minute if the notification is sent to a medical volunteer regarding an emergence case that needs his/her involvement, whereas it will be set to 3 minutes if it is regarding an update about an emergency event for which the user registered to receive recent updates, as detailed in chapter 6); and (iii) store the notification payload with its ID in the system database and mark it as "pending".

2. After sending the notification and when the notification is received by the user: here the user has to send acknowledgement of having received this notification by attaching the ID of this notification. Once the system receives the ACK request, it will update the record of this notification to "received".

However, if the user does not receive the notification in the predefined time, the same notification will be sent again to the user with a different ID and the previous notification will be marked as "unreceived". This operation will be limited to three times only, to avoid wasting resources. When the third time is applied and the user does not send an acknowledgement of having received the notification, the system assumes that this user is offline, which means the status of this user will be updated, as in chapter 5.

7.2.5 Communication using Audio and Video Streaming Services

Another important service of the middleware system is a service that allow users to establish voice or a video call with someone else such as a doctor. In general, there are different examples of this service, which are listed as follows:

- Delivering voice and video instructions to people in case of emergency.
- Live communication between emergency staff and hospital.
- Communication between professional in case of crowded place scenario.
- Starting a video session between a user and a professional to discuss a concern.

7.2.6 Social media service

This service is connected to a social media network, which is Twitter as presented in chapter 6.

One of the main goals of this service is to help in disaster management by collecting on–the–ground data and helping management centres to understand an event more clearly.

However, this service can also help ordinary people to see recent updates regarding an event and use a social media network as a communication platform to seek help from the system and establish communication with someone, such as a rescuer, to explain the current situation to him/her.

7.2.7 MANET management service

The idea of this MANET service is to assist in situations in which a large number of users are present in the same place (perhaps hundreds of thousands) at the same time.

Therefore, the ability to create a MANET will lead to improvement in managing crowds, as well as providing healthcare to most users who are present in these places. Some of the features that can be gained from this service are:

Using a neighbour's link: this is feasible due to the number of users connected to the system. Hence, one of the users in such a MANET can use another user's link to reach the system's services by offering a more stable and larger bandwidth.

Sharing knowledge: if one of the MANET's members retrieves a piece of information from the cloud and another member is looking for the same information, sharing it will be done locally without having to connect to the system again.

Live communication: the MANET will, at all times, allow its members to communicate directly without having to request this from the system. To ensure that users will benefit from this service, the system has the role of managing this service, taking into account the following:

- Creating a MANET that is not used in an attack.
- Monitoring the usage of each MANET to get some information back, such as the number of links needed and allocating/deallocating resources.

- Checking the existence of each MANET and ensuring that members are active.

Saving active sessions: another advantage of this service is that it allows users to save interrupted sessions in case a break occurs or the sudden departure of one of the corresponding parties. The session can then be resumed from the same point that was reached before the break. Session details and data are needed to accomplish this service. In addition, both of the users involved in the session have to provide their details to be used in this service.

7.3 Meeting Pre–defined Requirements

After presenting the reference architecture for the middleware system and the services found in the system, this section revisits the performance requirements that were defined in chapter 2.

Based on the research work of this thesis, as presented in this chapter, and the experimental results and evaluation in chapters 5 and 6, respectively, the following discussion assesses whether the work in this thesis meets these requirements. The requirements can be grouped as follows.

1. Providing reliable connections
When considering the usage of mobile devices and MANET–type networks, it is recognized that the possibility of disconnections or breaks in links is ever present. However, by using the proposed middleware system, the reliability level can be enhanced in two ways: first, by allowing users to create or join a MANET network to reach the services that are provided and hosted in the cloud.

Therefore, users can use other members' links to reach these services if no direct link to the cloud is detected. Similarly, the cloud can reach those who do not have a direct link to the cloud through their neighbours or the cloud agent (as shown in chapter 5); and second, by allowing users to save the progress of active sessions in the cloud to resume them if a break occurs. This will improve the robustness and reliability of the system, as the part of the session that has been executed will not be repeated, which means that no resources will be wasted.

2. Ensuring a reasonable level of security and privacy
Introducing a mobile cloud will raise security and privacy concerns, such as where and how data are stored and who accesses them.

Furthermore, working within healthcare will require more attention in terms of privacy, as the system will be dealing with sensitive data such as individuals' medical status.

This research work did not, in general, deal with the lack of security and privacy when introducing a mobile cloud to an m–health system. However, in the thesis, security enactments were deployed and tested to ensure that a high level of protection is delivered. For instance, cloud services can only be accessed by users who have an active cloud account.

Furthermore, a two–tier authentication process was introduced, whereby users have to complete a login to their account by providing a valid user name and password. Then, the system sends an SMS message to a pre–defined mobile phone number containing a unique code that can only be used within a short space of time, which means that no access is allowed if this code is not entered correctly. Existing data encryption solutions and secure connection protocols were also considered and discussed in chapters 5 and 6.

3. Responding to emergencies has to be achieved quickly

One of the most important requirements is minimizing the time needed to respond to people who are in an emergency situation because any delay in responding will affect the medical emergency case and could result in loss of life.

In this thesis, two mobile cloud services were designed and evaluated with the aim of responding to an emergency case in a matter of seconds in order to increase the chance of survival. These services also take advantage of a MANET–cloud model, which means that users can create/join a MANET to seek help and receive a swift response.

However, these two services could be essential in some scenarios, such as in crowded places (e.g., Al–Hajj) where the arrival of traditional emergency services (e.g., ambulances) can be delayed. A medical professional could be present in these areas and provide a swift response to any emergency case — which is exactly what this mobile cloud service is intended to do.

4. Allowing collaboration and dissemination

Dealing with people's health involves the need to access medical data, such as medical history or any item of information that could affect the decisions that will be taken regarding someone's health when there is an emergency situation, such as allergic reactions to certain medicines.

Furthermore, in some situations, collaboration between the medical professionals themselves is needed to deliver appropriate care to someone who is in an emergency situation.

For example, an ambulance crew might need to speak to an expert to confirm a course of action or a doctor might want to contact an expert who has a speciality in a certain medical area in order to discuss an emergency case.

In this research, collaboration is enabled between the parties who are involved in delivering healthcare and support to people in emergency situations, whereby professionals can create a new MANET network or join an existed one to share ideas and information locally regarding an emergency case.

Furthermore, a notification service is provided as part of the proposed middleware system, in which a user who has expertise in a medical field can be notified using this service.

With regard to the dissemination of useful data in emergency situations, the proposed middleware system provides two phases to enhance data dissemination: (1) using the MANET–cloud model whereby the cloud can push useful information to as large a number of users as possible through cloud agents who are responsible for delivering these messages to other members, particularly those who do not have an active link. Furthermore, the members of a MANET can broadcast feeds/news locally regarding an event to all known neighbours; and (2) by using a social media service that is part of the proposed middleware system with the aim of making the best use of a social media network to help users who are involved in a disaster and rescuers who are managing and delivering support.

This research chose to use Twitter because of its popularity and suitability to the research scope. As shown in chapter 6, using this service can build a communication platform between ordinary people who are facing issues because of a disaster that has occurred suddenly and the rescuers or risk and management centres who are responsible for taking care of the situation.

Users can submit the most recent news/updates and valuable on–the–ground information regarding an event and rescuers can read these data to gain a better understating of the event as well as help in deciding the actions to be taken.

7.4 Chapter Summary

This chapter presented the reference architecture with all the services that form the middleware system. Then, each service was discussed in detail, together with the benefits that can be offered to system users. One of these is the MANET–cloud model services, which play an important role in the research work by providing the ability to create a peer–to–peer network such as a MANET or join existing networks whereby all of the MANETs are managed and monitored by the cloud.

Shifting the management of MANET networks to the cloud reduces some of the known issues regarding this type of network, such as the sudden departure of its members. However, this service is considered an infrastructure service to the other services for the purpose of enhancing system reliability or even system connectivity, for example by allowing direct/local communication between two users.

Finally, the chapter recalled all the requirements that were defined in chapter 2 to determine if the work that has been done in this thesis meets this set of requirements. The requirements were discussed in terms of reliability and robustness, security and privacy, response time, and data dissemination. Detailed discussions of the experimental results and evaluation work from chapters 5 and 6, respectively, were provided to demonstrate that the proposed middleware system meets these requirements.

In summary, the middleware system is feasible and can enhance system reliability and robustness, as well as allowing the wide dissemination of data in emergency situations and collaboration between all the parties involved in an emergency event or situation. Although this middleware does not deal directly with the lack of security or the privacy aspects of introducing a mobile cloud to the m–health field, a number of mechanisms were discussed and deployed to ensure a reasonable level of protection of users' data.

The next chapter provides the conclusions regarding this research work and emphasizes the scientific contributions made by this thesis. Furthermore, the chapter presents a discussion of possible improvements to the proposed middleware system and future directions for this research work.

Chapter 8

Conclusions and Future Work

8.1 Thesis Conclusions

Mobilizing a healthcare service can benefit both users who are seeking medical help and the professionals who can provide it. One of the most important advantages lies in reducing the high cost that results from providing healthcare services to the public, which is a critical issue in the health field and particularly in developing countries.

Therefore, the idea here is to provide healthcare support on the move, which means these services will be provided on mobile devices (such as smartphones, tablets, and sensors). Today, mobile devices have a number of features that can enhance healthcare services, such as portability and built–in sensors. However, mobile devices also face some limitations, such as limited storage capability and battery life.

Therefore, a mobile cloud computing solution is provided here to deal with some of these limitations. The main idea behind the mobile cloud is to deploy all services on the cloud and make them accessible over the Internet by mobile users. Here, the mobile devices will not be involved in processing tasks, and thus only the results are provided when the task is executed, for the purpose of minimizing the usage of mobile device resources such as battery life. For instance, in obtaining directions to the nearest hospital, the cloud will take on the role of collecting the requesting user's location and then finding the best match with a hospital to deal with the user's case. However, deploying an MCC solution is not straightforward, as it faces some issues such as a lack of security and privacy.

In addition to the above limitations, the integration of a mobile cloud into the m–health field raises another set of challenges, such as the sensitivity of medical data. Furthermore, some cases, such as emergencies, can happen to people when they are on the move, so the provision of healthcare services in the shortest possible period of time can be life–saving. A system that can support such a service requires unique features, such as availability, robustness, access to web/cloud information through different networking technologies, and the efficient management of mobile users.

In this research work, a new middleware system prototype was designed that consists of two main components:

- Two main services were developed that are hosted in the cloud and accessible via mobile devices. Both services are aimed at providing medical support to users who need it in emergency situations. The main difference is that the first service aims at responding swiftly to emergency cases, while the second consumes a social media stream (Twitter was used in this research work) to allow the early detection of emergencies that might be considered large–scale events, such as sudden flooding in a city.
- As connectivity between users and the health service system is a key feature, this thesis introduced a MANET management model that is set between users and the cloud to allow mobile users to be reached, including those without cellular connectivity but who do have access to Wi–Fi.

Then, as part of the proposed system, at the top of this middleware, a number of applications/scenarios that might occur in people's day–to–day lives were defined, such as being in a crowded area.

The proposed system was tested after deploying it on an Amazon EC2 instance and Android–based mobile devices, which showed that it is feasible and:

- Referred to reliable connections by using neighbouring links to access cloud services, as well as the cloud being able to reach users via MANET members' links (e.g., via a cloud agent).
- Improved the system availability by deploying an SMS gateway in the cloud to allow SMS messaging between users and the cloud.
- Enhanced the dissemination of user details or medical data as well as any useful information, either inside MANET networks or between the user who requests the aid and the professional who provides it.

- Eased collaboration between all parties, such as merging two (or more) MANETs together to allow more people to interact with the emergency case and share useful advice.

With regard to security and privacy matters, security techniques were implemented and tested, such as an SMS verification scheme and a request checkout framework.

In summary, a MANET–cloud model is one of the important parts of this research work and brings significant benefits such as allowing local communication and the sharing of medical data. In addition, managing this type of network in the cloud will not only address its known limitations, such as the sudden departure of its members and the complexity of its management, but will also allow the cloud to reach users who do not have a direct link to the cloud using neighbouring links. Furthermore, this model can act as an infrastructure service in the proposed middleware system to support other services, such as a directory service that allows users to search for a medically trained professional.

8.2 Future Work Directions

The results and analysis from the evaluations that have been performed have shown that the proposed system could deliver healthcare services to people in emergencies using the concept of point–of–care. Furthermore, the proposed middleware (that is part of the proposed system) has shown that the connectivity and reliability of the whole system could be improved. However, the work still presents limitations and challenges that could be addressed in future work with further research and evaluation, which can be summarized as follows.

Including several mobile OS

The proposed system currently only uses the Android OS mobile platform as users' space. This could easily be extended to several platforms, such as the Windows Phone and Apple iOS. However, the most difficult part might be performing ad–hoc networking on these platforms. As seen in this thesis, forcing mobile devices to switch to ad–hoc mode requires superuser access (also known as root access) to allow a modification to the Wi–Fi chip from regular/normal mode to ad–hoc mode. Therefore, research into and the study of possible ways to perform ad–hoc networking in these OSs (e.g., Apple iOS and Windows Phone) should be considered and discussed.

The most important benefit that could be gained from porting the users' space from the proposed system to several mobile platforms is not limited to users who access the system using Android–based mobile devices. In other words, serving as wide a number of users as possible without being restricted to the type of mobile device or the platform running in these devices.

The proposed system uses the GCM notification platform that is responsible for delivering notifications from the system to the users who access the system via Android–based mobile devices.

Here, porting the user space on other platforms such as an iOS or Windows Phone requires using each one's own notification platform. This means the system has to have a controller that is responsible for determining to which platform the notification should be sent based on the OS on each user's device.

However, another solution could be to design and implement a self–notification system (i.e., that does not rely on a shelf–ready one such as GCM) that can deliver notifications to all kinds of mobile OS platforms. See [180] for further reading about what is known as cross–platform notification.

Security enhancement

As mentioned previously, this thesis has not focused on the security and privacy aspects of the proposed system. However, a number of authentication and authorization mechanisms were implemented onto the proposed system, such as an SMS verification scheme (which asks the user to type in a unique code that has already been sent to his/her mobile phone number to complete a login process), a request validation scheme (as the user who seeks help from the system has to retype a random code that is shown on the mobile device screen before sending a request to the cloud to ensure the requesting user is real), and distributing security keys, whereby two users perform a session to accomplish a particular job (e.g., sharing a map that shows the affected and safe areas of a city that has been flooded) and the two keys are used to retrieve the session data or resume its work if a break occurs.

However, more mechanisms could be deployed onto the system, such as encrypting users' medical data that are stored in the cloud. Furthermore, a strong security mechanisms could be deployed for the data sharing between users inside a MANET or between users and the cloud to ensure that data are protected and communications are safe. For instance, introducing strong secure communication protocols, such as [181].

Implementing more services

In this thesis, two mobile cloud services were implemented: one mainly to help users in emergencies as quickly as possible by allowing users who are in an emergency situation to look up medical volunteers near their location who could help them with their medical needs; the second service takes advantage of a social media network (Twitter in this case) to achieve the early detection of large–scale disasters, such as an earthquake or flood, as well as updating/notifying public users regarding an active risk in their location.

However, the system could benefit from planning the design of more cloud–based services that are aimed at helping people in emergencies, along with enhancing the delivery of healthcare services to mobile devices.

An example of a service could be one that interacts with another type of social media network, such as Facebook or Google Plus, to collect information regarding a disaster and could also be used to communicate with people who need help.

Another example could be a service that works closely with wearable medical devices, in which all data are collected and analysed using one of the existing cloud-based frameworks, such as [182], in order to monitor patients' medical status in order to detect any abnormalities and report them to the appropriate emergency department.

Further experiments and evaluations

The proposed system was tested and evaluated in a laboratory environment with two main metrics: time (delay) and energy (power consumption). However, to determine the benefit of introducing a mobile cloud solution, the services could be deployed directly on mobile devices and the results (power consumption and network throughput) could then be compared with the results of hosting the services in the cloud and accessing them via mobile devices.

Furthermore, the proposed system could also be tested in a real–world scenario (e.g., during Al–Hajj) to report how the system behaves when there is a high number of users. Other performance metrics could also be introduced here, such as scalability and bandwidth usage, to evaluate the validity of the system.

Bibliography

[1] "The advent of digital health." Available online at: http://www.strategy-business.com/blog/The-Advent-of-Digital-Health?rssid=healthcare&gko=f2f63.

[2] E. MacIntosh, N. Rajakulendran, Z. Khayat, and A. Wise, "Transforming health: Shifting from reactive to proactive and predictive care," 2014. Available online at: https://www.marsdd.com/news-and-insights/transforming-health-decentralized-connected-care/.

[3] "Ims institute for healthcare informatics." Available online at: http://www.imedicalapps.com/2015/09/ims-health-apps-report/.

[4] "Is mobile healthcare the future?." Available online at: http://www.greatcall.com/greatcall/lp/is-mobile-healthcare-the-future-infographic.aspx.

[5] A. Garcia, A. Pomykala, and S. Siegel, "Us health care is moving upstream.," *Health progress (Saint Louis, Mo.)*, vol. 94, no. 1, pp. 6–13, 2012.

[6] "Philips." Available online at: http://www.usa.philips.com/healthcare/about/events-calendar/sxsw-2015#hackathon.

[7] A. Gulavani and M. Shinde, "Occupational stress and job satisfaction among nurses," *International Journal of Science and Research (IJSR)*, vol. 3, no. 4, pp. 733–740, 2014.

[8] H. N. Alshareef, "Introducing mobile cloud technology into m-health to deliver better care/support in case of emergencies," *The Boolean*, pp. 20–26, 2015.

[9] "The boolean.." Available online at: http://publish.ucc.ie/boolean/home.

[10] H. Alshareef and D. Grigoras, “Mobile ad-hoc network management in the cloud,” in *Parallel and Distributed Computing (ISPDC), 2014 IEEE 13th International Symposium on*, pp. 140–147, IEEE, 2014.

[11] H. Alshareef and D. Grigoras, “Robust cloud management of manet checkpoint sessions,” in *Parallel and Distributed Computing (ISPDC), 2015 14th International Symposium on*, pp. 66–73, IEEE, 2015.

[12] H. Alshareef and D. Grigoras, “Robust cloud management of manet checkpoint sessions,” *Concurrency and Computation: Practice and Experience*, pp. n/a–n/a, 2016. CPE-15-0380.

[13] H. Alshareef and D. Grigoras, “First responder help facilitated by the mobile cloud,” in *Cloud Technologies and Applications (CloudTech), 2015 International Conference on*, pp. 1–8, IEEE, 2015.

[14] D. Johnson and D. Maltz, *Mobile computing*. Kluwer academic publishers Dordrecht, 1996.

[15] O. M. C. Rendon, F. O. M. Pabón, M. J. G. Vargas, and J. A. H. Guaca, “Architectures for web services access from mobile devices,” in *Third Latin American Web Congress (LA-WEB’2005)*, pp. 93–97, IEEE, 2005.

[16] A. Battestini, C. Del Rosso, A. Flanagan, and M. Miettinen, “Creating next generation applications and services for mobile devices: Challenges and opportunities,” in *2007 IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1–4, IEEE, 2007.

[17] X. Zhang, A. Kunjithapatham, S. Jeong, and S. Gibbs, “Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing,” *Mobile Networks and Applications*, vol. 16, no. 3, pp. 270–284, 2011.

[18] J. W. Rittinghouse and J. F. Ransome, *Cloud computing: implementation, management, and security*. CRC press, 2016.

[19] X. Xu, “From cloud computing to cloud manufacturing,” *Robotics and computer-integrated manufacturing*, vol. 28, no. 1, pp. 75–86, 2012.

[20] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, “A break in the clouds: towards a cloud definition,” *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 50–55, 2008.

[21] P. Mell and T. Grance, “The nist definition of cloud computing,” 2011.

[22] D. Puthal, B. Sahoo, S. Mishra, and S. Swain, “Cloud computing features, issues, and challenges: a big picture,” in *Computational Intelligence and Networks (CINE), 2015 International Conference on*, pp. 116–123, IEEE, 2015.

[23] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, *et al.*, “Above the clouds: A berkeley view of cloud computing,” 2009.

[24] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, *et al.*, “A view of cloud computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[25] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing: state-of-the-art and research challenges,” *Journal of internet services and applications*, vol. 1, no. 1, pp. 7–18, 2010.

[26] Y. Jadeja and K. Modi, “Cloud computing-concepts, architecture and challenges,” in *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, pp. 877–880, IEEE, 2012.

[27] S. Abolfazli, Z. Sanaei, M. Sanaei, M. Shojafar, and A. Gani, “Mobile cloud computing: The-state-of-the-art, challenges, and future research,” *Encyclopedia of Cloud Computing, Wiley, USA*, 2015.

[28] M. J. O’Sullivan and D. Grigoras, “The mobile cloud-more than a cloud,” *eChallenges 2013, Dublin, Ireland, 9-11 Oct 2013*, 2013.

[29] D. Kovachev, Y. Cao, and R. Klamma, “Mobile cloud computing: a comparison of application models,” *arXiv preprint arXiv:1107.4940*, 2011.

[30] P. Hazarika, V. Baliga, and S. Tolety, “The mobile-cloud computing (mcc) roadblocks,” in *2014 Eleventh International Conference on Wireless and Optical Communications Networks (WOCN)*, pp. 1–5, IEEE, 2014.

[31] A. Tuli, N. Hasteer, M. Sharma, and A. Bansal, “Exploring challenges in mobile cloud computing: An overview,” in *Confluence 2013: The Next Generation Information Technology Summit (4th International Conference)*, pp. 496–501, IET, 2013.

[32] X. Fan, J. Cao, and H. Mao, “A survey of mobile cloud computing,” *ZTE Corporation*, 2011.

[33] P. Kulkarni and R. Khanai, "Addressing mobile cloud computing security issues: a survey," in *Communications and Signal Processing (ICCSP), 2015 International Conference on*, pp. 1463–1467, IEEE, 2015.

[34] B. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud," in *Proceedings of the sixth conference on Computer systems*, pp. 301–314, ACM, 2011.

[35] E. E. Marinelli, "Hyrax: cloud computing on mobile devices using mapreduce," tech. rep., DTIC Document, 2009.

[36] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, 2009.

[37] "Apache™ hadoop®." Available online at: `http://hadoop.apache.org/`.

[38] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[39] T. Verbelen, P. Simoens, F. De Turck, and B. Dhoedt, "Cloudlets: bringing the cloud to the mobile user," in *Proceedings of the third ACM workshop on Mobile cloud computing and services*, pp. 29–36, ACM, 2012.

[40] A. Artail, K. Frenn, H. Safa, and H. Artail, "A framework of mobile cloudlet centers based on the use of mobile devices as cloudlets," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, pp. 777–784, IEEE, 2015.

[41] Q. Xia, W. Liang, and W. Xu, "Throughput maximization for online request admissions in mobile cloudlets," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, pp. 589–596, IEEE, 2013.

[42] M. Quwaider and Y. Jararweh, "Cloudlet-based for big data collection in body area networks," in *Internet Technology and Secured Transactions (ICITST), 2013 8th International Conference for*, pp. 137–141, IEEE, 2013.

[43] S. C. Mukhopadhyay, "Wearable sensors for human activity monitoring: A review," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1321–1330, 2015.

[44] B. M. Silva, J. J. Rodrigues, I. de la Torre Díez, M. López-Coronado, and K. Saleem, "Mobile-health: a review of current state in 2015," *Journal of biomedical informatics*, vol. 56, pp. 265–272, 2015.

[45] R. S. Istepanian, E. Jovanov, and Y. Zhang, "Guest editorial introduction to the special section on m-health: Beyond seamless mobility and global wireless health-care connectivity," *IEEE Transactions on information technology in biomedicine*, vol. 8, no. 4, pp. 405–414, 2004.

[46] S. Akter, P. Ray, *et al.*, "mhealth-an ultimate platform to serve the unserved," *Yearb Med Inform*, vol. 2010, pp. 94–100, 2010.

[47] R. S. Istepanian and J. C. Lacal, "Emerging mobile communication technologies for health: some imperative notes on m-health," in *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*, vol. 2, pp. 1414–1416, IEEE, 2003.

[48] P. N. Mechael, "The case for mhealth in developing countries," *innovations*, vol. 4, no. 1, pp. 103–118, 2009.

[49] J. E. Bardram and H. B. Christensen, "Pervasive computing support for hospitals," *IEEE Pervasive Computing*.

[50] S. Akter, J. D'Ambra, and P. Ray, "User perceived service quality of m-health services in developing countries," 2010.

[51] M. Kay, J. Santos, and M. Takane, "mhealth: New horizons for health through mobile technologies," *World Health Organization*, vol. 64, no. 7, pp. 66–71, 2011.

[52] "Vital wave consulting, mhealth for development: The opportunity of mobile technology for healthcare in developing world." Available online at: `http:// www.vitalwaveconsulting.com/   insights/ mHealth.htm`.

[53] A. v. Heerden, M. Tomlinson, and L. Swartz, "Point of care in your pocket: a research agenda for the field of m-health," *Bulletin of the World Health Organization*, vol. 90, no. 5, pp. 393–394, 2012.

[54] E. Ozdalga, A. Ozdalga, and N. Ahuja, "The smartphone in medicine: a review of current and potential use among physicians and students," *Journal of medical Internet research*, vol. 14, no. 5, p. e128, 2012.

[55] S. Kumar, W. J. Nilsen, A. Abernethy, A. Atienza, K. Patrick, M. Pavel, W. T. Riley, A. Shar, B. Spring, D. Spruijt-Metz, *et al.*, "Mobile health technology evaluation: the mhealth evidence workshop," *American journal of preventive medicine*, vol. 45, no. 2, pp. 228–236, 2013.

[56] "British red cross." Available online at `http://www.redcross.org.uk/`.

[57] "Help me – emergency siren device with auto sms help message." Available online at `http://www.skyant.com/`.

[58] N. Matias and M. J. Sousa, "Mobile health as a tool for behaviour change in chronic disease prevention: A systematic literature review," in *2016 11th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 1–6, June 2016.

[59] Z. Romano and S. Cangiano, "Open sourcing wearables," in *Empowering Users through Design*, pp. 153–175, Springer, 2015.

[60] B. Latré, B. Braem, I. Moerman, C. Blondia, and P. Demeester, "A survey on wireless body area networks," *Wireless Networks*, vol. 17, no. 1, pp. 1–18, 2011.

[61] M. E. Rosenberger, M. P. Buman, W. L. Haskell, M. V. McConnell, and L. L. Carstensen, "Twenty-four hours of sleep, sedentary behavior, and physical activity with nine wearable devices.," *Medicine and science in sports and exercise*, vol. 48, no. 3, pp. 457–465, 2016.

[62] H. Zhang, K. Liu, W. Kong, F. Tian, Y. Yang, C. Feng, T. Wang, and Q. Chen, "A mobile health solution for chronic disease management at retail pharmacy," in *e-Health Networking, Applications and Services (Healthcom), 2016 IEEE 18th International Conference on*, pp. 1–5, IEEE, 2016.

[63] S. A. Hameed, V. Miho, W. AlKhateeb, and A. Hassan, "Medical emergency and healthcare model: Enhancemet with sms and mms facilities," in *Computer and Communication Engineering (ICCCE), 2010 International Conference on*, pp. 1–6, IEEE, 2010.

[64] M. Mohandes, "Pilgrim tracking and identification using the mobile phone," in *Consumer Electronics (ISCE), 2011 IEEE 15th International Symposium on*, pp. 196–199, IEEE, 2011.

[65] S. El-Masri, "Mobile comprehensive emergency system," *Handbook of Research in Mobile Business: Technical, Methodological, and Social Perspectives: Technical, Methodological, and Social Perspectives*, vol. 1, p. 106, 2006.

[66] M. B. Greer Jr and J. W. Ngo, "Personal emergency preparedness plan (pepp) facebook app: using cloud computing, mobile technology, and social networking services to decompress traditional channels of communica-

tion during emergencies and disasters," in *Services Computing (SCC), 2012 IEEE Ninth International Conference on*, pp. 494–498, IEEE, 2012.

[67] "Clinicloud." Available online at https://clinicloud.com.

[68] "doctor on demand." Available online at http://www.doctorondemand.com.

[69] Y. Du, Y. Chen, D. Wang, J. Liu, and Y. Lu, "An android-based emergency alarm and healthcare management system," in *IT in Medicine and Education (ITME), 2011 International Symposium on*, vol. 1, pp. 375–379, IEEE, 2011.

[70] K. Mitra, C. Ahlund, *et al.*, "A mobile cloud computing system for emergency management," *Cloud Computing, IEEE*, vol. 1, no. 4, pp. 30–38, 2014.

[71] M. Muthaiyan, N. Goel, and D. S. Prakash, "Virtual e-medic: A cloud based medical aid," in *Proceedings of World Academy of Science, Engineering and Technology*, no. 71, p. 1344, World Academy of Science, Engineering and Technology (WASET), 2012.

[72] K. Ma and Z. Tang, "An online social mutual help architecture for multi-tenant mobile clouds," *International Journal of Intelligent Information and Database Systems*, vol. 8, no. 4, pp. 359–374, 2014.

[73] P. J. Medina, G. J. Villarruel, and T. B. Corona, "Proposal for an m-health system," in *Electronics, Robotics and Automotive Mechanics Conference, 2009. CERMA'09.*, pp. 55–59, IEEE, 2009.

[74] H. Jemal, Z. Kechaou, M. B. Ayed, and A. M. Alimi, "Mobile cloud computing in healthcare system," in *Computational Collective Intelligence*, pp. 408–417, Springer, 2015.

[75] M. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing," in *INFOCOM, 2013 Proceedings IEEE*, pp. 1285–1293, IEEE, 2013.

[76] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, "Mobile cloud computing: A survey, state of art and future directions," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 133–143, 2014.

[77] D. Kopec, M. Kabir, D. Reinharth, O. Rothschild, and J. Castiglione, "Human errors in medical practice: systematic classification and reduction with automated information systems," *Journal of medical systems*, vol. 27, no. 4, pp. 297–313, 2003.

[78] C. Araujo, F. Silva, I. Costa, F. Vaz, S. Kosta, and P. Maciel, "Supporting availability evaluation in mcc-based mhealth planning," *Electronics Letters*, vol. 52, no. 20, pp. 1663–1665, 2016.

[79] H. H. Chang, P. B. Chou, and S. Ramakrishnan, "An ecosystem approach for healthcare services cloud," in *e-Business Engineering, 2009. ICEBE'09. IEEE International Conference on*, pp. 608–612, IEEE, 2009.

[80] M. Bamiah, S. Brohi, S. Chuprat, and J. Ab Manan, "A study on significance of adopting cloud computing paradigm in healthcare sector," in *Cloud Computing Technologies, Applications and Management (ICCCTAM), 2012 International Conference on*, pp. 65–68, IEEE, 2012.

[81] J. Agarkhed, S. Mundewadi, S. S. Patil, *et al.*, "Mobile health monitoring system using cloud computing," in *Wireless Communications, Signal Processing and Networking (WiSPNET), International Conference on*, pp. 1301–1305, IEEE, 2016.

[82] C. Low and Y. H. Chen, "Criteria for the evaluation of a cloud-based hospital information system outsourcing provider," *Journal of medical systems*, vol. 36, no. 6, pp. 3543–3553, 2012.

[83] M. M. Singh and J. K. Mandal, "Reliability analysis of mobile ad hoc network," in *Computational Intelligence and Communication Networks (CICN), 2015 International Conference on*, pp. 161–164, IEEE, 2015.

[84] A. Malathi and D. Radha, "Analysis and visualization of social media networks," in *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, pp. 58–63, Oct 2016.

[85] D. Grigoras and M. Riordan, "Cost-effective mobile ad hoc networks management," *Future Generation Computer Systems*, vol. 23, no. 8, pp. 990–996, 2007.

[86] S. Nesargi and R. Prakash, "Manetconf: Configuration of hosts in a mobile ad hoc network," in *INFOCOM 2002. Twenty-First Annual Joint Confer-*

ence of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol. 2, pp. 1059–1068, IEEE, 2002.

[87] M. Conti and S. Giordano, “Mobile ad hoc networking: milestones, challenges, and new research directions,” *Communications Magazine, IEEE*, vol. 52, no. 1, pp. 85–96, 2014.

[88] L. Zhang, X. Ding, Z. Wan, M. Gu, and X. Li, “Wiface: a secure geosocial networking system using wifi-based multi-hop manet,” in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*, p. 3, ACM, 2010.

[89] D. Huang, X. Zhang, M. Kang, and J. Luo, “Mobicloud: building secure cloud framework for mobile computing and communication,” pp. 27–34, 2010.

[90] “Wi-fi alliance, wi-fi direct,” Available online at `http://www.wi-fi.org/discover-wi-fi/wi-fi-direct`.

[91] “Softap - wikipedia, the free encyclopedia.” Available online at `http://en.wikipedia.org/wiki/SoftAP`.

[92] “Welcome to the freedom of portable wi-fi,” *Intel® My Wi-Fi Technology: Synch, Share, Show & Print on the Go*, 2008.

[93] P. Gardner-Stephen, “The serval project: Practical wireless ad-hoc mobile telecommunications,” *Flinders University, Adelaide, South Australia, Tech. Rep*, 2011.

[94] H. Sinnreich and A. B. Johnston, *Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol*, vol. 27. John Wiley & Sons, 2012.

[95] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, “Optimized link state routing protocol for ad hoc networks,” in *Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings. IEEE International*, pp. 62–68, IEEE, 2001.

[96] “The serval mesh - android apps on google play.” Available online at `https://play.google.com/store/apps/details?id=org.servalproject`.

[97] “Manet manager - android apps on google play.” Available online at `https://play.google.com/store/apps/details?id=org.span`.

[98] J. Thomas and J. Robble, “Off grid communications with android: Meshing the mobile world,” *Proc. IEEE Conference on Technologies for Homeland Security, Waltham, MA*, pp. 401–405, 2012.

[99] A. Hinds, M. Ngulube, S. Zhu, and H. Al-Aqrabi, “A review of routing protocols for mobile ad-hoc networks (manet),” *International Journal of Information and Education Technology*, vol. 3, no. 1, p. 1, 2013.

[100] A. Khunteta and P. Sharma, “A survey of checkpointing algorithms in mobile ad hoc network,” *Global Journal of Computer Science and Technology*, vol. 12, no. 12-E, 2012.

[101] C. Men, Z. Xu, and X. Li, “An efficient checkpointing and rollback recovery scheme for cluster-based multi-channel ad hoc wireless networks,” in *Parallel and Distributed Processing with Applications, 2008. ISPA’08. International Symposium on*, pp. 371–378, IEEE, 2008.

[102] M. Ono and H. Higaki, “Consistent checkpoint protocol for wireless ad-hoc networks.,” in *PDPTA*, pp. 1041–1046, 2007.

[103] A. J. Goldsmith and S. B. Wicker, “Design challenges for energy-constrained ad hoc wireless networks,” *Wireless Communications, IEEE*, vol. 9, no. 4, pp. 8–27, 2002.

[104] B. Nicolae and F. Cappello, “Blobcr: efficient checkpoint-restart for hpc applications on iaas clouds using virtual disk image snapshots,” in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, p. 34, ACM, 2011.

[105] E. N. Elnozahy, L. Alvisi, Y. Wang, and D. B. Johnson, “A survey of rollback-recovery protocols in message-passing systems,” *ACM Computing Surveys (CSUR)*, vol. 34, no. 3, pp. 375–408, 2002.

[106] R. Handorean, R. Sen, G. Hackmann, and G. Roman, “Context aware session management for services in ad hoc networks,” in *Services Computing, 2005 IEEE International Conference on*, vol. 1, pp. 113–120, IEEE, 2005.

[107] S. Wu, S. Ni, J. Sheu, and Y. Tseng, “Route maintenance in a wireless mobile ad hoc network,” *Telecommunication Systems*, vol. 18, no. 1-3, pp. 61–84, 2001.

[108] P. Khamrui and K. Majumder, “A trusted node based checkpointing scheme for mobile ad-hoc networks (manets),” in *Electronics and Communication*

Systems (ICECS), 2015 2nd International Conference on, pp. 831–836, IEEE, 2015.

[109] P. K. Jaggi and A. K. Singh, “Opportunistic rollback recovery in mobile ad hoc networks,” *Advance Computing Conference (IACC)*, pp. 860–865, 2014.

[110] P. K. Jaggi and A. K. Singh, “Preventing useless checkpoints in manets,” in *Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on*, pp. 533–538, IEEE, 2012.

[111] H. Tsai, T. Chen, and C. Chu, “An on-demand routing protocol with backtracking for mobile ad hoc networks,” in *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, vol. 3, pp. 1557–1562, IEEE, 2004.

[112] P. K. Jaggi and A. K. Singh, “Staggered checkpointing and recovery in cluster based mobile ad hoc networks,” pp. 122–134, 2011.

[113] J. Li, Q. Li, S. U. Khan, and N. Ghani, “Community-based cloud for emergency management,” in *System of Systems Engineering (SoSE), 2011 6th International Conference on*, pp. 55–60, IEEE, 2011.

[114] S. Luna and M. Pennock, “Social media in emergency management advances, challenges and future directions,” in *Systems Conference (SysCon), 2015 9th Annual IEEE International*, pp. 792–797, IEEE, 2015.

[115] A. Mills, R. Chen, J. Lee, and H. Raghav Rao, “Web 2.0 emergency applications: how useful can twitter be for emergency response?,” *Journal of Information Privacy and Security*, vol. 5, no. 3, pp. 3–26, 2009.

[116] B. R. Lindsay, “Social media and disasters: Current uses, future options, and policy considerations,” 2011.

[117] M. Ganea, A. Constantin, A. D. Comsa, *et al.*, “Quick overview of the social media,” *Romanian Statistical Review Supplement*, vol. 60, no. 3, pp. 113–115, 2012.

[118] M. Imran, C. Castillo, F. Diaz, and S. Vieweg, “Processing social media messages in mass emergency: a survey,” *ACM Computing Surveys (CSUR)*, vol. 47, no. 4, p. 67, 2015.

[119] R. González-Ibánez, S. Muresan, and N. Wacholder, “Identifying sarcasm in twitter: a closer look,” in *Proceedings of the 49th Annual Meeting of the*

Association for Computational Linguistics: Human Language Technologies: short papers, pp. 581–586, Association for Computational Linguistics, 2011.

[120] J. Yin, A. Lampert, M. Cameron, B. Robinson, and R. Power, "Using social media to enhance emergency situation awareness," *IEEE Intelligent Systems*, vol. 27, no. 6, pp. 52–59, 2012.

[121] A. J. Lazard, E. Scheinfeld, J. M. Bernhardt, G. B. Wilcox, and M. Suran, "Detecting themes of public concern: A text mining analysis of the centers for disease control and prevention's ebola live twitter chat," *American journal of infection control*, vol. 43, no. 10, pp. 1109–1111, 2015.

[122] G. Chakraborty, M. Pagolu, and S. Garla, *Text mining and analysis: Practical methods, examples, and case studies using SAS.* SAS Institute, 2014.

[123] N. R. Aljohani, S. A. Alahmari, and A. M. Aseere, "An organized collaborative work using twitter in flood disaster," 2011.

[124] K. Grolinger, *Disaster Data Management in Cloud Environments.* PhD thesis, The University of Western Ontario, 2013.

[125] M. Moi, T. Friberg, R. Marterer, C. Reuter, T. Ludwig, D. Markham, M. Hewlett, and A. Muddiman, "Strategy for processing and analyzing social media data streams in emergencies," in *2015 2nd International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)*, pp. 42–48, IEEE, 2015.

[126] T. H. Nazer, F. Morstatter, H. Dani, and H. Liu, "Finding requests in social media for disaster relief," in *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pp. 1410–1413, IEEE, 2016.

[127] L. Zeng, K. Starbird, and E. S. Spiro, "Rumors at the speed of light? modeling the rate of rumor transmission during crisis," in *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pp. 1969–1978, IEEE, 2016.

[128] S. H. Almotiri, M. A. Khan, and M. A. Alghamdi, "Mobile health (m-health) system in the context of iot," in *Future Internet of Things and Cloud Workshops (FiCloudW), IEEE International Conference on*, pp. 39–42, IEEE, 2016.

[129] S. Müller, "Spaces of rites and locations of risk: The great pilgrimage to mecca," in *The Changing World Religion Map: Sacred places, identities, practices and politics*, pp. 841–853, Springer, 2015.

[130] K. Haase, H. Z. Al Abideen, S. Al-Bosta, M. Kasper, M. Koch, S. Müller, and D. Helbing, "Improving pilgrim safety during the hajj: An analytical and operational research approach," *Interfaces*, vol. 46, no. 1, pp. 74–90, 2016.

[131] Q. A. Ahmed, Y. M. Arabi, and Z. A. Memish, "Health risks at the hajj," *The Lancet*, vol. 367, no. 9515, pp. 1008–1015, 2006.

[132] J. A. Al-Tawfiq, P. Gautret, S. Benkouiten, and Z. A. Memish, "Mass gatherings and the spread of respiratory infections: Lessons drawn from the hajj," *Annals of the American Thoracic Society*, no. ja, 2016.

[133] S. A. Hameed, "Ict to serve hajj: Analytical study," pp. 1–7, 2010.

[134] M. Mohandes, "Pilgrim tracking and identification using the mobile phone," in *Consumer Electronics (ISCE), 2011 IEEE 15th International Symposium on*, pp. 196–199, IEEE, 2011.

[135] M. Conti and S. Giordano, "Mobile ad hoc networking: milestones, challenges, and new research directions," *Communications Magazine, IEEE*, vol. 52, no. 1, pp. 85–96, 2014.

[136] J. Jing, A. S. Helal, and A. Elmagarmid, "Client-server computing in mobile environments," *ACM computing surveys (CSUR)*, vol. 31, no. 2, pp. 117–157, 1999.

[137] R. Droms, "Dynamic host configuration protocol," vol. 3, no. 1, 1997.

[138] M. M. Alani, "Manet security: A survey," in *Control System, Computing and Engineering (ICCSCE), 2014 IEEE International Conference on*, pp. 559–564, IEEE, 2014.

[139] J. V. D. Merwe, D. Dawoud, and S. McDonald, "A survey on peer-to-peer key management for mobile ad hoc networks," *ACM computing surveys (CSUR)*, vol. 39, no. 1, p. 1, 2007.

[140] "Amazon elastic compute cloud (ec2)." Available online at `http://aws.amazon.com/ec2/`.

[141] "Amazon relational database service (rds)." Available online at `https://aws.amazon.com/rds`.

[142] "Amazon simple storage service (s3)." Available online at `https://aws.amazon.com/s3/`.

[143] Qualcomm, "Trepn profiler." Available online at `https://developer.qualcomm.com/software/trepn-power-profiler`.

[144] "Amazon simple notification service (sns) documentation." Available online at `http://aws.amazon.com/documentation/sns/`.

[145] "Local and push notification programming guide: Apple push notification service." Available online at `{https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html}`.

[146] "Google cloud messaging (gcm)." Available online at `{http://developer.android.com/google/gcm/gs.html}`.

[147] G. E. Box, W. G. Hunter, J. S. Hunter, *et al.*, "Statistics for experimenters," 1978.

[148] F. J. Gravetter and L. B. Wallnau, *Statistics for the behavioral sciences.* Cengage Learning, 2016.

[149] Qualcomm, "Trepn profiler." Available online at `https://developer.qualcomm.com/software/trepn-power-profiler`.

[150] M. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing," in *INFOCOM, 2013 Proceedings IEEE*, pp. 1285–1293, IEEE, 2013.

[151] H. Saevanee, N. Clarke, and S. Furnell, "Sms linguistic profiling authentication on mobile device," in *Network and System Security (NSS), 2011 5th International Conference on*, pp. 224–228, IEEE, 2011.

[152] "Twilio." Available online at `https://www.twilio.com/`.

[153] D. Chen and H. Zhao, "Data security and privacy protection issues in cloud computing," in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, vol. 1, pp. 647–651, IEEE, 2012.

[154] C. He, X. Fan, and Y. Li, "Toward ubiquitous healthcare services with a novel efficient cloud platform," *Biomedical Engineering, IEEE Transactions on*, vol. 60, no. 1, pp. 230–234, 2013.

[155] Y. S. Lee, N. Bruce, T. Non, E. Alasaarela, and H. Lee, "Hybrid cloud service based healthcare solutions," in *Advanced Information Networking and Applications Workshops (WAINA), 2015 IEEE 29th International Conference on*, pp. 25–30, IEEE, 2015.

[156] S. Heron, "Advanced encryption standard (aes)," *Network Security*, vol. 2009, no. 12, pp. 8–12, 2009.

[157] H. S. Cabin and S. Henry, *The heart and circulation*, p. 5. New York,: Hearst Books, 1992.

[158] "Heart rate: What's normal?." Available online at `http://www.mayoclinic.org`.

[159] "Tizen developer." Available online at `https://developer.tizen.org/development/tools/download`.

[160] "Samsung gears." Available online at `http://www.samsung.com/global/microsite/gears/`.

[161] "Samsung accessory protocol (sap)." Available online at `http://developer.samsung.com`.

[162] J. Elton and P. W. Chung, "An integrated communications platform incorporating sms and e–mail to support mobile applications," *International Journal of High Performance Computing and Networking*, vol. 8, no. 1, pp. 3–15, 2014.

[163] J. K. Kim, R. Sharman, H. R. Rao, and S. Upadhyaya, "Efficiency of critical incident management systems: Instrument development and validation," *Decision Support Systems*, vol. 44, no. 1, pp. 235–250, 2007.

[164] E. T. Wilde, "Do emergency medical system response times matter for health outcomes?," *Health economics*, vol. 22, no. 7, pp. 790–806, 2013.

[165] J. D. Mayer, "Emergency medical service: delays, response time and survival," *Medical care*, pp. 818–827, 1979.

[166] T. A. Ambulance, "Contracting for emergency ambulances services," *Sacramento, CA: The American Ambulance*, pp. 15–8, 1994.

[167] L. Aboueljinane, Z. Jemai, and E. Sahin, “Reducing ambulance response time using simulation: The case of val-de-marne department emergency medical service,” in *Proceedings of the Winter Simulation Conference*, pp. 1–2, Proceedings of the Winter Simulation Conference, 2012.

[168] M. Neethu and R. Rajasree, “Sentiment analysis in twitter using machine learning techniques,” in *Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on*, pp. 1–5, IEEE, 2013.

[169] O. Kolchyna, T. T. P. Souza, P. C. Treleaven, and T. Aste, “Twitter sentiment analysis,” *CoRR*, vol. abs/1507.00955, 2015.

[170] F. Koto and M. Adriani, “A comparative study on twitter sentiment analysis: Which features are good?,” pp. 453–457, 2015.

[171] B. Liu and L. Zhang, “A survey of opinion mining and sentiment analysis,” *Mining text data*, pp. 415–463, 2012.

[172] Y. Zhang, R. Jin, and Z. Zhou, “Understanding bag-of-words model: a statistical framework,” *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 43–52, 2010.

[173] A. McCallum, K. Nigam, *et al.*, “A comparison of event models for naive bayes text classification,” in *AAAI-98 workshop on learning for text categorization*, vol. 752, pp. 41–48, Citeseer, 1998.

[174] W. J. Wilbur and K. Sirotkin, “The automatic identification of stop words,” *Journal of information science*, vol. 18, no. 1, pp. 45–55, 1992.

[175] “Node.js foundat ion.” Available online at `https://www.nodejs.org/en/` .

[176] “Amazon aws web services (aws).” Available online at `http://aws.amazon.com`.

[177] “The streaming apis | twitter developers.” Available online at `https://dev.twitter.com/streaming/overview`.

[178] F. De Backere, P. Bonte, S. Verstichel, F. Ongenae, and F. De Turck, “The ocareplatform: A context-aware platform to support independent living,” *Computer Methods and Programs in Biomedicine*, 2016.

[179] “Gcm syntax reference.” Available online at: `https://developers.google.com/cloud-messaging/http-server-ref`.

[180] P. R. de Andrade, A. B. Albuquerque, O. F. Frota, R. V. Silveira, and F. A. da Silva, "Cross platform app: a comparative study," *arXiv preprint arXiv:1503.03511*, 2015.

[181] G. Risterucci, T. Muntean, and L. Mugwaneza, "A new secure virtual connector approach for communication within large distributed systems," in *2015 14th International Symposium on Parallel and Distributed Computing*, pp. 185–193, IEEE, 2015.

[182] R. Pakdel and J. Herbert, "Scalable cloud-based analysis framework for medical big-data," in *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, vol. 2, pp. 647–652, IEEE, 2016.

Appendix A

Services and Scenarios

As an important part of this research work, this appendix brings together all the services that were developed in order to introduce them to the previously presented set of scenarios. In other words, this appendix will move step–by–step in determining how these services can be requested and would interact in the various scenarios. The manner in which these services are obtained and processed is also discussed.

A.1 On the Road

In a highway scenario, someone could be involved in an accident or experience an emergency medical situation. In this scenario, by utilizing the proposed middleware system, that person can: create a peer–to–peer network to request help, request medical advice from the system, search for a nearby doctor, or call a specific department.

The following subsections outline how these services can be processed through the system in accordance with its three layers (hardware, middleware and application) and the design principle provided previously.

A.1.1 Creating/joining a MANET to seek help

Someone who is involved in a car accident could, for example, benefit from the idea of creating a new MANET or joining an existing one in order to seek help.

This can be beneficial if that person does not have an Internet connection or network coverage in general but can at least use the Wi–Fi ability of a handheld device to search for people in his/her area.

Furthermore, that person could create a new MANET network locally to submit his/her needs whereby others can join the network to deliver support, such as by acting as a bridge to deliver the user's needs to the system.

The system hosts services that can deal with the case and, if necessary, reach the person's location to provide appropriate and swift help. However, a better outcome might result if the user who has just joined the created MANET to deliver support to the person experiencing the emergency is a medical professional, such as a doctor or nurse. In the last example, a high quality of care will be guaranteed.

Once the system is reachable, either by the person who created the MANET or anyone who joined it, the registration of this MANET can be done through the system, as shown in Figure A.1.

This step will ensure that support in terms of resources (e.g., storage) can be provided, as well as allowing the members of the MANET to benefit from other services such as starting a video call with a doctor to discuss the emergency case in detail or request an ambulance if needed.

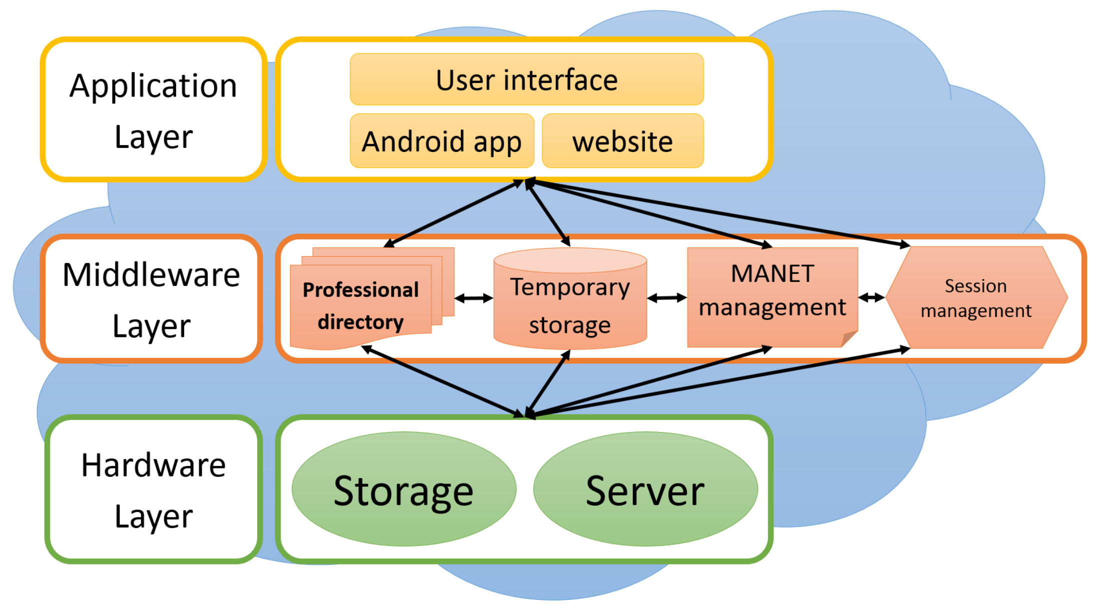


Figure A.1: Diagram Showing How a MANET Can Be Registered in The System When One of Its Members Has a Direct Link to The System

The registration request could be sent by any member who has an active link to the system of this MANET to which is attached some basic information, such as the total number of users (or at least the members known by this user), the name and ID of the MANET and the main purpose for creating the MANET (e.g., providing care/support to one of its members).

Once the MANET is registered successfully, resources such as storage will be allocated and a monitoring process will be started to report the behaviours of the MANET and its members for the purpose of guaranteeing that a high–quality service is delivered by this MANET to its members.

A.1.2 Retrieving medical advice

If an incident is relatively minor and only medical advice is needed, the user can access the system to obtain instructions. In other words, the usage of the system involves appropriate health content retrieval and delivery. Users need to provide some keywords (e.g., "injured", "minor cut") to allow the cloud to match this case with the appropriate help. The cloud will then reply with useful information using different kinds of multimedia, such as text, video and voice.

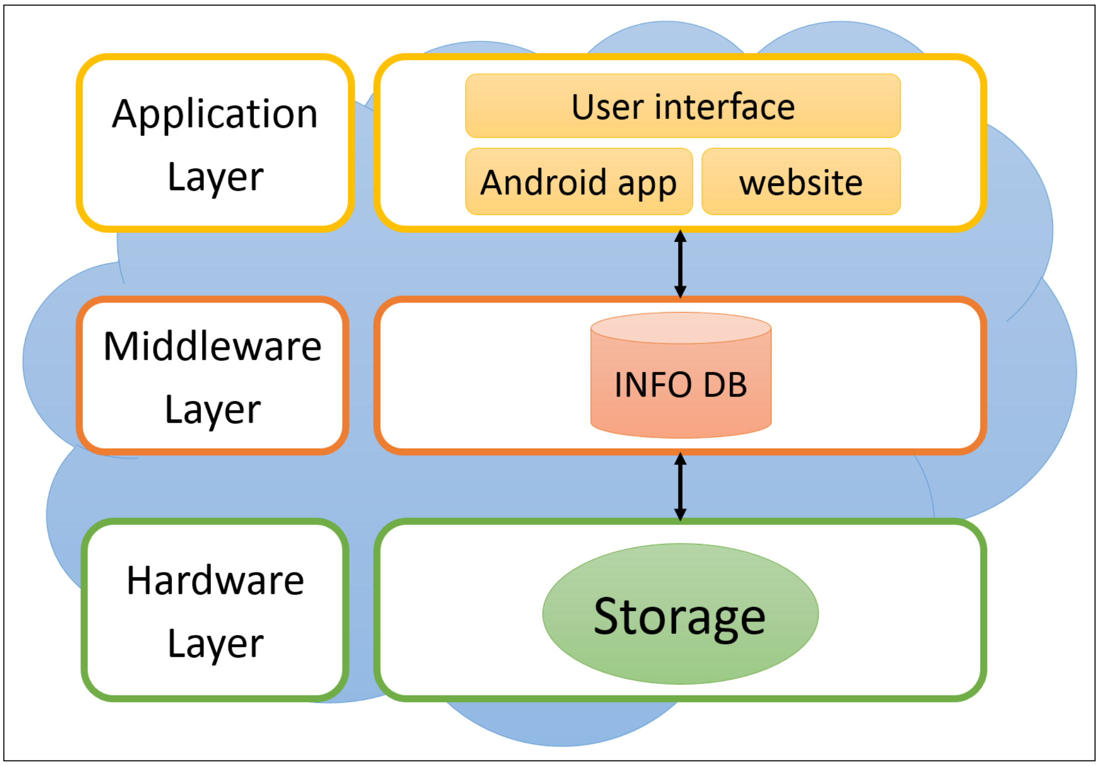


Figure A.2: Diagram Showing How Medical Advice Can Be Retrieved from The Cloud

In the cloud, once a request arrives at the application layer (the user interface component), it will travel to the “Information database”, which contains a number of cases and instructions about how to deal with each case, including pictures, text, voice and video.

The “Information database” is already connected to the “Storage” component that is located in the lowest of the cloud layers (hardware). When the requested case is found in the “Information database”, the cloud will reply to the requester and include all possible data that might be useful for the chosen case (as in Figure A.2).

A.1.3 Reporting accidents

When an accident occurs, anyone who discovers it should report it. Hence, this person, the reporter, needs to access the cloud to notify the Road Safety Department, for example.

In addition, the reporter can describe the state of the people who were involved in the accident, as well as passing on any useful information about the incident. The location of the reporter will be used to direct help to the accident location. Finally, the cloud will assign this case to the most appropriate department.

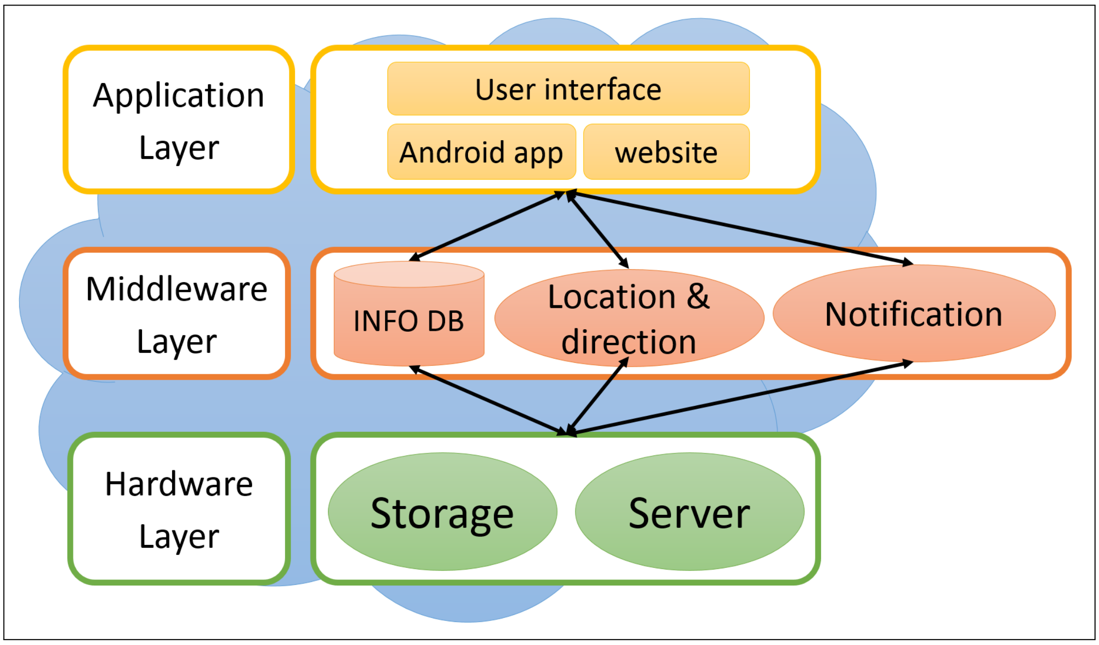


Figure A.3: Diagram Showing How a User Can Report an Accident to Get Help from The Cloud

Inside the cloud, the request will travel from the application layer to the "Location & direction service" to obtain the accident location, and to the "Notification" service to send a request to the relevant department.

Both services connect to the "Server" component, which is found in the hardware layer. After that, when the cloud receives an acknowledgement from the chosen department (depending on availability, location, and the patient's state), it will send directions to the accident to that department. An estimated time for when help will arrive will also be sent to the reporter. Figure A.3 demonstrates this operation.

A.1.4 Looking for doctors nearby

First, doctors need to set their status as "Available" to allow users to discover them. Then the user has to send a "Doctor near me" request to the system that will detect the user's location and search the "Professional directory" to find anyone who is available and near the user's location. When someone is found, a request is sent to the person and wait for acceptance. When acceptance is received, the cloud will provide directions to the user's location and any information about the case. Finally, the cloud will send an estimated time for when the doctor will arrive and establish communication between both of them if required.

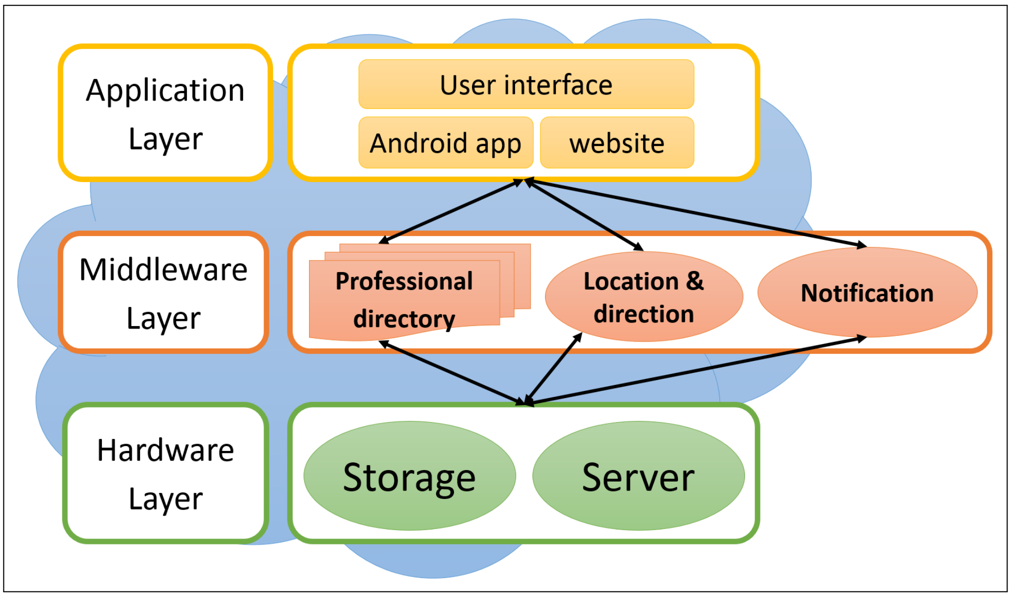


Figure A.4: Diagram Showing How a User Can Search for a Doctor Near to His/her Location

Moving to the cloud itself, the request will travel from the application level to the "Location & direction" on the service level to detect user location, then to the "Professional directory" to select a suitable person.

When someone is found, the request is sent through the "Notification" service. When the job is accepted, directions to the user's location will be sent using the "Location & direction" service that is connected to the "Server" component on the hardware level. Figure A.4 explains these steps.

A.2 At Home

Medical cases in the home are usually minor and the person or people concerned do not need to be taken to hospital. To ensure a professional manner in dealing with emergency cases, advice from experts will be delivered by the system as well as allowing users to call a doctor or search for the nearest medical centre. The next subsections detail some of these services.

A.2.1 Contacting experts

Once a case has been successfully treated, there may be situations in which someone needs to contact a doctor to ensure that the correct treatment is provided and to determine the next steps to take. The system allows users to send a request to the cloud after clarifying the status and needs of the case (e.g., the type of injury and the patient's age). The cloud will then query the "Professional directory" to find someone who can take the job. When an expert is chosen, the cloud will establish communication between the requester and the chosen expert.

Inside the cloud, the request will travel from the application layer to the "Video & Voice Call" component at the service layer, which is responsible for preparing live conversations between any two parties. The two services will cooperate to provide expert information that is available and suitable for this emergency case.

In the hardware layer, the "Connectivity" component provides all that this required by the "Video & Voice Call" service to establish communication. Figure A.5 presents this interaction.

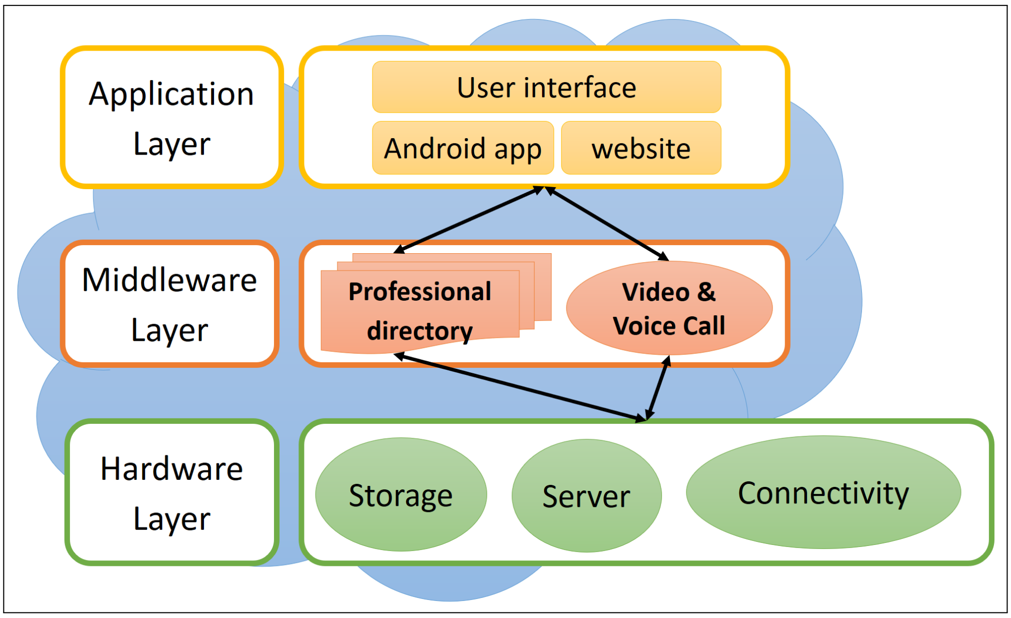


Figure A.5: Diagram Showing How a User Can Set Up a Communication Link With One of The Cloud's Registered Experts

A.2.2 Searching for a medical centre

If a user at home is willing to take the patient to hospital, directions to the nearest available hospital will be delivered by the system. The user needs to access the cloud to search for a medical centre, attaching location coordinates and patient needs.

Then, the cloud will check the "Hospital Directory" in the services layer to match the user's request based on the availability and suitability of the patient's case, as well as the distance from the patient's house.

When a hospital is found, the request will be sent to the "Location & directions" service to prepare the directions. Finally, the cloud will send the chosen hospital information and location to the requester. All these steps are shown in Figure A.6.

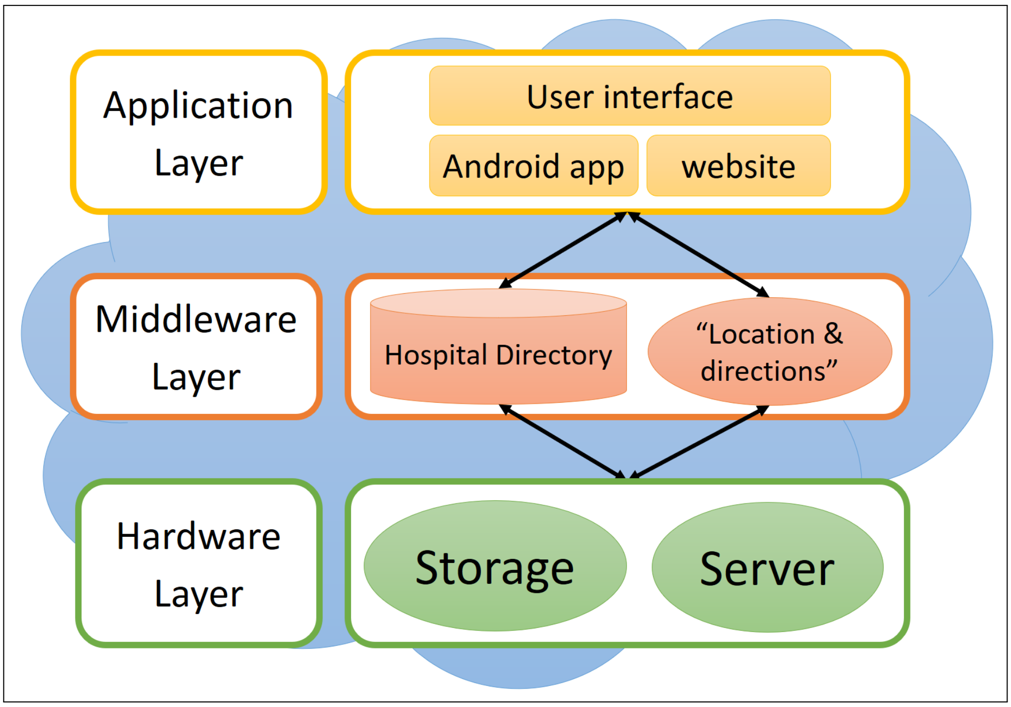


Figure A.6: Diagram Showing How a User Can Get Information of the Nearest Medical Centre from The Cloud

A.3 In Crowded Places

Detecting emergency cases in crowded places and delivering healthcare to anyone who needs it will lead to the better management of crowded places and ensure a high level of confidentiality to the people who are in those areas.

The system provides features to voluntary staff who are qualified to give care to people to allow them to do their job professionally, which will lead to further improvement in the level of healthcare.

The following two subsections explain how the previously presented mobile cloud services can be obtained in a crowded place.

A.3.1 Creating/joining a MANET

Users can benefit from creating a new MANET network or joining an existing one, such as accessing the cloud services over neighbouring links or sharing data with other members of the connected MANET.

On the other side, the cloud can provide more support to users who are present in crowded places, such as reaching as many users as possible to deliver notifications or disseminate useful information to the public.

Inside the cloud, the request to create/join a MANET will travel from the application layer to the "MANET management" component in the service layer. Then, this component will cooperate with the "Temporary storage" component to store/update the MANET table in the same layer.

The "Connectivity" component that is found in the hardware layer is responsible for meeting the needs of the "MANET management" service. However, if two users start a session to exchange messages or share content, interaction with the "Session management" component in the services layer will be required. Figure A.7 shows these steps.

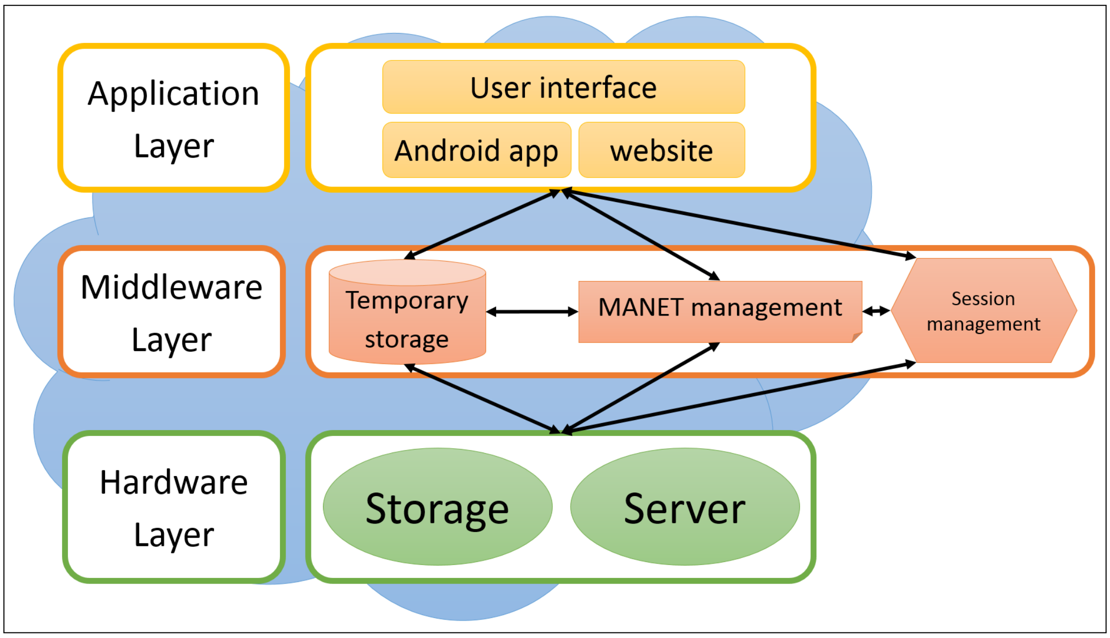


Figure A.7: Diagram Showing How a User Can Create or Join a Manet That is Managed by The Cloud

A.3.2 Social media service

Users who are in a crowded place can access the cloud via social media and send a tweet that contains the cloud hashtag to request help or check the latest updates from the cloud published on social media. This will enhance the availability of the cloud in general. This service can also benefit rescuers, as they can provide healthcare services or detect emergencies in real time.

With respect to the system architecture, this service sits on the service layer and can be called from the "User interface", which is one of the components of the application layer.

This service also cooperates with the main database for storage purposes and is connected to the "Storage" component in the hardware layer. Figure A.8 shows these interactions.

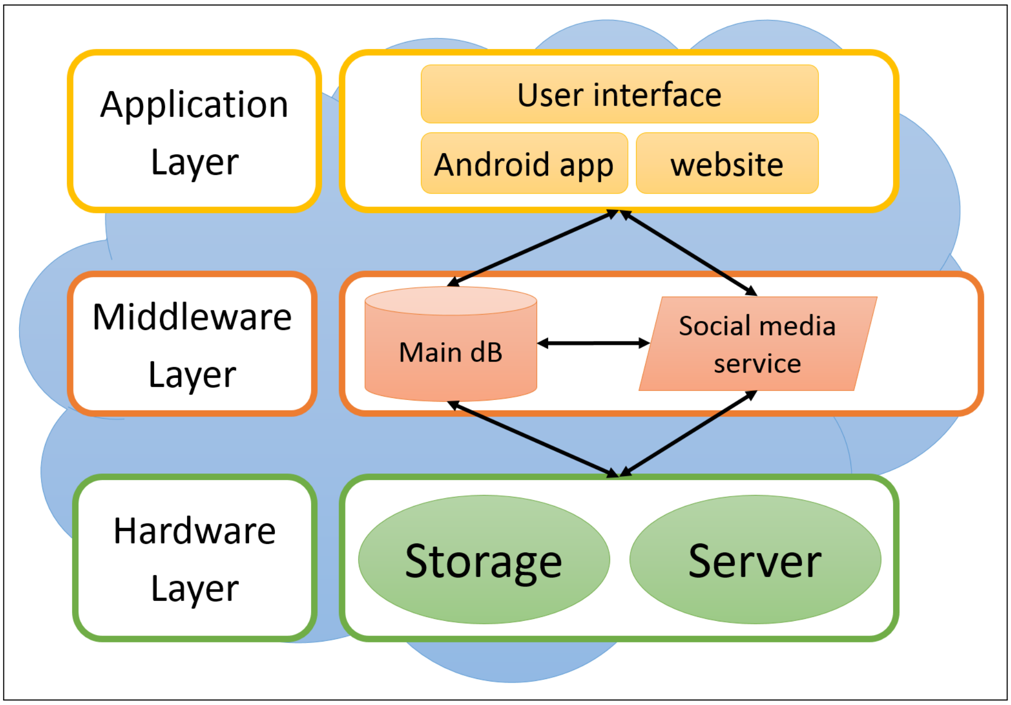


Figure A.8: Diagram Showing How a Social Media Service Can Be Used to Provide Support in Emergencies

A.4 Conclusion

This appendix has presented an important part of this research work by explaining how the services that were developed could be of relevance in the scenarios defined in chapter 2. For example, how users can access medical services delivered by the proposed system in crowded places such as Al–Hajj that was considered as just such an event in this thesis. The type of information and data needed when the service is requested were discussed, as well as providing an overview of the proposed middleware system and its services.

Appendix B

The Android App

Details are presented here of the Android app that is designed and installed on the user side to allow all the required interactions with the cloud and with neighbours to achieve a point–of–care healthcare system. The app is composed of three main components:

1. ***Users' account management***, such as signing up, signing in and signing out.
2. ***Connectivity***, such as creating, joining, and leaving a mobile ad–hoc network (MANET).
3. ***Services***, such as chatting, file sharing, and requesting emergency help.

The following sections present these components in detail.

B.1 Users' account management

Users are required to create an account in the cloud to use the app. Only users who successfully validate their accounts on the app ***sign–in screen*** are able to use the services provided.

The following subsection explains how these users can obtain a new account in the cloud and how that account can be used in this app.

B.1.1 Sign up

From the app's ***main screen***, users can click on the **Sign up** button to open a ***registration screen***. Four important pieces of information must be provided on this screen:

1. **User's email address:** as a default, the app will fill this field in advance with the device's email account (Gmail). This email address is, however, editable, making it possible for the user to enter a different email address.
2. **User's mobile telephone number:** the user's telephone number helps in retrieving sign–in information, as well as allowing an SMS to be received from the cloud. Users need to type in their telephone number, including the country code, for example: +353–877777777.
3. **Username:** a unique username (or nickname) is required here. It can contain letters (a—z) and numbers (0—9). It must also contain a minimum of four characters.
4. **Password:** a string that has to be at least four characters or digits in length. This password is required, along with the username, to accomplish the sign–in operation.

When the **Sign up** button is clicked, the app verifies that the provided data:

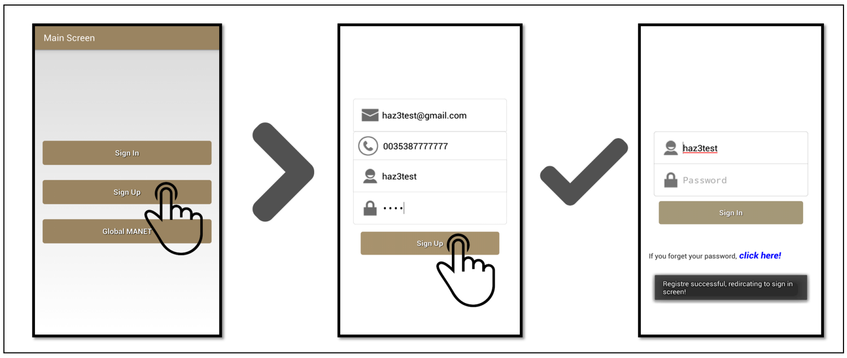
1. are in the correct format, such as the email having the at sign (@);
2. meet all the requirements, such as the minimum length for the username and password; and
3. that the device has an Internet connection and can reach the cloud.

If all data pass the verification process, an HTTP request is sent to the cloud that includes this information. The cloud then checks the request parameters, especially the username. The cloud will then make one of the following replies:

- If the username provided does NOT exist in the database, the cloud will add the user information and notify the user that the registration process has been completed successfully.
- If the username is already in the cloud database, the cloud will reply to the user that registration has failed.

Depending on the reply from the cloud, the app will take appropriate further action. If the reply is positive, the ***sign–in screen*** is launched with the username field completed and the password field awaiting user input. However, if the reply is negative, the user will be notified and all fields will be reset.

The following screenshots show the ***sign–up screen*** in the app, and how a user can create a new account in the cloud:



haz3test@gmail.com
0035387777777
haz3test
····|

haz3test

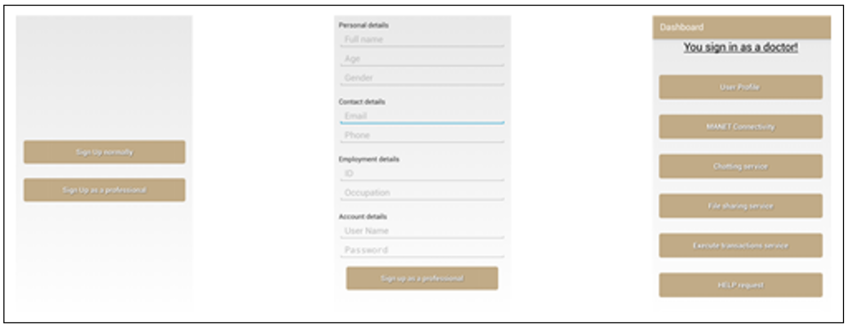
If you forget your password, ***click here!***

Registre successful, redircating to sign in screen!

B.1.2 Professional registration

In the case that the user who wants to register in the cloud is a medical professional, such as a doctor or a nurse, a special ***registration screen*** is provided to gain extra information, such as user employment ID, occupation, etc.

The following screenshots show how a professional can create a new account in the cloud using the Android app.



You sign in as a doctor!

B.1.3 Sign in

B.1.3.1 With an existing account

As mentioned above, each user has to provide a previously defined username and password to be able to access app services. Again, on the ***main screen***, there is a **Sign in** button that leads to the ***sign–in screen***. When this screen is shown, the user has to type in the username and password. The sign–in process starts when the button on this screen is clicked and both the username and password have been provided.

First, the app sends the username and password to the cloud to check that the user is authorized to access the app services. The cloud's reply will depend on whether the username is found in the database, as well as whether the password provided is correct. If the information provided is invalid, the user will be notified and the fields will reset to empty. If the username and password are valid, an SMS authentication process will start, providing a high level of protection.

The cloud sends a one–time unique generated code via the Twilio platform as an SMS message. If the user receives this message, it will re–input the code on the screen to send it to the cloud. The cloud finally validates the code to either allow or deny access to this user. If the user is allowed, the cloud updates the user ***last–seen field***, suggests a MANET_ID and IP address for that user, and replies that the user is authorized. A number of actions will then be performed:

1. A timer is established in the background that has a value of one hour, whereby the user will be signed out from the app when the time expires unless the user updates the timer. (More information about the timer framework is provided in the timer section below.)

2. A new shared preference is saved in the app that contains the user information as well as the suggested MANET_ID and IP address.

3. In the background, the current location of the user is submitted to the cloud.

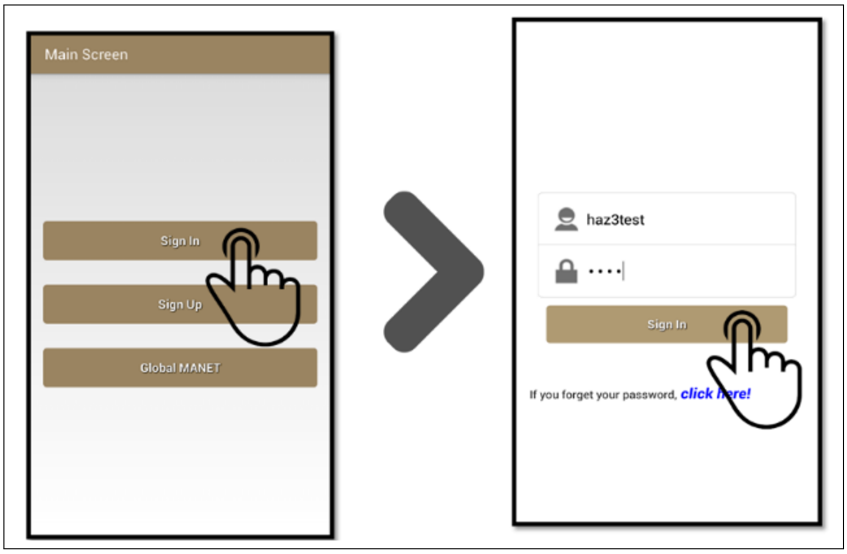
4. The app obtains the Google Cloud Messaging (GCM) ID of the device and sends it to the cloud to be used in the notification service.

5. The device status (such as battery level and connection strength) is sent to the cloud.

6. Finally, the ***dashboard screen*** is launched to allow the user to use the services provided.

However, if the user is a return user and has already signed in, there is no need to repeat the sign–in process. The user will be forwarded to the ***dashboard screen***. When the user clicks on the **Sign in** button, this checks whether shared preferences have been created for that user. If shared preferences have been created, the timer is still active. If not, the user needs to sign in again.

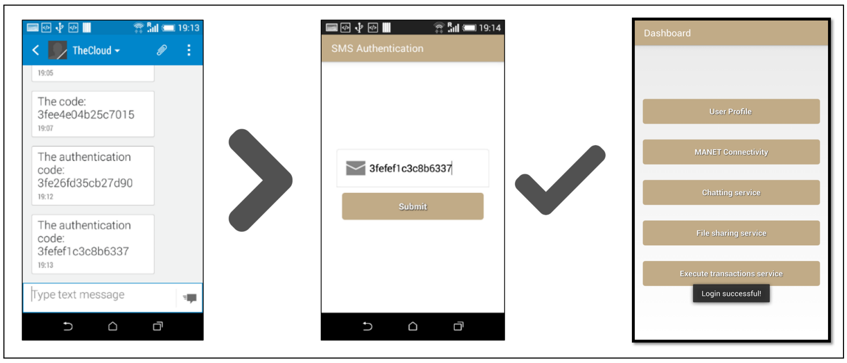
The following screenshots show the sign–in process in the app, including SMS authentication:



haz3test

••••

If you forget your password, click here!



19:05

The code: 3fee4e04b25c7015
19:07

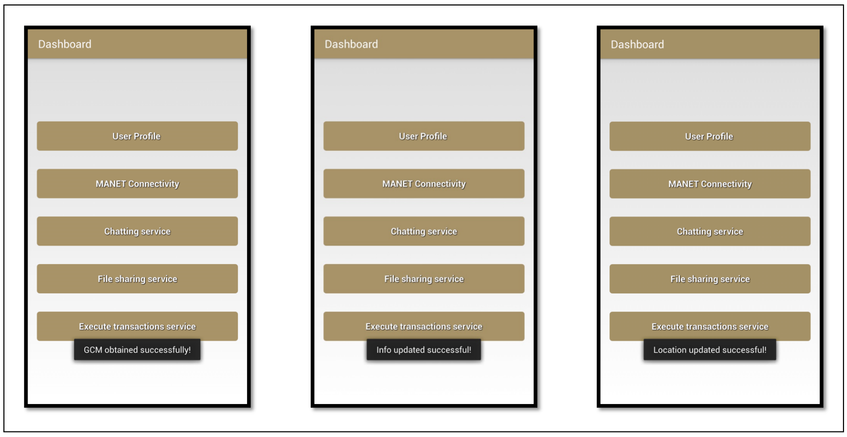
The authentication code: 3fe26fd35cb27d90
19:12

The authentication code: 3fefef1c3c8b6337
19:13

Type text message

3fefef1c3c8b6337

Login successful!



GCM obtained successfully!

Info updated successful!

Location updated successful!

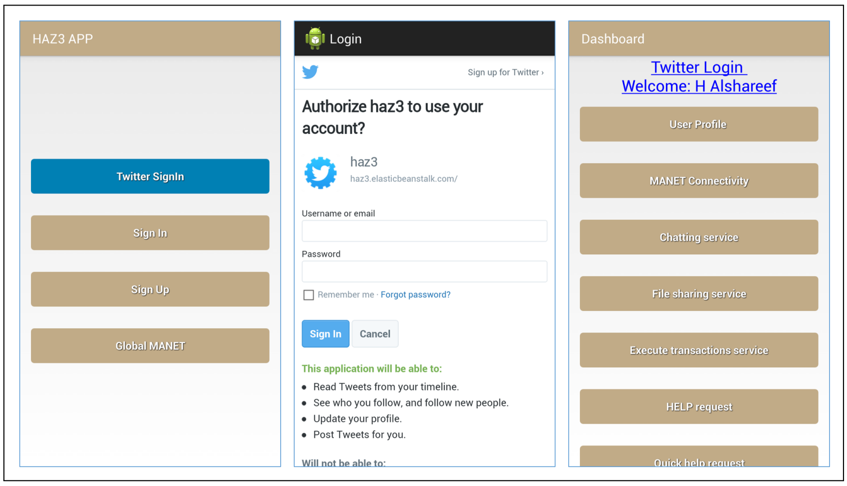
B.1.3.2 Sign in with Twitter account

To extend the user base of the proposed system and introduce another way to access cloud services, users are allowed to use Twitter account credentials (username and password) to sign in. Once a user clicks the ***Twitter Sign in*** button, the app gains authorization from the Twitter API. If the username and password provided are correct, the app redirects the user to the app's ***Dashboard*** activity.

There are more services coordinating with the cloud in this activity, such as sending a help request to the cloud. At the same time, the user profile in the cloud will be updated. If there is no existing account for that user, the cloud will create a new account for that user based on information from the Twitter account.

Furthermore, actions taken in the previous section, such as obtaining the GCM ID and ascertaining current location, will also be taken here. The SMS authentication process will not be executed here because Twitter's verification is sufficient.

The following screenshots show how a user can sign into the Android app using a Twitter username and password:



Authorize haz3 to use your account?

haz3
haz3.elasticbeanstalk.com/

Username or email

Password

□ Remember me · Forgot password?

This application will be able to:

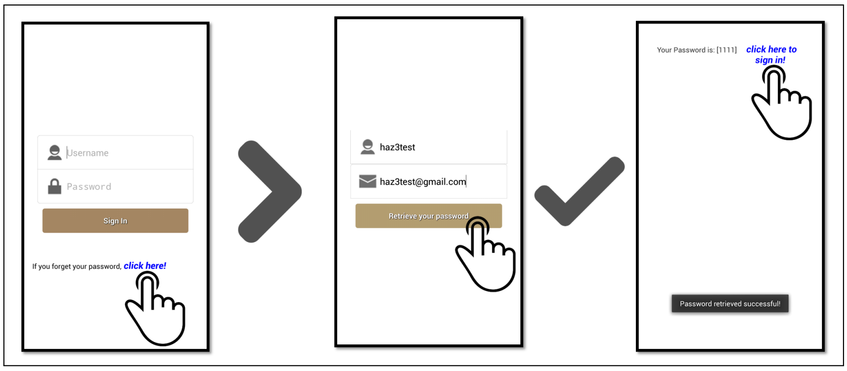
- Read Tweets from your timeline.
- See who you follow, and follow new people.
- Update your profile.
- Post Tweets for you.

Will not be able to:

B.1.3.3 Retrieving sign–in information

The app offers an ***if you forget your password*** feature, whereby users can retrieve their password by clicking on a special link found on the ***sign–in screen***. This link leads to another screen that asks users to provide their account username and email address. Both values are sent to the cloud to check whether the username has been registered and the email address provided is correct.

If the cloud validates these values, the password will be received from the cloud and shown in the app screen. The following screenshots show how a user can retrieve a lost password:



If you forget your password, click here!

haz3test

haz3test@gmail.com

Your Password is: [1111]

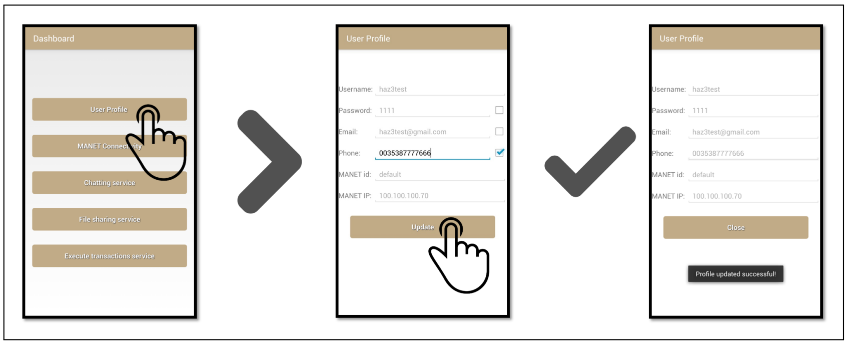
Password retrieved successful!

B.1.4 User profile

If a user signs in successfully, that user's details will be held in a shared preferences format for the purpose of fast access and ease of editing. Users are able to view/update their profiles using the ***profile screen***. All the information on this screen is listed in disabled text fields. Next to each text field there is a checkbox that allows the text file to be edited if ticked.

Once the user edits the text field, an **Update** button is shown to complete the updating action on the mobile device (locally) and in the cloud. However, username and MANET details cannot be edited because the first requires a new registration and the second can be edited from the connectivity screen.

The following screenshots show how users can view/edit their profile:



Username: haz3test
Password: 1111 □
Email: haz3test@gmail.com □
Phone: 0035387777666 ☑
MANET id: default
MANET IP: 100.100.100.70

Username: haz3test
Password: 1111
Email: haz3test@gmail.com
Phone: 0035387777666
MANET id: default
MANET IP: 100.100.100.70

Profile updated successful!

B.1.5 Sign out

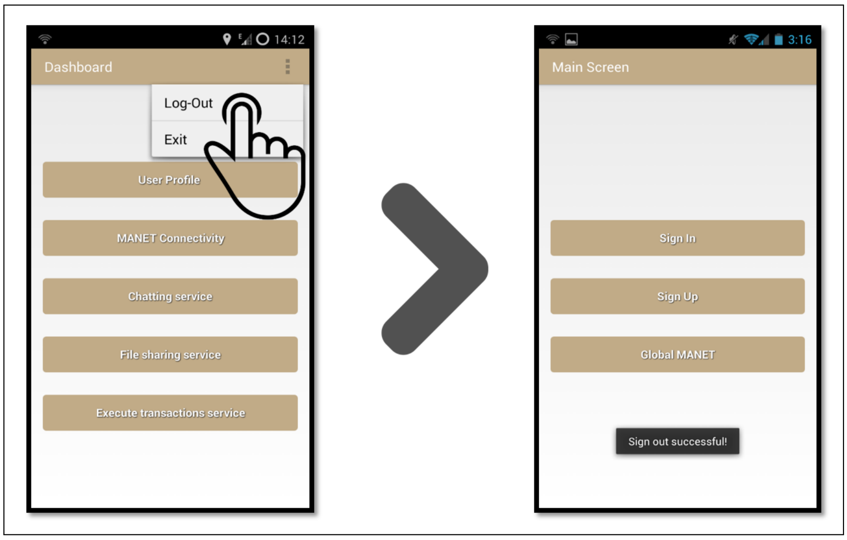
The following two scenarios will result in the user signing out of the app.

B.1.5.1 Performing a normal sign–out process

Users can leave the app by clicking the **Log out** option on the ***dashboard screen***. This leads to a number of actions:

1. Leaving the connected MANET if the user is connected to one already. Any background services associated with the MANET connection will stop, such as neighbour discovery.
2. Sending a sign–out request to the cloud, whereby the cloud will mark the user as inactive.
3. Deleting the timer that started when the user signed in.
4. Moving the user to the ***main screen***.

The following screenshots show the sign–out process from the app:



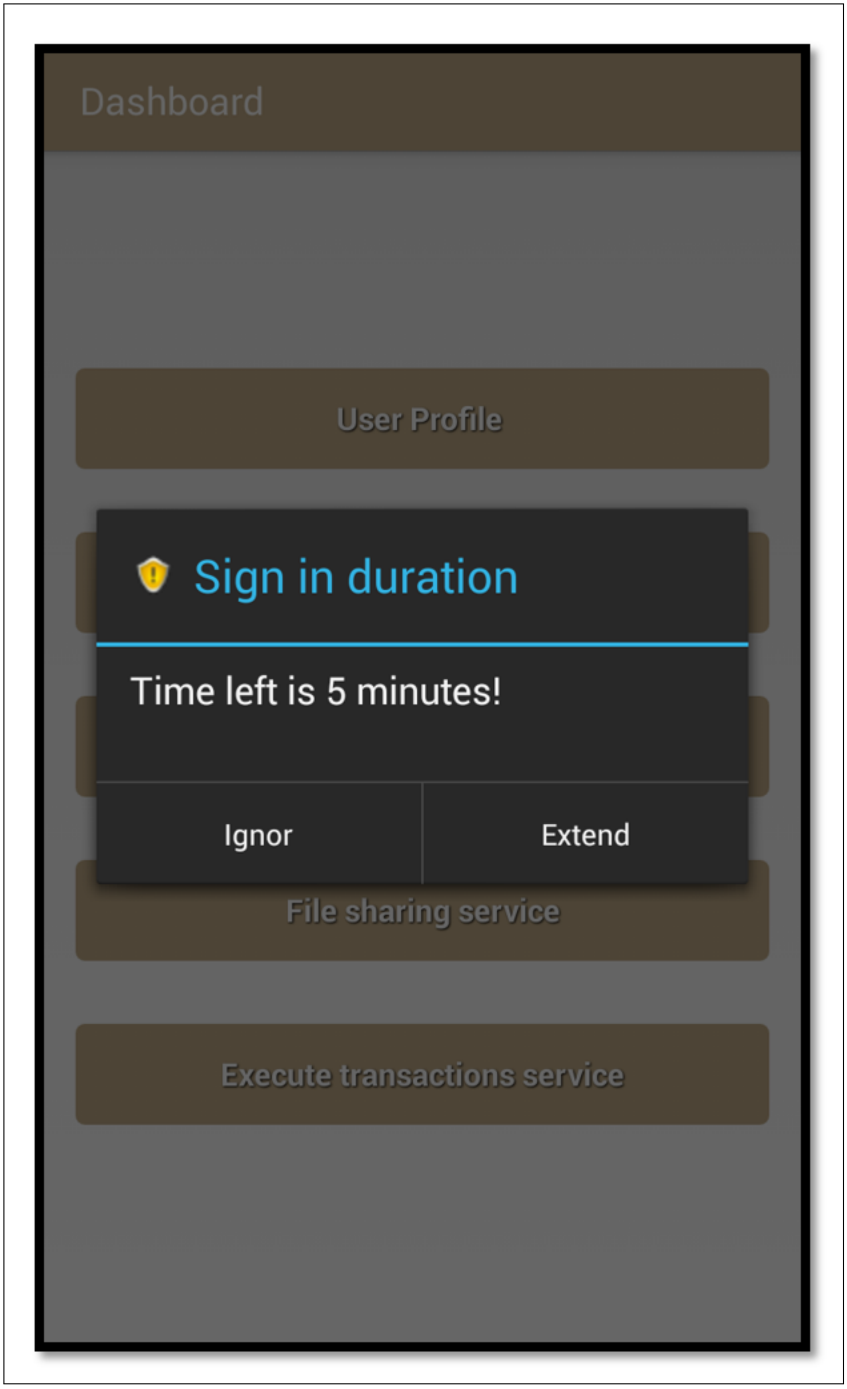
Sign out successful!

B.1.5.2 Sign–in timer

As mentioned previously, when a user signs in successfully, a one–hour countdown timer is started and runs in the background. The user will be alerted when there are five minutes remaining.

The user can extend the timer by clicking the **Extend** button on the ***alert screen***, which results in another hour being added to the timer's total time.

Clicking the **Ignore** button or not taking any action will result in waiting until the remaining time has finished, then the user being forced to sign out and being marked as inactive in the cloud. The following screenshot shows the timer alert:



Dashboard

Sign in duration

Time left is 5 minutes!

B.2 MANET connectivity

This section explains how an ad–hoc networking mode is gained and how users can create, join and leave a MANET network.

B.2.1 Device Wi–Fi mode

One of the difficult issues when enabling ad–hoc networking in mobile devices is that most venders block this feature for various reasons. Therefore, devices have to be rooted to allow Wi–Fi chip access, whereby a device's Wi–Fi chip can be configured to accept an ad–hoc networking mode. As a result, the Wi–Fi chip will not be able to connect to the Internet via a router or broadband. In other words, an Internet connection can only be allowed using a mobile network, such as 3G or 4G. For this reason, there are a limited number of devices operating with this modification.

However, once super–user access is allowed and the Wi–Fi chip is configured to ad–hoc mode, devices can perform in this mode more easily. The proposed app uses a *MANET Manager app* as a library to access a device's kernel to flick to Wi–Fi mode using commands that are predefined in C++ files and executed using a Native Development Kit (NDK) package.

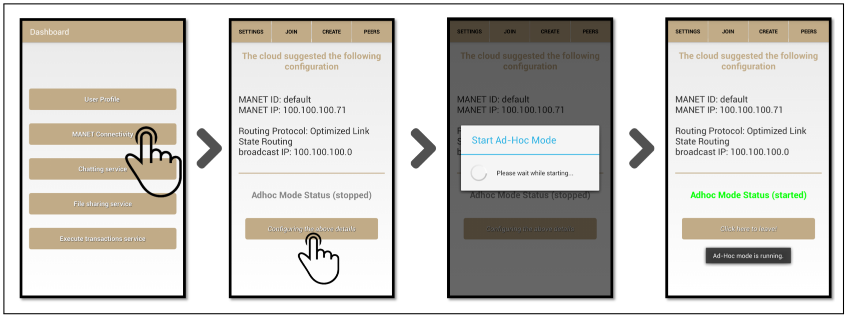
B.2.2 Routing protocol

Optimized Link State Routing (OLSR) is chosen here as it is the main protocol for all MANET networks. This protocol is proactive, which means the routing table will be built from all the available neighbours and the route for reaching each neighbour.

B.2.3 Connectivity screen

There is a button, called **MANET Connectivity**, on the ***dashboard screen*** that leads to the ***connectivity screen***. When the button is clicked, summary information is shown to users. The ability to change Wi–Fi mode is also offered. A suggested MANET ID and IP address are shown on the screen so that the user can join the suggested MANET or create/join another one.

The following screenshots show the main connectivity screen:



The cloud suggested the following configuration

MANET ID: default
MANET IP: 100.100.100.71

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

Adhoc Mode Status (stopped)

Start Ad-Hoc Mode

Please wait while starting...

Adhoc Mode Status (started)

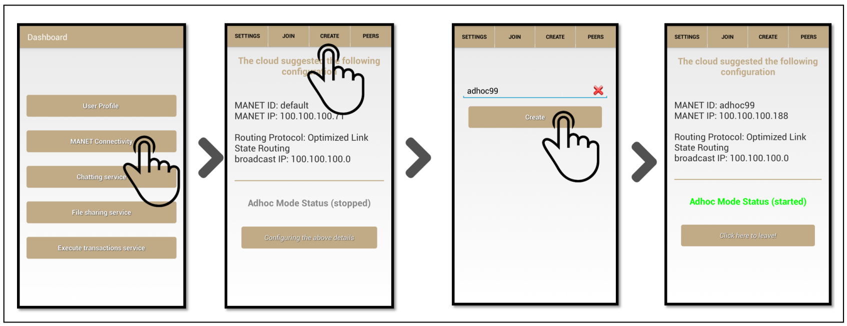
Ad-Hoc mode is running.

B.2.4 Creating a new MANET

There is an action bar at the top of the ***settings*** screen where users can create a new network or join an existing one. To create a new MANET network, the user needs to click on the **CREATE** tab, and then type in the name of the new network. It has to appear as one word without any spaces. Once it has been typed in, a **Create** button is provided that, when clicked, sends a request to the cloud. The cloud checks whether the new name conflicts with other networks and, if not, generates a new IP address if needed.

Once verification of the network has been achieved, the user will be brought back to the ***settings*** screen after updating the MANET_ID and IP address. Finally, the device is connected to the new MANET.

The following screenshots show the above framework:



The cloud suggested the following configuration

MANET ID: default
MANET IP: 100.100.100.71

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

Adhoc Mode Status (stopped)

adhoc99

The cloud suggested the following configuration

MANET ID: adhoc99
MANET IP: 100.100.100.188

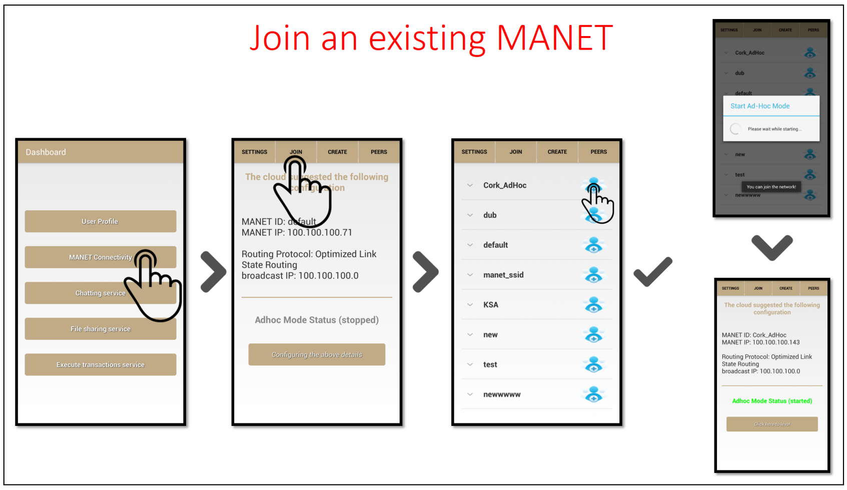
Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

Adhoc Mode Status (started)

B.2.5 Joining a MANET

If a user clicks on the **JOIN** tab on the ***connectivity screen***, a list of all the existing MANET networks in the cloud is shown. Clicking on one of these networks will result in a request being sent to the cloud to join this network. In the cloud, the user information will be added to this network. In the app, the device will be joined to this MANET and the ad–hoc mode starts.

The following screenshots show how users can join an existing MANET:



Join an existing MANET

The cloud suggested the following configuration

MANET ID: default
MANET IP: 100.100.100.71

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

Adhoc Mode Status (stopped)

Please wait while starting...

You can join the network!

The cloud suggested the following configuration

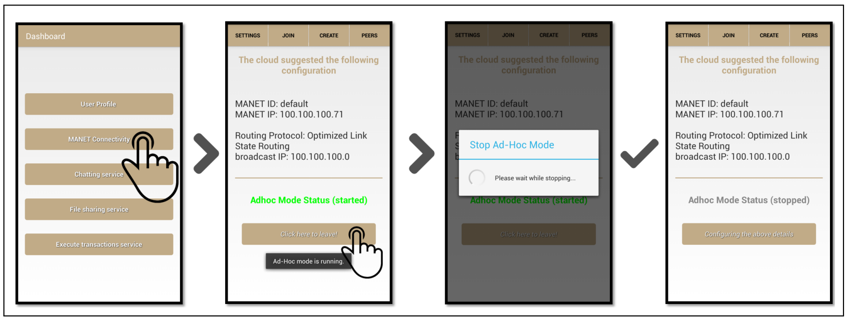
MANET ID: Cork_AdHoc
MANET IP: 100.100.100.143

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

Adhoc Mode Status (started)

B.2.6 Leaving the MANET

Users can leave the connected MANET and flip back from Wi–Fi to normal mode by clicking on the **Click here to leave!** button found on the ***settings*** screen. The following screenshots show the leaving framework:



The cloud suggested the following configuration

MANET ID: default
MANET IP: 100.100.100.71

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

Adhoc Mode Status (started)

Ad-Hoc mode is running.

Please wait while stopping...

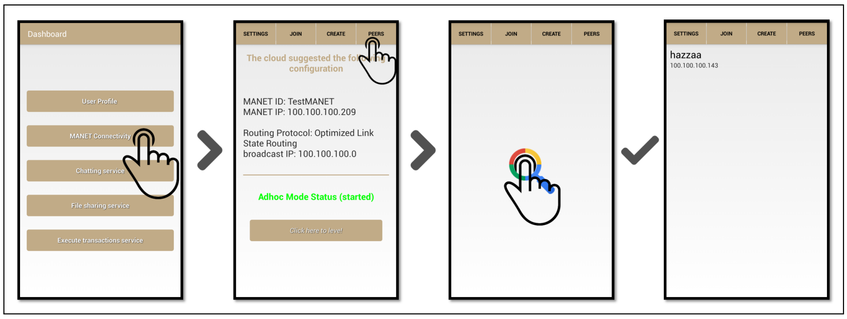
Adhoc Mode Status (stopped)

B.2.7 Neighbour discovery

Once the device is configured to ad–hoc mode and connected to a MANET, a User Datagram Protocol (UDP) message broadcasts a "Hello" message to the Wi–Fi range to draw the attention of other devices. The same device starts a background service to listen to any incoming "Hello" messages from neighbours.

Using the app, users can find neighbours by clicking on the **PEERS** tab on the ***connectivity screen*** to view a list of all nearby neighbours.

The following screenshots show how users can view a list of neighbours:



The cloud suggested the fol ing configuration

MANET ID: TestMANET
MANET IP: 100.100.100.209

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.100.100.0

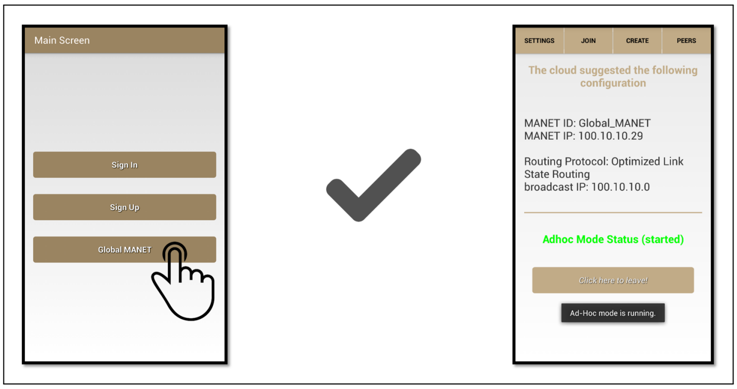
Adhoc Mode Status (started)

hazzaa
100.100.100.143

B.2.8 Global MANETs

If a user does not have an Internet connection to sign in to the cloud, or does not have an account in the cloud at all, the app provides a way to access its services temporarily until the cloud can be reached. The idea is to create a temporary account in which a username is taken from the device manager, an IP address composed from the device's media access control (MAC) address to avoid conflict, and a MANET_ID assigned to "Global–MANET".

In this MANET, there will already be users who are connected to it for the same reason: no Internet connection or active account. Once the Internet connection is available, users have the choice of signing in to the cloud, creating a new account if they do not already have one, or staying in the network after sending a notification to the cloud. The benefit of sending a notification to the cloud is that the cloud will consider the user's link as active and use it to reach other users in the MANET. The following screenshots show how users can connect to a Global MANET:



The cloud suggested the following configuration

MANET ID: Global_MANET
MANET IP: 100.10.10.29

Routing Protocol: Optimized Link State Routing
broadcast IP: 100.10.10.0

Adhoc Mode Status (started)

Ad-Hoc mode is running.

B.3 App services

The app provides a number of services to its users with the help of the cloud. The following subsections present these services, including details of how users can use each one.

B.3.1 Chat service

This service allows users who are in the same MANET to exchange messages. Only devices configured to ad–hoc mode and connected to a MANET network can use this service.

Framework

A user can start a chat service by clicking on the **Chatting service** button on the ***dashboard screen***. This button leads to the ***chatting setup screen***. After checking that the device is connected to a MANET, all discovered neighbours are included in a dropdown list to allow users to select a new peer to chat with.

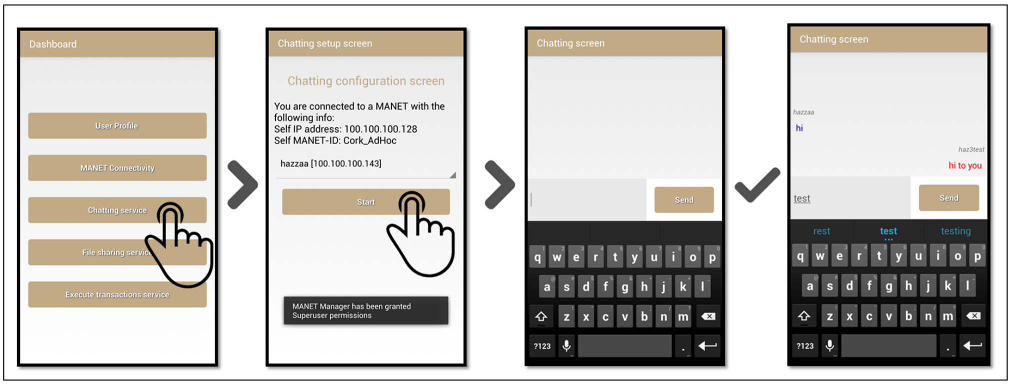
Once a neighbour has been chosen, the user is ready to start a chat session and a request is sent to that peer using the Transmission Control Protocol (TCP) socket protocol.

The chosen peer can accept or reject the request. If the request is accepted, both devices will open a **chat screen**, whereby each device has a text field for typing a message and a button to send this message, as well as a listview to show the conversation.

Leaving a chat session can be done simply by closing this screen, which means that if one of the users does not receive a new message, it will be assumed that the other user has left the session.

The following screenshots show how users can start a chat session with a neighbour:

Sender side



Chatting configuration screen

You are connected to a MANET with the following info:
Self IP address: 100.100.100.128
Self MANET-ID: Cork_AdHoc

hazzaa [100.100.100.143]

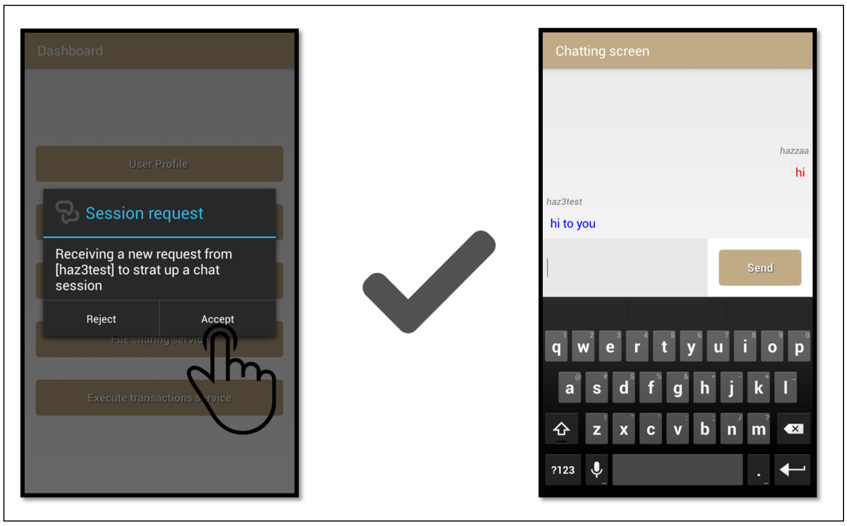
MANET Manager has been granted Superuser permissions

hazzaa
hi

haz3test
hi to you

test

Receiver side



Session request

Receiving a new request from [haz3test] to strat up a chat session

hazzaa
hi

haz3test
hi to you

B.3.2 File–sharing service

This service offers users the ability to upload files to the cloud or share files with one–hop neighbours.

B.3.2.1 Uploading files to the cloud

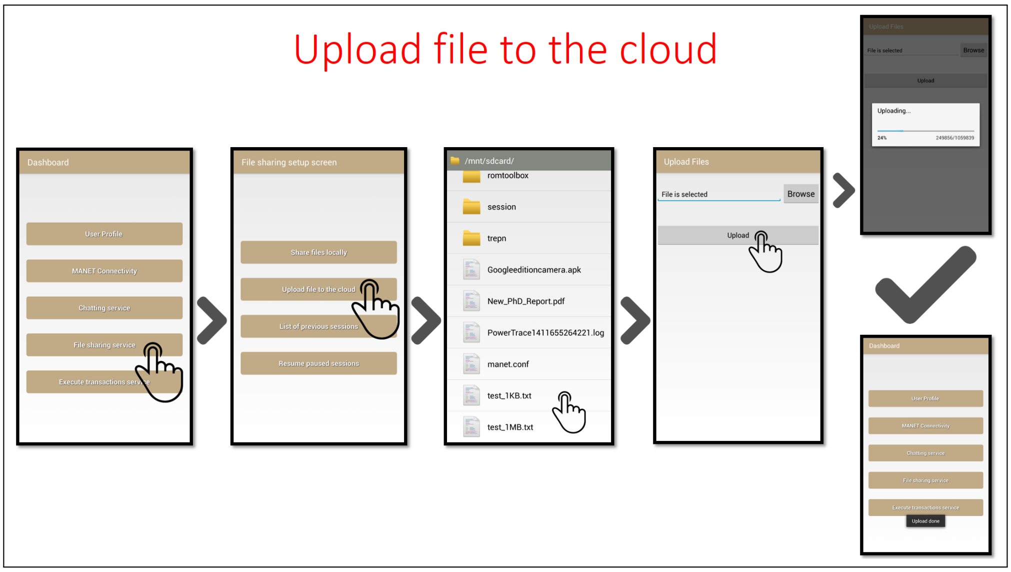
The service allows users to upload an existing file from a mobile device Secure Digital (SD) card to the cloud to be stored in the S3 storage of the cloud platform.

Framework

Users can navigate from the ***dashboard screen*** to upload the file service by clicking on the **File sharing service** button, which leads to the ***file sharing setup screen***. An **Upload file to the cloud** button is provided on this screen. Once this button is clicked, all files in the mobile device are listed to allow selection.

When a file is selected, an **Upload** button is shown which starts the upload process to the cloud once it is clicked. A progress bar pops up to show upload progress. If the file is uploaded to the cloud successfully, the user will be notified and brought back to the ***dashboard screen***.

The following screenshots show this framework:



Role of the cloud

The cloud's role here is handling the multi–part HTTP request from the user by reading its contents (such as file name and content), then redirecting the request to a servlet class responsible for dealing with this type of request. The uploading file is checked in this servlet class and stored in the S3 storage. To avoid file name conflicts, a random value is added to the filename as a prefix to ensure uniqueness.

B.3.2.2 Sharing files locally

Files can be shared with MANET members who are one hop away from the sender. Each file will be split into four parts of equal size, and a checkpointing technique can be used to ensure that the session is accomplished.

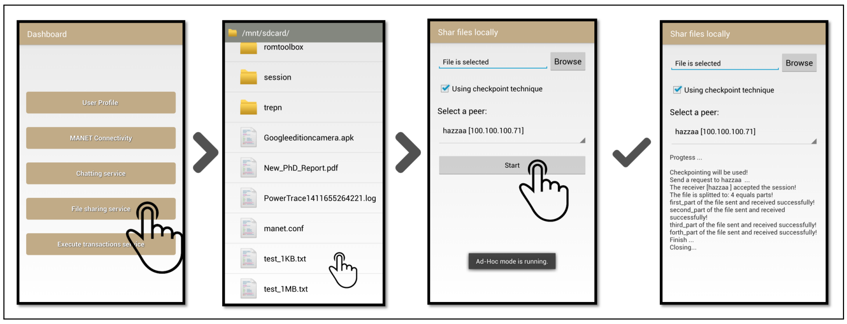
Framework

From the **File sharing** button on the ***dashboard screen***, users can click the **Share files locally** button to start a file–sharing session. First, the user has to select a file to share from the list of available files. Then the user has to choose whether to use a checkpointing technique, by ticking the checkbox provided, to save session progress to prevent data being lost in the case of a broken link. Second, the user has to select a destination neighbour from the dropdown list provided. Finally, the user has to click the **Start** button to begin the session. The session starts by sending a request to the destination node and waiting for a reply. If the node accepts the request, the selected file is split into four equal–sized parts. Then the sender starts to send the parts one by one, whereby no new part will be sent unless the previous one has been received.

Alternatively, if checkpointing was selected, each time a part is received successfully, a new checkpoint is created in both mobile devices. When all parts have been sent and received successfully without any issues, the cloud will be notified about the termination of the session, whereby the most recent checkpoint will be uploaded to the cloud in case the session is interrupted. Finally, the receiver will merge all the received parts to restore the original file.

The following screenshots show how users can share files using such a MANET:

Sender side



File is selected

☑ Using checkpoint technique

Select a peer: hazzaa [100.100.100.71]

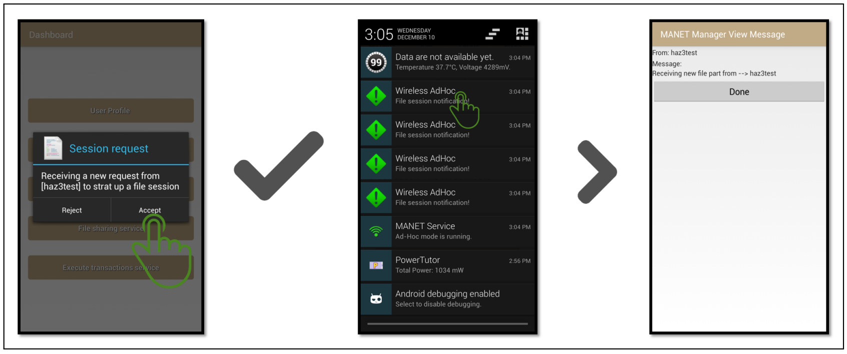
File is selected

☑ Using checkpoint technique

Select a peer: hazzaa [100.100.100.71]

```
Progtess ...

Checkpointing will be used!
Send a request to hazzaa  ...
The receiver [hazzaa ] accepted the session!
The file is splitted to: 4 equals parts!
first_part of the file sent and received successfully!
second_part of the file sent and received
successfully!
third_part of the file sent and received successfully!
forth_part of the file sent and received successfully!
Finish ...
Closing...
```

Receiver side



Session request

Receiving a new request from [haz3test] to strat up a file session

Data are not available yet. 3:04 PM
Temperature 37.7°C, Voltage 4289mV.

Wireless AdHoc 3:04 PM
File session notification!

Wireless AdHoc 3:04 PM
File session notification!

Wireless AdHoc 3:04 PM
File session notification!

Wireless AdHoc 3:04 PM
File session notification!

MANET Service 3:04 PM
Ad-Hoc mode is running.

PowerTutor 2:56 PM
Total Power: 1034 mW

Android debugging enabled
Select to disable debugging.

MANET Manager View Message

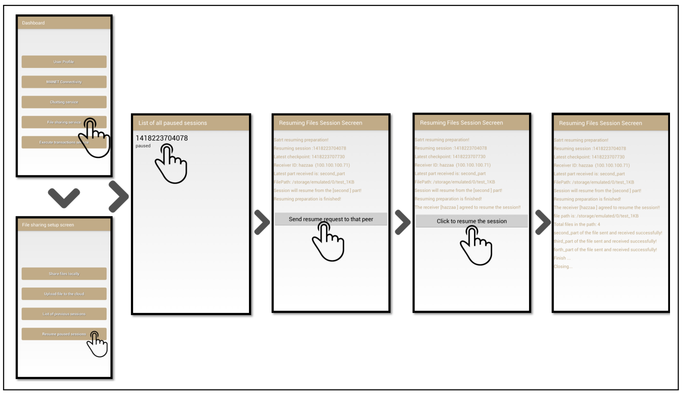
From: haz3test
Message:
Receiving new file part from --> haz3test

Resuming a paused session

A list of all paused sessions is presented if the **Resume paused sessions** button is clicked on the ***file sharing screen***. If a user clicks on one of these paused sessions, a resuming operation is started. First, the session information, such as session ID, receiver ID, and the latest checkpoint, is retrieved from the local database. From the latest checkpoint ID, the app can retrieve information about which parts were sent and which were not. The app will then perform a look up in the routing table to establish whether a connection with the receiver is feasible. If so, a resuming request will be sent to the receiver. If the receiver accepts the request, the remaining parts will be supplied to the receiver, waiting for ACKs to create new checkpoints. Finally, if the session is completed, the database record will be updated and the cloud notified.

The following screenshots show the framework for resuming a paused file–sharing session:

Sender side


Dashboard
User Profile
MANET Connectivity
Chatting service
File sharing service
Execute transactions service
File sharing setup screen
Share files locally
Upload file to the cloud
List of previous sessions
Resume paused sessions
List of all paused sessions
Resuming Files Session Secreen
Send resume request to that peer
Click to resume the session

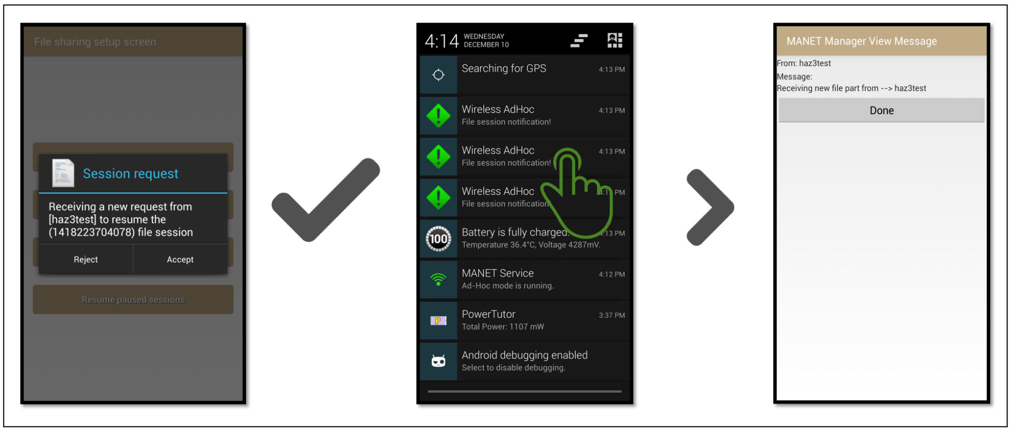
1418223704078
paused

Satrt resuming preparation!
Resuming session :1418223704078
Latest checkpoint: 1418223707730
Receiver ID: hazzaa (100.100.100.71)
Latest part received is: second_part
FilePath: /storage/emulated/0/test_1KB
Session will resume from the [second] part!
Resuming preparation is finished!

Satrt resuming preparation!
Resuming session :1418223704078
Latest checkpoint: 1418223707730
Receiver ID: hazzaa (100.100.100.71)
Latest part received is: second_part
FilePath: /storage/emulated/0/test_1KB
Session will resume from the [second] part!
Resuming preparation is finished!
The receiver [hazzaa] agreed to resume the session!!

Satrt resuming preparation!
Resuming session :1418223704078
Latest checkpoint: 1418223707730
Receiver ID: hazzaa (100.100.100.71)
Latest part received is: second_part
FilePath: /storage/emulated/0/test_1KB
Session will resume from the [second] part!
Resuming preparation is finished!
The receiver [hazzaa] agreed to resume the session!!
file path is: /storage/emulated/0/test_1KB
Total files in the path: 4
second_part of the file sent and received successfully!
third_part of the file sent and received successfully!
forth_part of the file sent and received successfully!
Finish ...
Closing...

Receiver side


File sharing setup screen
Session request
Reject
Accept
Resume paused sessions
4:14 WEDNESDAY DECEMBER 10
Searching for GPS 4:13 PM
Wireless AdHoc 4:13 PM
File session notification!
Wireless AdHoc 4:13 PM
File session notification!
Wireless AdHoc 4:1 PM
File session notification
Battery is fully charged. 4:13 PM
MANET Service 4:12 PM
PowerTutor 3:37 PM
Android debugging enabled
Select to disable debugging.
MANET Manager View Message
Done

Receiving a new request from [haz3test] to resume the (1418223704078) file session

Temperature 36.4°C, Voltage 4287mV.

Ad-Hoc mode is running.

Total Power: 1107 mW

From: haz3test
Message:
Receiving new file part from --> haz3test

B.3.3 Database queries (transactions)

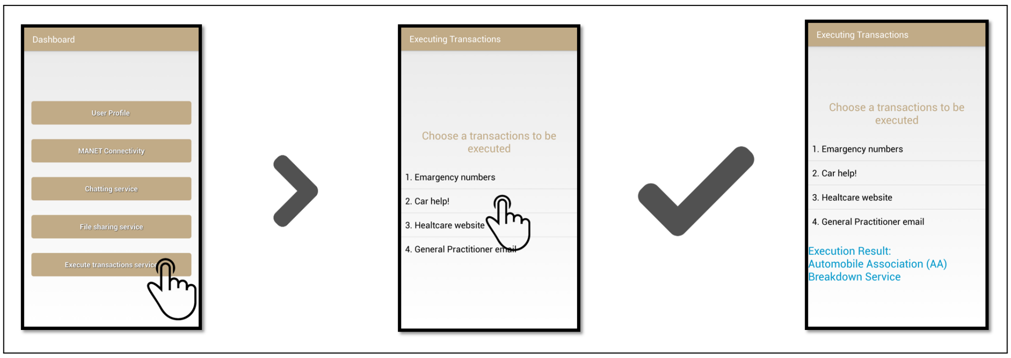
Users are offered a way to seek help from the app by sending a request that results in a database execution. A number of queries have already been added to the cloud database, linking to such information as the nearest medical centre or public emergency number. Two ways of using this service are offered in the app: sending a direct request to the cloud, or over a neighbouring link in case the user does not have an Internet connection.

Framework

There is an **Execute transactions service** button on the app ***dashboard screen*** that leads to a ***queries list screen***. A list of all the possible tags/queries accepted by the cloud are shown on this screen. By clicking on one of these queries, an HTTP request will be sent to the cloud for execution.

Once a request has arrived in the cloud, a look–up process will be performed to find the matching text/message for that query. The cloud then replies to the requester, attaching the text that is found in the database. Finally, the app presents the cloud response on the screen of the device.

The following screenshots show this framework:



Execution Result:
Automobile Association (AA)
Breakdown Service

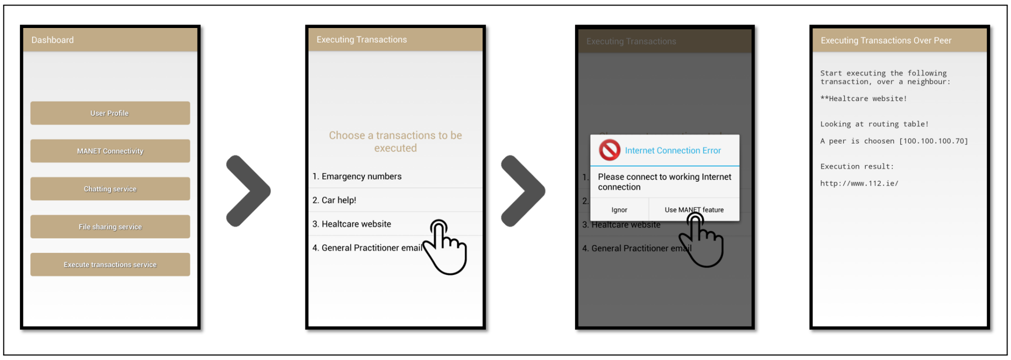
However, if the user does not have an Internet connection but is connected to a MANET, the app offers a way of executing a transaction over neighbouring links. The user will not have any control over routing the request or selecting the neighbouring node.

If the app deduces that the user cannot reach the cloud, it will redirect the request to one of the neighbours found in the routing table. That neighbour will receive the request, then redirect it to the cloud on behalf of the user.

Once a reply from the cloud is received, the neighbour will redirect the reply to the original user.

The following screenshots show how users can execute transactions even if no Internet connection is available:

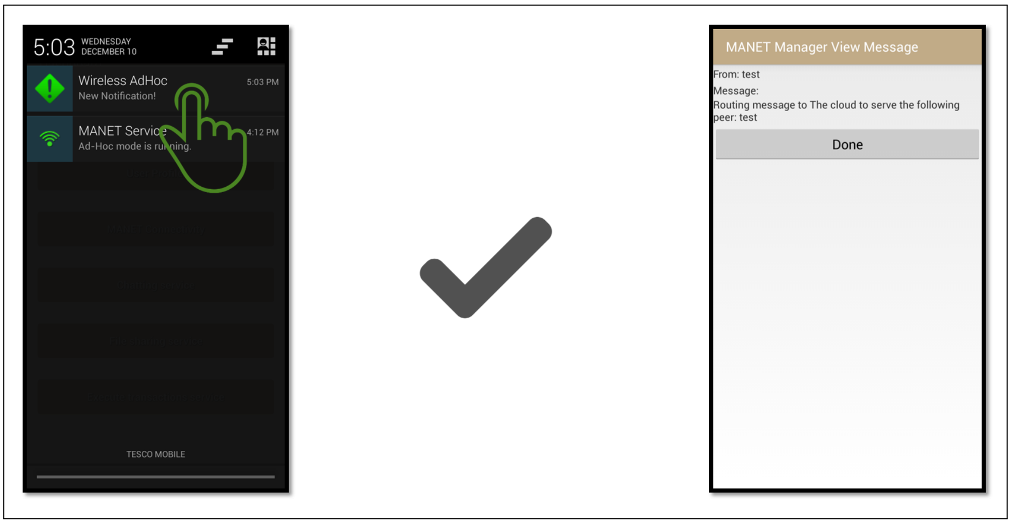
Requester side



Please connect to working Internet connection

```
Start executing the following
transaction, over a neighbour:

**Healtcare website!

Looking at routing table!

A peer is choosen [100.100.100.70]

Execution result:

http://www.112.ie/
```

Neighbour side



From: test
Message:
Routing message to The cloud to serve the following peer: test

B.3.4 Send a help request to the cloud

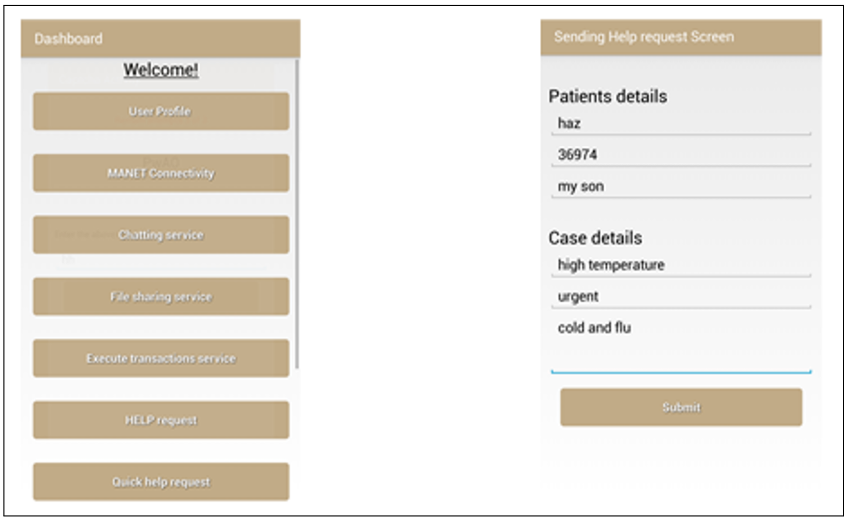
The app provides four ways of seeking help, depending on the requesting user's medical status. The following sections discuss these methods in detail.

B.3.4.1 Completing a form

From the **HELP request** button on the ***dashboard screen***, users can send a HELP request to look up a professional to discuss a medical case or to help another person who needs medical assistance (e.g., an injured child).

Clicking that button will start a ***sending help request screen*** that requires:

- Patient's details: whom this request is for, including EMR ID if the user has one.
- Case details: some information explaining the user's medical status, such as a keyword (e.g., "heart", "breathing" or "fainted").



Patients details

haz

36974

my son

Case details

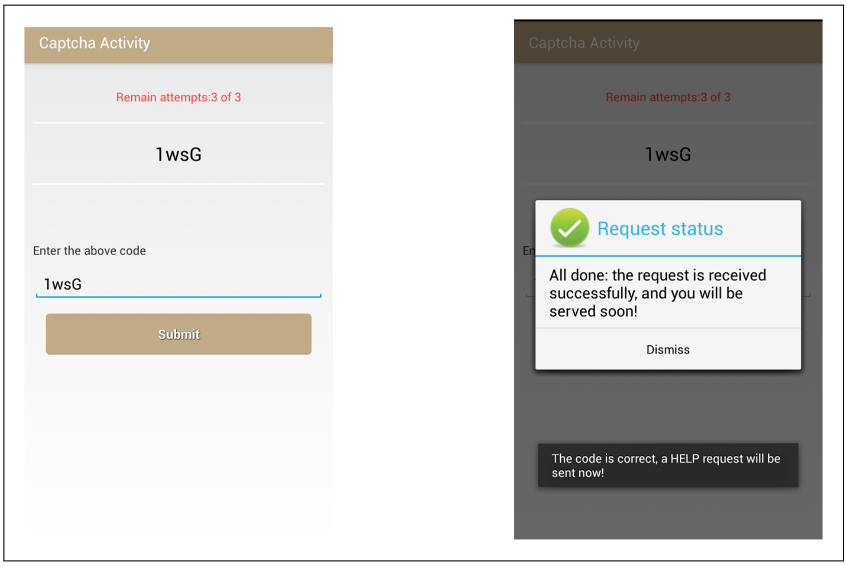
high temperature

urgent

cold and flu

To ensure that the help requests come from a real user and are not being used as a method of attack, the app requires the user to type a code that is generated randomly after completing the form. The user has only three attempts to enter the correct code and thereby proceed with the request.

The following screenshots show the implementation of this method:



Remain attempts:3 of 3

1wsG

Enter the above code

1wsG

Remain attempts:3 of 3

1wsG

Request status

All done: the request is received successfully, and you will be served soon!

The code is correct, a HELP request will be sent now!

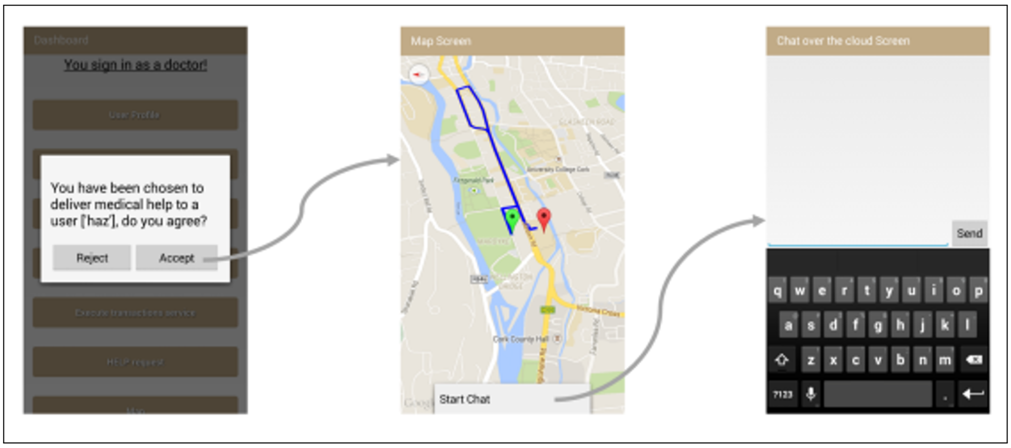
Next, in the background, the app collects the current location of the requester and connectivity status, such as to which MANET the user is connected. Then combines this information with what has been provided by the user and sends it as an HTTP request to the cloud to select the most suitable professional from the directory service.

This request will result in selecting a professional from the cloud. Here the app will pop up an alert on the selected professional's screen asking him/her to accept or reject the request. If the job is accepted, a ***map screen*** is started, showing directions to the requester using a Google Maps API, as well as the ability to start a chat screen with the requested user. Furthermore, a request will be sent to the cloud to acknowledge acceptance of the job and to notify the requester.

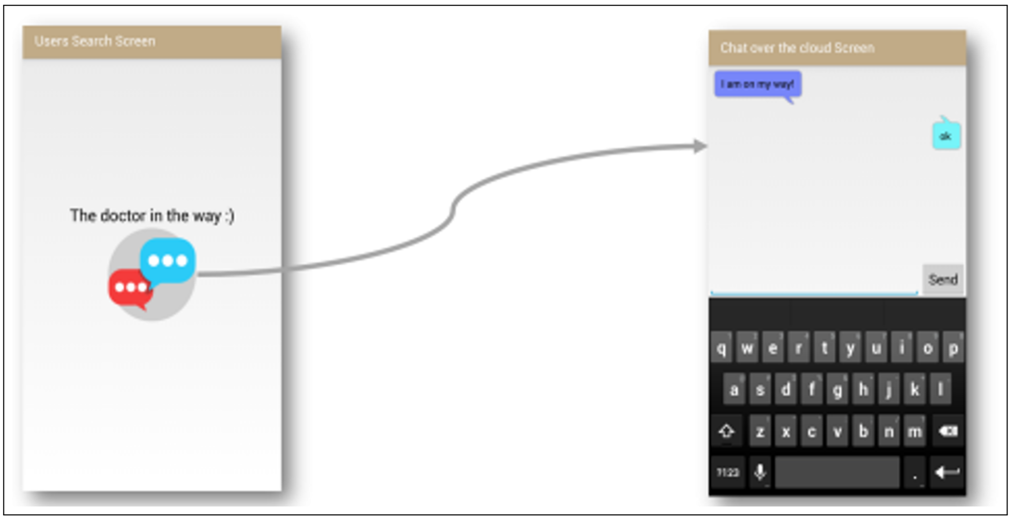
However, if the selected professional rejects the job, the cloud will look for another professional to deal with the case.

The following screenshots show what happens when a selected professional receives a new job to provide help to a user, as well as screenshots that display a notification on the requester screen that contains relevant updates of the help requested.

The selected professional's side



You have been chosen to deliver medical help to a user ['haz'], do you agree?

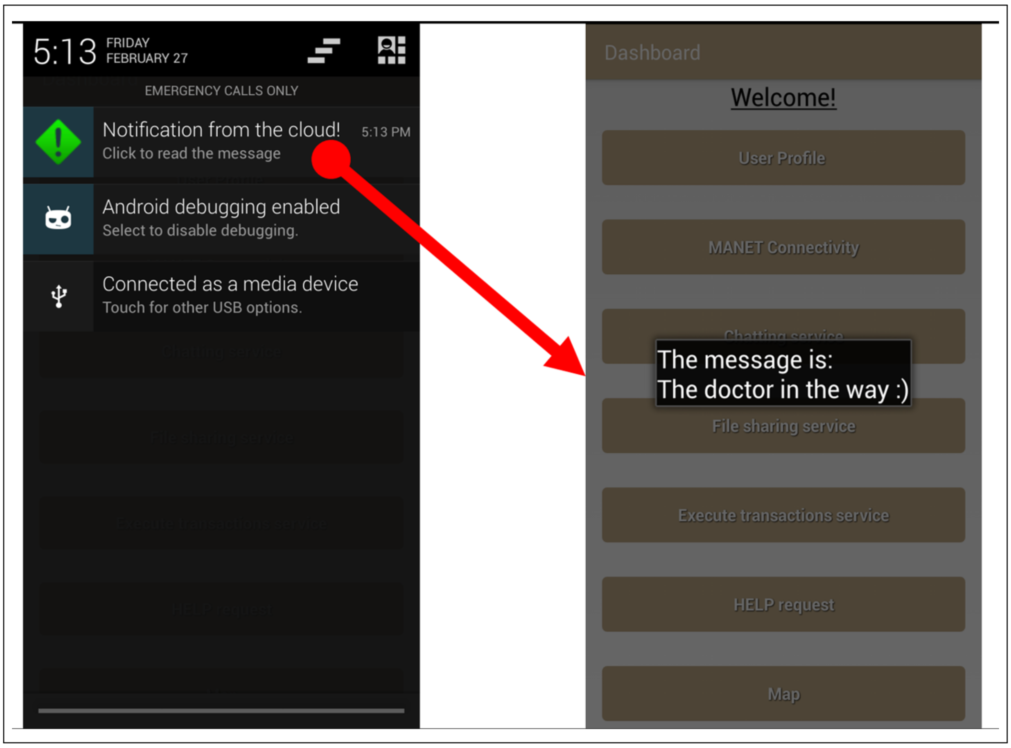


The doctor in the way :)

I am on my way!

ok

The requester's side



5:13 FRIDAY FEBRUARY 27

EMERGENCY CALLS ONLY

Notification from the cloud! 5:13 PM
Click to read the message

Android debugging enabled
Select to disable debugging.

Connected as a media device
Touch for other USB options.

The message is:
The doctor in the way :)

B.3.4.2 Sending a help request by SMS

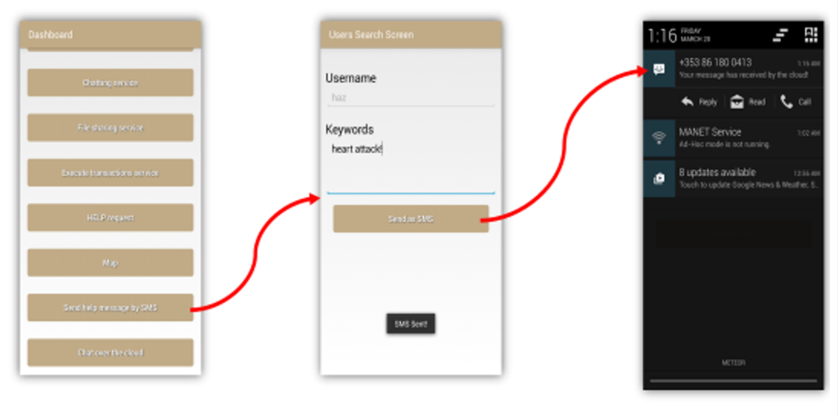
Using the **Send HELP request as SMS** button on the ***dashboard screen***, users can send a HELP request to the cloud as an SMS message. Clicking on this button will result in starting a new activity that allows users to type in some keywords regarding the help requested, whereby the username is brought from the shared preference that has the user account details.

In the background, the app constructs the HELP request as an SMS payload and sends it to the cloud via the Twilio platform. An acknowledgement will be sent to notify the user that the cloud has received the message successfully.

When the cloud selects a professional to interact with the requested case, a notification is sent to that professional, including the requester's contact information, for the purpose of starting an SMS conversation. The cloud also feeds the selected professional's details to the requesting user.

The following screenshots show how a user can seek help via SMS:

The requester's side



Username: haz

Keywords: heart attack

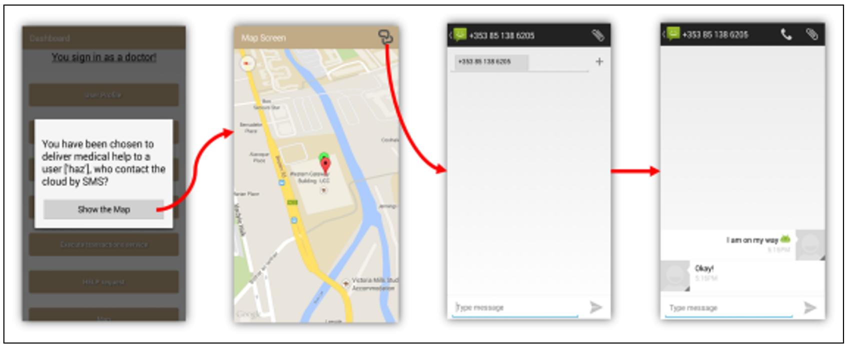
1:16 FRIDAY MARCH 28

+353 86 180 0413 — Your message has received by the cloud!

MANET Service — Ad-Hoc mode is not running.

8 updates available — Touch to update Google News & Weather, S...

The selected professional's side



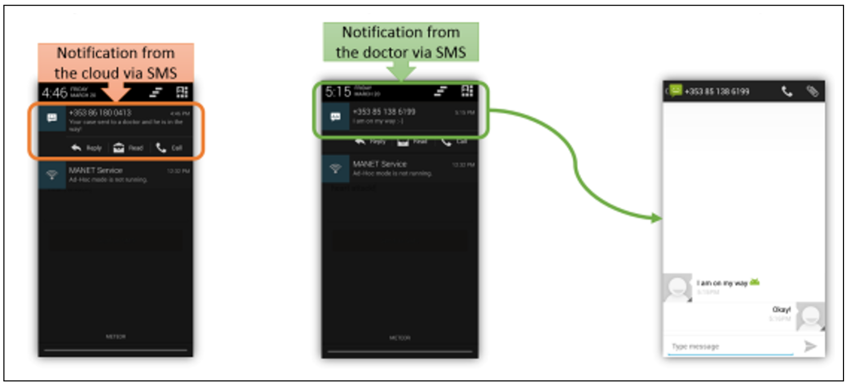
You sign in as a doctor!

You have been chosen to deliver medical help to a user ['haz'], who contact the cloud by SMS?

I am on my way

Okay!

The requester's side (notifications)


Notification from the cloud via SMS
Notification from the doctor via SMS
4:46
5:15
Reply
Read
Call
METEOR
+353 85 138 6199
Type message

+353 86 180 0413 — Your case sent to a doctor and he is in the way!

+353 85 138 6199 — I am on my way :-)

MANET Service — Ad-Hoc mode is not running.

I am on my way

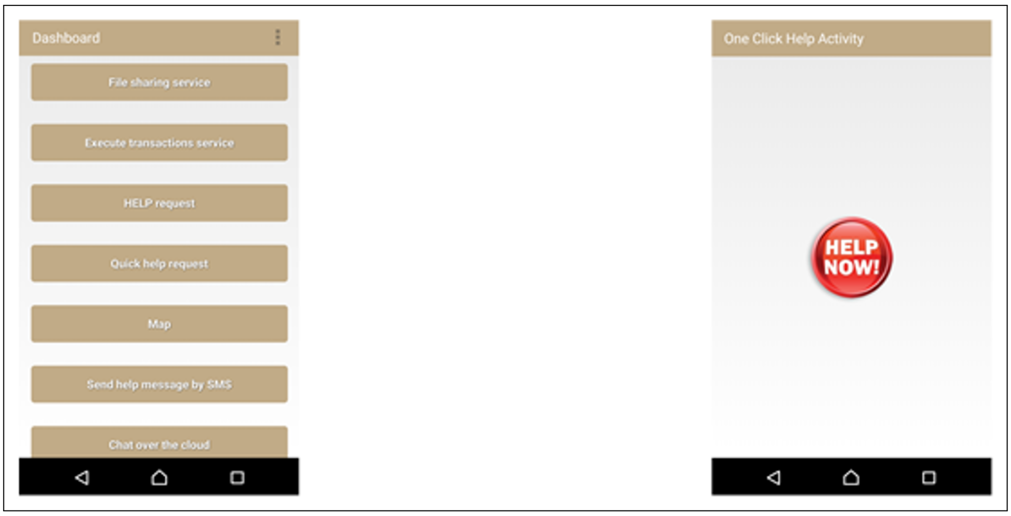
Okay!

B.3.4.3 Quick help

If a user clicks on the **Quick help request** button on the ***dashboard screen***, a ***One Click Help Activity*** will start that has only a **HELP NOW!** button.

Once this button is clicked, a request is sent to the cloud, which includes the user's ID and location, taking into account that the requester is at high risk. As a result, the cloud will redirect this information to a selected professional and emergency department to seek a swift and appropriate outcome.

The following screenshots show this service from the Android app:



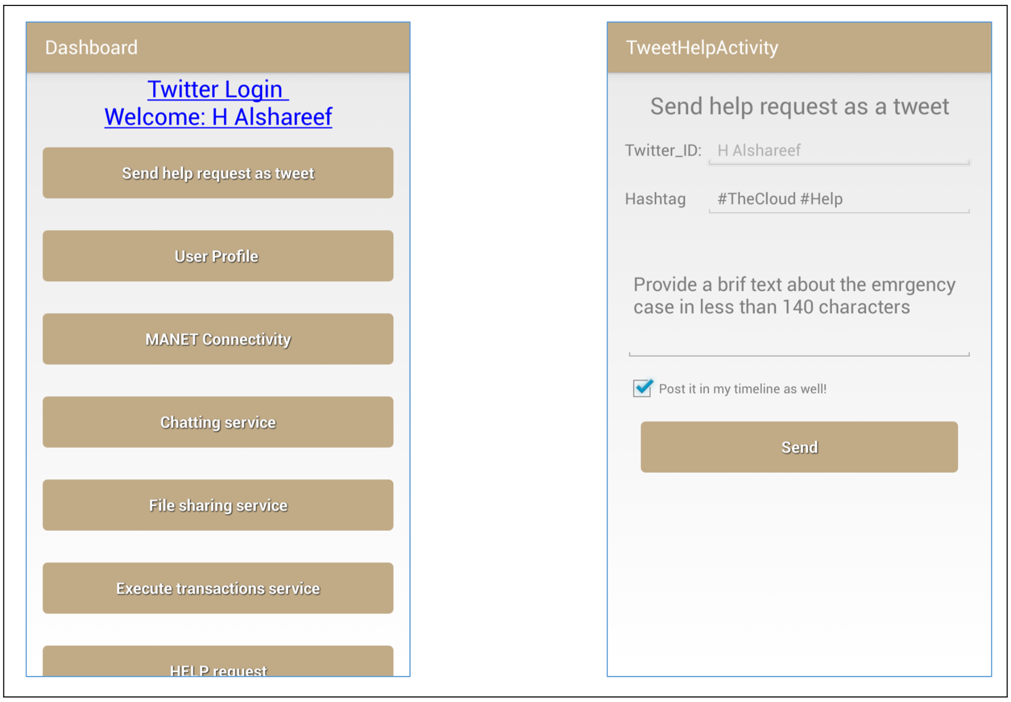
B.3.4.4 Tweet help

By clicking on the **Send help request as tweet** button on the ***dashboard screen***, users can send help requests as tweets. Users have to provide a brief description of the kind of help they need.

In the background, the app adds the #TheCloud hashtag, highlighting keywords, ensuring that the body of the text is below the limit (140 characters), and attaching current location.

Once the **Send** button is clicked, the request is tweeted and will show on the user's timeline. However, if the user has unchecked the ***Post it in my timeline as well!*** box, this will result in sending this request as an HTTP request to the cloud.

The following screenshots show how the user can use the app to send a help request as a tweet:



Twitter_ID: H Alshareef

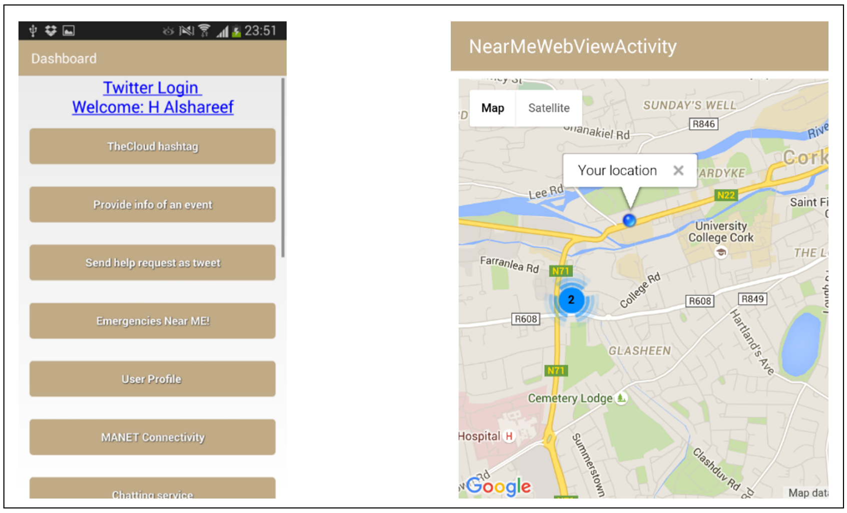
Hashtag #TheCloud #Help

Provide a brif text about the emrgency case in less than 140 characters

☑ Post it in my timeline as well!

B.3.5 Watching updates of nearby emergency events

The app allows users to watch updates from the cloud regarding emergency events. On the app, clicking on the **Emergencies Near ME** button on the ***dashboard screen*** will result in starting a web–view activity that contains an interactive map from the cloud showing any events near the user's location. In the background, the app sends the user's current location to the cloud, then receives permission to load an interactive map that is hosted on the cloud on which the user's current location is marked.

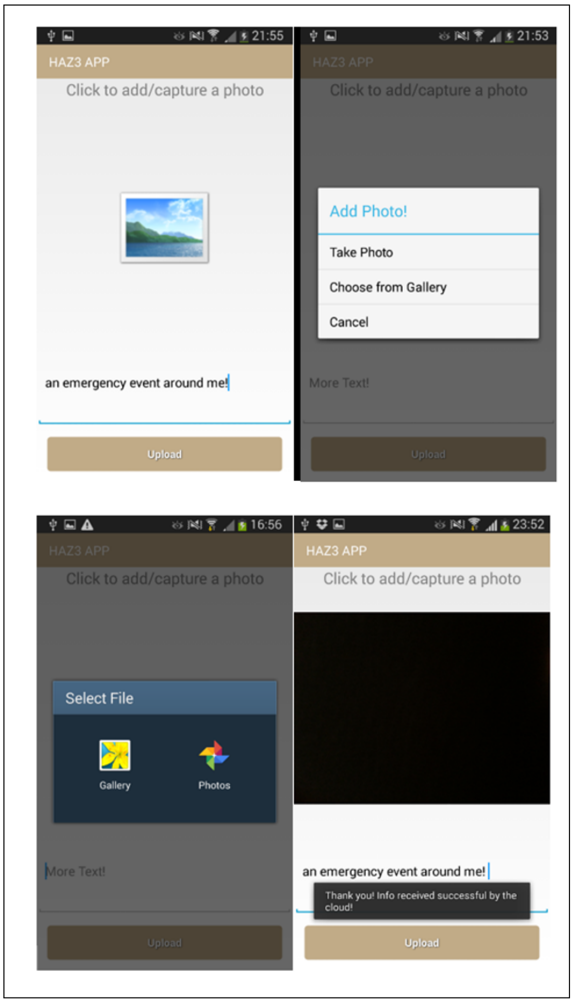


B.3.6 Reporting an emergency

The app allows users to report what they see on the ground with regard to emergency events. By clicking on the **Provide info of an event** button on the ***dashboard screen***, a new activity is started that allows users to capture an image on the camera or choose an existing one from the gallery, as well as adding some text to the photograph.

In the background, the app codes the image using the base64 coding scheme, then sends an HTTP request (that has the encoded photograph's string), the requester's information (username and location), and the text provided to the cloud. Once the cloud receives the request, it will acknowledge the sender. Acknowledgement will be shown in the app as a toast.

The following screenshots show how the user can provide on–the–ground information regarding an event using the Android app.



an emergency event around me!

an emergency event around me!

Thank you! Info received successful by the cloud!